WolfBot: A Distributed Mobile Sensing Platform for Research and Education

Joseph Betthauser, Daniel Benavides, Jeff Schornick, Neal O'Hara, Jimit Patel, Jeremy Cole, Edgar Lobaton

Abstract— **Mobile sensor networks are often composed of agents with weak processing capabilities and some means of mobility. However, recent developments in embedded systems have enabled more powerful and portable processing units capable of analyzing complex data streams in real time. Systems with such capabilities are able to perform tasks such as 3D visual localization and tracking of targets. They are also well-suited for environmental monitoring using a combination of cameras, microphones, and sensors for temperature, air-quality, and pressure. Still there are few compact platforms that combine state of the art hardware with accessible software, an open source design, and an affordable price. In this paper, we present an in-depth comparison of several mobile distributed sensor network platforms, and we introduce the *WolfBot* platform which offers a balance between capabilities, accessibility, cost and an open-design. Experiments analyzing its computer-vision capabilities, power consumption, and system integration are provided.**

Index Terms— **Distributed sensing platform, Swarm robotics, Open design platform.**

I. Introduction

Wireless sensor networks have been used in a variety of applications including surveillance for security purposes [30], monitoring of wildlife [36], [25], [23], and measuring pollutant concentrations in an environment [37] [24]. Initially, compact platforms were deployed in order to perform low-bandwidth sensing (e.g., detecting motion or recording temperature) and simple computations. Since then, new developments in embedded systems have enabled the design of compact, computationally powerful, and energy-efficient processing units which facilitate more complex data processing at each node. These systems are now capable of higher bandwidth processing of information collected from sensors such as video cameras. Furthermore, sensor networks have become mobile and integrated into ground [32], [19], aquatic [15], [34], [35], and aerial [28], [31], [21] platforms.

Manuscript received February 5, 2014. This work was supported by the National Science Foundation under award CNS-1239243 and the Army Research Office under award W911NF-12-1-0423.

J. Betthauser, D. Benavides, J. Schornick, N. O'Hara, J. Cole and E. Lobaton are with North Carolina State University, Cam-pus Box 7911, Raleigh, NC 27695, USA. J. Patel is with HM Electronics, Inc., 14110 Stowe Dr, Poway, CA 92064, USA. E. Lobaton (edgar.lobaton@ncsu.edu) is the corresponding author.

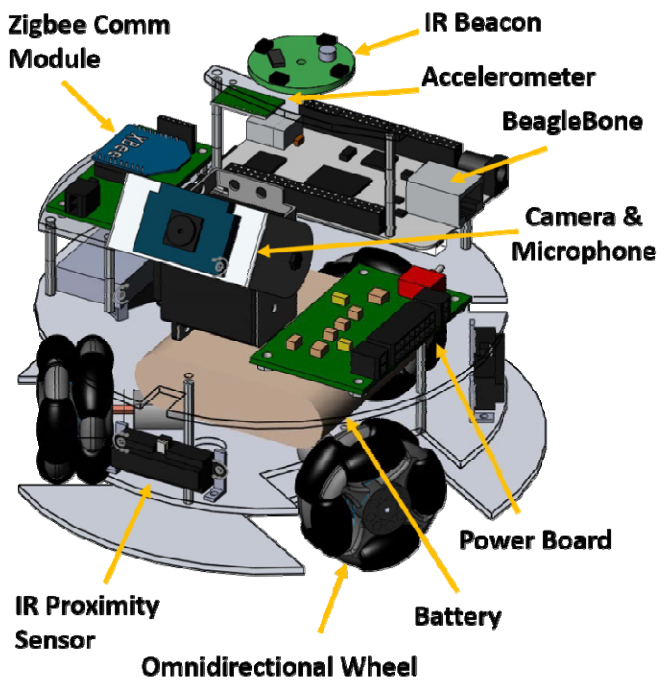

Fig 1.WolfBot mobile sensing platform and some of its features.

As the number of mobile devices increases, tools for distributed control and motion planning for swarm robotic systems are being incorporated [31].

An ideal hardware platform for mobile distributed sensing must combine the sensing and data aggregation capabilities from traditional wireless sensor networks with the coordination and control functionalities from swarm robotic systems. To be attractive as a research and educational tool, the platform must be affordable, and should provide an accessible open source software development interface as well as an open hardware design that facilitates modification and upgrade. While there is no one optimal design for all applications due to the interplay between these desired features, a variety of more specialized platforms are certainly feasible.

In this paper, we introduce the WolfBot platform (see Fig. 1), which represents a compromise between capabilities, accessibility, and cost while remaining competitive with other platforms in all three areas. This platform is a compact (17.5 cm in diameter) mobile sensing unit powered by a 1GHz ARM Cortex A8 processor capable of visual data acquisition and real-time processing; precise motion control via an omnidirectional drive system with on-board inertial and proximity sensing; wireless communication using WiFi and ZigBee; environmental sensing capabilities using on-board devices such as ambient light and distance sensors, as well as compatible external modules (enabling temperature, humidity, pressure, air quality sensing, etc.); and self-localization and agent identification enabled by an IR module and other visual tracking systems. In order to enable accessibility for educational and research purposes, the platform has an open design whose hardware schematics and bill of materials (approximately $550 per unit), with instructions for assembly, are available online [16]. The software is available in online repositories together with a MATLAB simulation environment for easy testing and implementation of algorithms. Furthermore, in order to facilitate software development, the platform provides a familiar software interface via its embedded Linux operating system, a Python [13] API, a system for distributed management of the platform, and access to OpenCV [6] for computer vision applications.

The rest of the paper is outlined as follows: section II provides an overview of state of the art of mobile sensor networks and swarm robotic platforms; sections III and IV outline our corresponding hardware and software design choices for this platform; experimental results are included in section V which highlight its on-board computer vision capabilities, a battery life of over 6 hours at full operational load, and its system integration for distributed sensing; and final remarks are provided in section VI.

II. RELATED WORK

There are a variety of mobile sensor network and swarm robotic platforms that can be used for distributed sensing applications. Table I shows a comparison of several systems that have similar features to the WolfBot, and discuss many more in this section.

The Kilobot [33] is a small robot from Harvard University. Its low-cost (roughly $14 in parts) allows it to be used in experiments requiring large numbers of agents. It has sensors capable of proximity sensing, but it has no imaging capability.

The R-One robot [27] from Rice University is designed to be an open-source swarm robot platform. It has a variety of sensors capable of proximity, inertial, and environmental sensing; however, it has no imaging capability.

The E-Puck [29] is a robot from EPFL designed for education. The E-Puck is an open-source platform with a 40x40 pixel camera. The robot is designed to be expandable, with several open-source extension boards that add additional sensing and communication abilities to the robot.

The Mindstorm EV3 [4] is a robot kit from LEGO. The kit contains a range of proximity and inertial sensors, and is designed to interface with LEGO blocks. The low cost of the robot kit (roughly $350) make it an attractive choice for education.

The Khepera III is a robot from K-team that is designed for research applications [3]. The Khepera's Kore-Bot II expansion module has a powerful embedded processor and support for USB cameras, allowing onboard image processing. The Khepara III costs approximately $3000 for the basic configuration, which can inhibit applications requiring large numbers of robots.

The MarXbot [19] is a wheeled robot from EPFL capable of moving over rough terrain. The robot has a large array of proximity, inertial, and environmental sensors, with the ability for self-assembly.

The PixHawk [28] is an aerial robot from ETHZ consisting of four 752x480 resolution cameras. It is capable of onboard image processing at 60 frames per second.

The Sensor Fly [31] from CMU is a small aerial robot designed for indoor sensing. It has several proximity and inertial sensors, but no imaging capabilities.

The Drifter [15] from UC Berkeley is an aquatic robot designed for mapping water flow and water sensing. It is equipped with GPS and environmental sensors, though it does not have imaging capabilities.

In addition to mobile sensing platforms, there are stationary sensing platforms such as the CITRIC camera mote from UC Berkeley [20].

The need for an affordable and compact mobile sensing platform capable of processing video in real-time, led us to

TABLE I
Mobile Sensing Platform Comparison

Name	Processor	Sensing	Comm.	Imaging	Cost
R-One	ARM Cortex-M3, 50 MHz, 64KB RAM	Accelerometer, Gyroscope, Bump, IR, Ambient Light	Radio (2 Mbps), IR (1.25 Kbps)	None	$220 (Parts)
E-puck	dsPIC 30F6014A, 64MHz, 8KB RAM	IR, Accelerometer, Microphone, Camera	Serial, Bluetooth, IR	40x40 pixels @ 4 FPS. External processing	$340 (Parts)
MarXBot	ARM11, 533 MHz, 128MB RAM	Camera, Accelerometer, Gyroscope, RFID, 2D Force, Microphone	WiFi, Bluetooth	Up to 2048x1536 pixel s @ 12 FPS	N/A
Khepera III	XScale, 600MHz, 128MB RAM	IR, Ambient Light, Ultrasonic, Camera	WiFi, Bluetooth	640x480 pixels @ 15 FPS	>$2000
CITRIC	XScale, 624 MHz, 64MB RAM	Camera, Microphone	Zigbee	1280x1024 pixels @ 15FPS	$600
WolfBot	ARM Cortex-A8, 1GHz, 512MB RAM	IR, Camera, Microphone, Ambient Light, Accelerometer, Magnetometer	WiFi, Zigbee	Up to 1280x720 pixels @ 50 FPS	$550 (Parts)

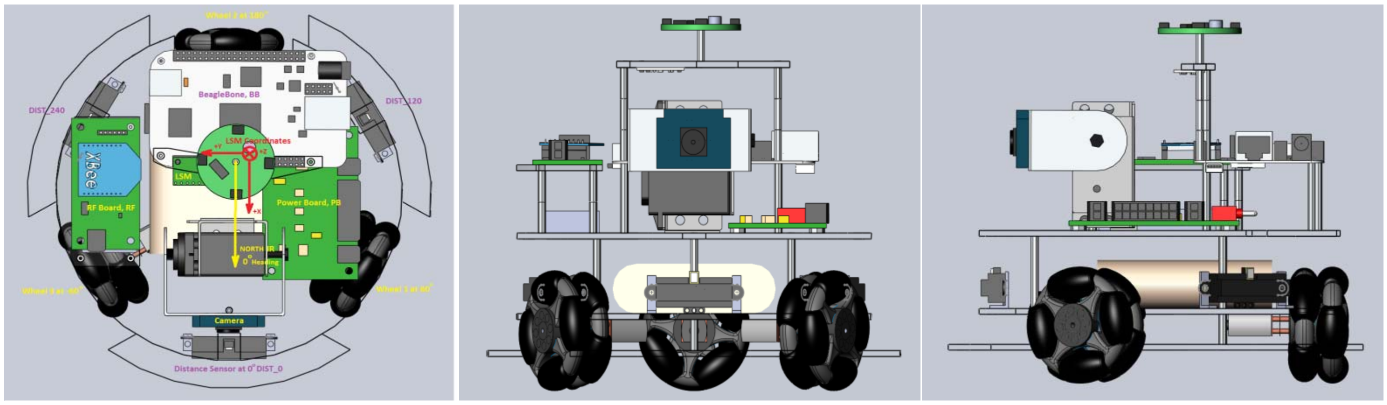

Fig 2. WolfBot's CAD design: top view (left), front view (middle), and side view (right).

build the WolfBot which has a powerful ARM Cortex A8 processor, a 720p HD resolution camera, robust communication protocols, sensors (3D accelerometer and magnetometer, ambient light sensors, distance sensors), and the capability for real time image processing and online video streaming and a low unit cost (under $550 in parts). The MarXBot and the Khepera III are platforms with similar capabilities to the WolfBot. However, our platform is better suited for computer vision tasks due to its more powerful processor, and its open design makes its available for educational and research purposes.

III. HARDWARE DESIGN

The design has been optimized to be compact, highly maneuverable using an omni-directional drive system, and possesses a suite of inertial, visual, and environmental sensors on-board. Fig. 2 and 3 shows CAD model views and actual views of the WolfBot. The main features considered during design are described below.

A. *Embedded Computing*

The WolfBot is built around the BeagleBone Black development platform. The BeagleBone is a compact single board computer (SBC) with a TI AM335x (ARM Cortex-A8, up to 1 GHz) processor [1]. With similar embedded computing capabilities and at a lower cost, the Raspberry Pi [9] could be an alternative to the BeagleBone. However, the Raspberry-Pi has 26 GPIO pins, whereas the Beaglebone Black has 92 GPIO pins, which is a critical distinction when considering the amount of hardware on the WolfBot. The choice of BeagleBone over Raspberry-Pi was based on its lower power consumption, higher number of GPIO pins, and a more powerful NEON Advanced SIMD unit in the BeagleBone Black's processor. For our design, the performance advantages outweigh the cost difference. Table II shows a comparison between these two options.

TABLE II
Comparison of BeagleBone Black and Raspberry Pi

Feature	BeagleBone Black	Raspberry Pi
Model	Rev A5A	Model B
Proc. Type	ARM Cortex A8	ARM 11
Proc. Speed	1 GHz	700 MHz
RAM	512 MB DDR3L	512 MB
GPY	SGX530 3D	Videocore 4
VFP version	NEON	VFPv2
Min Power	1.05W (210mA)	3.5W (700mA)
Cost	$45	$35

B. *Mobility*

An omnidirectional drive system for full translational and rotational control was chosen, which uses inertial sensing and IR proximity sensing for feedback and obstacle avoidance. The motors are controlled using a custom built motor control

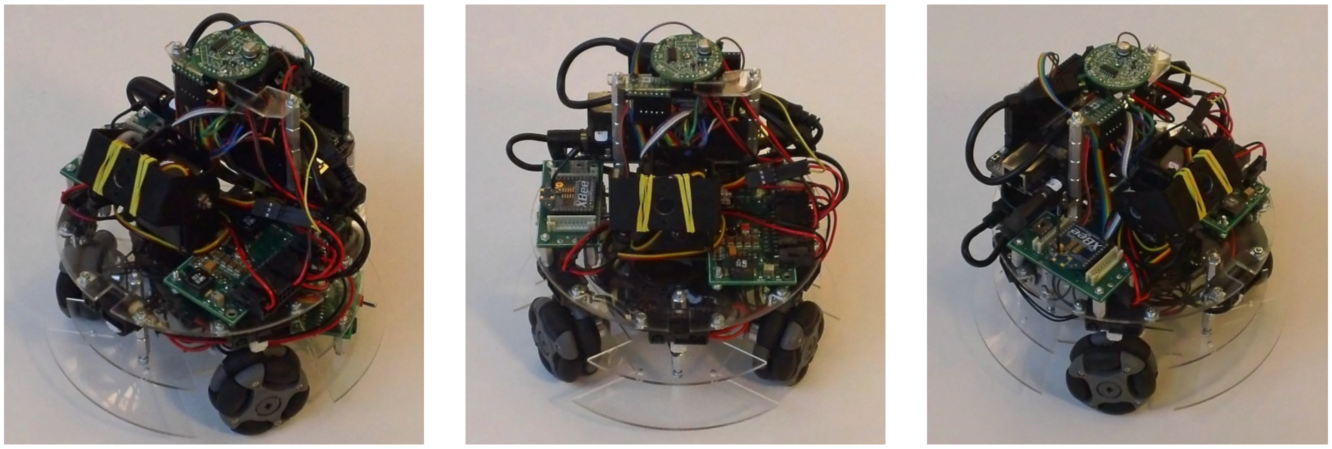

Fig 3. WolfBot's Physical design: isometric view (left), front view (middle), and alternate isometric view (right).

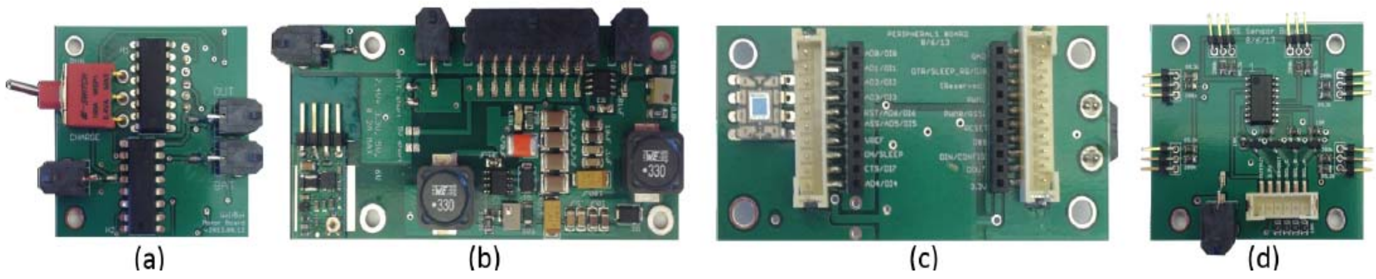

Fig 4. PCB boards designed: (a) Motor Control, (b) Power Distribution, (c) Peripheral, and (d) DMS.

board (see Fig. 4) which integrates H bridges for speed control of the motors, a power switch for the robot, and a voltage output for direct monitoring of the battery.

C. Sensing

There are several sensors on-board used for environment sensing as well as motion feedback:

Navigation Sensors. A 3D accelerometer provides the inertial measurements for overall evaluation of the robot's motion. A magnetometer is used to get a heading. IR based distance measurement sensors have been included in the platform which can be used for obstacle detection. The selected distance sensors provide granular measurements between 15 and 100 cm and binary obstacle detection below 15 cm. Six of these distance sensors are installed, evenly spaced, around the robot perimeter.

Environmental Sensing. An on-board ambient light sensor detects light intensity in the surrounding area. The BeagleBone provides a USB port and other standard IO ports for easy interface to sensing modules such as the SensorDrone [12], a compact (less than 7 cm in length, 3 cm in width and 1.3 cm in thickness) environmental sensing unit, which is capable of measuring temperature, pressure, and air quality, among others. This module costs $200 per unit.

Visual Sensor. The camera mounted on the robot provides image capture at 720p HD resolution. A servo motor is used to position the camera at different angles (from 0 to 90 degrees). The microphone in this camera can be used for audio recording and audio source localization applications. The platform is capable of storage, streaming, and real-time processing of video-streams.

D. Communication

WiFi and ZigBee technologies have been incorporated into our design. These two modalities offer alternatives between high bandwidth with high power consumption, and lower bandwidth with lower power consumption.

The selected WiFi adapter provides communication with a data rate up to 150Mbps. This method of communication is used for file transfer and real-time video streaming from the onboard camera. The peak current draw for the WiFi adaptor is approximately 150mA.

ZigBee communications has been implemented using low power XBee modules which provide a data rate up to 250 kbps. The ZigBee modules can establish a mesh network among several WolfBots and can be used to broadcast messages, data, or commands to the entire group. The effective range for these devices is around 41 m indoors and 122 m outdoors with a maximum current draw of 45 mA.

E. Localization

Pololu's IR transceiver beacon [8] with range from 15 cm to 5 m is used to provide line of sight identification and localization. The sensor provides a relative cardinal (north, south, east, west) orientation between agents. Because the beacon operates at specifically 56kHz frequency, it is effectively isolated from noise from other infrared light systems.

More precise localization can be achieved using visual SLAM algorithms [21] or using top-view camera systems (e.g., static camera networks or aerial cameras). We plan to provide support for these features using existing libraries in the near future. Currently, we are providing a software interface for the OptiTrack [7] Tracking Tools system, which uses infrared cameras to determine 3D position of markers that can be easily attached to individual WolfBots. By using WiFi to interface with the OptiTrack system, each agent can get their global coordinates streamed in real-time.

F. Power Management

The WolfBot uses a 7.4V Lithium-ion battery with a 5200mAh (38.48 Wh) capacity. Two separate TPS5420 DC-DC converters are used to step down the supply voltage to 5V and 3.3V, and a voltage regulator provides 6V. Although the power management is capable of providing a maximum of 2 Amps in current, the WolfBot's current draw peaks at 1.6 A with a much lower average draw under full-load. Per the testing detailed in section 6, this ensures a battery life of at least 6 hours of continuous operations at high performance.

The designed Power Distribution Board (see Fig. 4) regulates and distributes all voltages (3.3V, 5V, and 6V) required by the WolfBot. It also breaks out these signals to a number of separate lines to simplify routing power to various boards and peripherals. The single input to the board is an unregulated 6.4V - 8.4V signal from the motor control board which is provided from the source battery.

G. PCB Integration

In order to operate the above devices and sensors, four circuit boards had to be designed (as seen in Fig. 4): power distribution, motor control, peripheral interface, and distance measurement sensor (DMS) boards. The power and motor control boards have already been discussed.

The peripheral interface board is used to create an interface between the BeagleBone Black and the Wolf- Bot's various sensors and communication devices. This includes an XBee socket and breakout, a light sensor with support circuitry, and logic level shifting for the IR beacon.

The DMS board allows up to six IR distance sensors to be connected to the WolfBot. Primary features of the board include: Multiplexing the six sensor outputs to a single BeagleBone ADC, level shifting of analog sensor output (5V to 1.8V), level shifting of digital control signals (3.3V to 5V).

H. *Open Design and Cost*

The list of components, schematics for PCB boards, CAD design for the entire platform, and assembly instructions including videos are available online [16]. The cost of components per robot is under $550 for a single unit. As part of an educational activity, assembly can be integrated as a class project for undergraduate students.

IV. SOFTWARE AND MANAGEMENT

A key design goal of the WolfBot platform is to maximize accessibility to researchers. This goal has been realized in three parts: simplified access to hardware (sensors, actuators, communications), compatibility with popular software libraries (e.g. the Open Source Computer Vision (OpenCV) library [6] and Robot Operating System (ROS) [10]), and ease of scaling experiments to run on large groups of WolfBots.

A. *Accessible Hardware Interface*

As described in section 3, the WolfBot hardware platform includes many off-the-shelf sensors, actuators, and communications systems. At a low level, many of these devices interface with the BeagleBone Black in entirely different ways (analog versus I2C, etc.). A priority was placed on eliminating these details and providing a simple, consistent software interface.

The *Python* [13] programming language was chosen as the primary tool to develop this interface. It is a high level, object-oriented language known for its simplicity and extensibility; thus it has been gaining general popularity [14] and support in the scientific community.

On each WolfBot, scripts are only responsible for instantiating a single WolfBot object. The WolfBot class handles the instantiation of additional objects which are responsible for hardware access. To illustrate:

```
import wolfbot
w = wolfbot.wolfbot()
a = w.accel.read()
print "X:␣%d,␣Y:␣%d,␣Z:%d" % a
```

This reads the accelerometer and produces a tuple of 3-D acceleration values:

```
X: -448, Y: -64, Z: -17216
```

In addition, instantiation of the WolfBot class processes a variety of parameters via a YAML-based [17] configuration file. This structured file format allows each WolfBot to load robot-specific parameters such as sensor calibration information for all WolfBots.

B. *Additional Library Support*

Accessibility of the WolfBot platform is also highly dependent upon the availability and easy integration of popular libraries, such as OpenCV. Two choices in the WolfBot design address this: the choice to run a fully featured operating system and the choice of Python as the core language.

In contrast to less powerful platforms, the hardware choice of a modern ARM-based processor allows each of the WolfBots to run a fully-featured Linux-based operating system. The BeagleBone is shipped with Angstrom Linux, but additional work was done to deploy Debian Linux 7.1 to the device. As a result, thousands of popular software packages are immediately available via Debian Software Repository and run unmodified on the WolfBot. Simplified installation may be accomplished using the management interface described in Section IV.C.

Python has an extensive collection of mature libraries available for most common computational tasks. For example, the popular *NumPy* [5] package includes a set of linear algebra tools used extensively for kinematic motion calculations processed locally on the WolfBot.

C. *WolfBot Management*

As the number of WolfBots grows, care must be taken to ensure consistency and efficient control of the robots as a group. This issue is directly addressed through the development of the Packmaster component of the platform.

The Packmaster is a stand-alone Linux system, currently deployed on a spare BeagleBone. At the lowest level, the Packmaster provides the network infrastructure necessary for direct command and control of every WolfBot. This is employed as an isolated WiFi network, allowing the Packmaster to manage the assignment of names and IP addresses to individual WolfBots in a consistent manner via the Dynamic Host Control Protocol (DHCP). This network is also available to other clients such as laptops, providing access to individual WolfBots.

Because the Linux system on each WolfBot is complex, there is potential for system divergence, resulting in inconsistent or unexpected behavior during experimentation. This concern is addressed via the use of the SaltStack (“Salt") infrastructure management system [11]. Salt's benefits to the project are twofold; it provides both configuration management and remote execution.

State management ensures that registered files, software packages, and services are deployed consistently across all targets. A core configuration is defined on the Packmaster, and Salt is used to deploy that configuration to each registered WolfBot with a single command.

Furthermore, Salt provides a convenient remote execution system to control arbitrary groups of WolfBots during experimentation. Once software and configuration has been deployed via the aforementioned state management, Salt can be used to start the necessary program. While this results in reasonably parallel execution (<1s), tighter coordination is also available via network time synchronization provided by the Packmaster (<10ms).

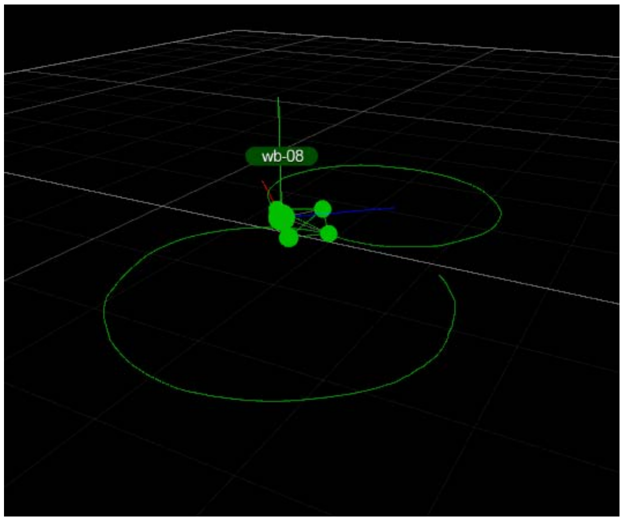

Fig 5. Recorded motion of the WolfBot tracing a figure 8.

D. *Physical Modeling*

A standard kinematic and dynamic equations are used to provide a physical model for the WolfBot platform based on the work by Y. Liu et al. [26]. Fig. 5 illustrates a sample motion recorded using an OptiTrack System in which the WolfBot is sent motion commands to trace a figure eight. Fig. 6 shows the output of a MATLAB's Simulink module made available for simulating simple motion algorithms.

E. *Open Source*

As with the hardware design, all aspects of the software and management system will be released under an open license and made available through a GitHub repository [16]. Documentation, tutorials and class projects will also be maintained in this website.

V. Experimental Results

In order to test and evaluate the capabilities of our platform, we performed three sets of experiments. The first set aims to test and characterize the image processing capabilities of the platform by executing image processing routines such as edge detection and more complex routines such as face detection.

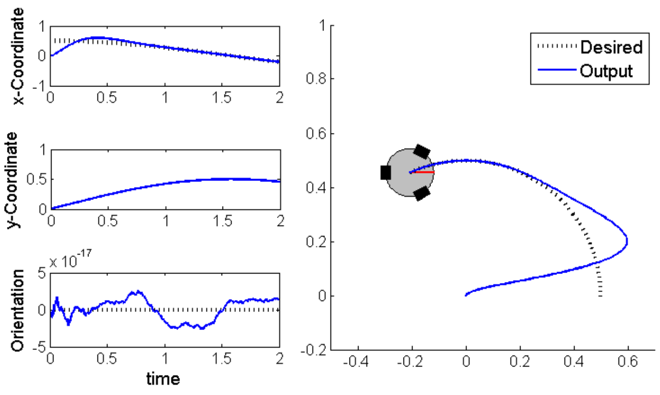

Fig 6. Motion simulation of robots following circular path. Initial overshoot is due to an offset on initial conditions. Time is in seconds and *xy* coordinates are in meters.

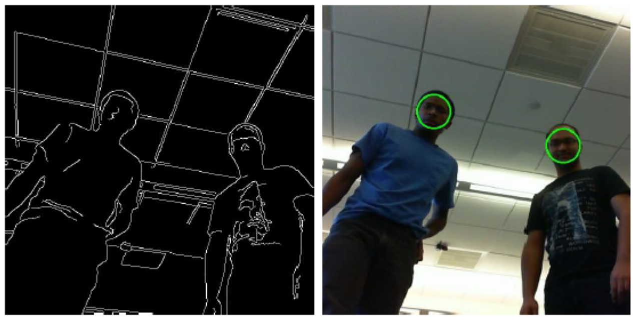

Fig 7. Experimental results for Computer Vision Evaluation on an image from WolfBot camera: Canny edge detection (left), and face detection using Haar cascades (right).

The second experiment characterizes the power consumption under a variety of loads. The final experiment validates the WolfBot's effectiveness as a mobile sensor network by performing a scalar field estimation.

A. *Computer Vision Evaluation*

OpenCV 2.4 has been installed in our platform in order to enable the fast development of computer vision applications. In order to evaluate the performance of the platform, we measured the time required to execute edge and face detection on images with a resolution of 640x480.These are two common tasks often used in computer vision applications. Fig. 7 shows some outcomes from these experiments.

The execution of the canny edge detector for the first task took an average of 43ms per image (≈ 23.3 frames per second). For comparison, the same function in the CITRIC platform [20] using IPP [2], a library optimized for Intel processors, takes 340ms on 512x512 images.

Face detection was implemented using the Haar cascades feature detector from OpenCV. The execution time was an average of 480ms per image.

B. *Power Consumption*

The WolfBot's power consumption is tested in three scenarios to simulate a range of workloads. In each case, the WolfBot's 5200mAh battery was fully charged and allowed to run down until operation ceased. Battery voltages were recorded periodically by the WolfBot's onboard battery sensor.

The maximum power draw configuration included nonstop high-speed movement to OptiTrack specified locations, full CPU utilization, constant WiFi data streaming, and repeated camera repositioning.

The mid-power case was similar, but included 20 second pauses between motions and only one camera reposition per move. CPU and WiFi remained in constant use.

The final test maximized battery life, though performed without putting the system to sleep. There was no motion, the CPU was nearly idle, and WiFi was disabled.

TABLE III
Summary of power consumption test results

Load	**Avg. Power**	**Avg. Current**	**Runtime**
High	6.27 W	0.86 A	6.14 h
Mid	4.41 W	0.60 A	8.72 h
Low	3.48 W	0.47 A	11.05 h

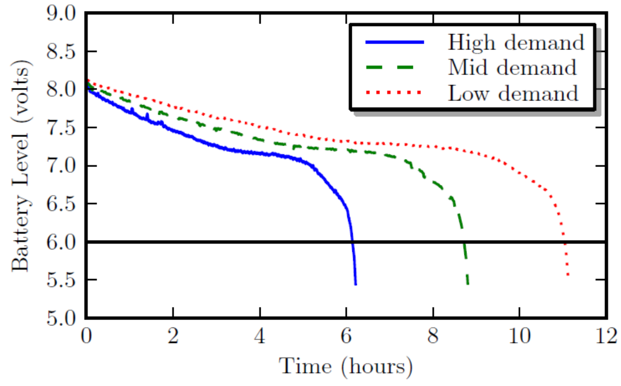

Fig 8. Battery level over time. Horizontal line represents discharge cutoff per manufacturer.

Even in the highly demanding scenario, the WolfBot was observed to run for over 6 hours before failure. A maximum runtime of over 11 hours was also observed. Fig. 8 and Table III detail our results. Runtime was selected based on a 6.0V discharge cutoff per the battery specifications; power and current are based on this period and a 38.48Wh battery capacity.

C. *Scalar Field Estimation*

Our third experiment demonstrates the suitability of the WolfBot platform as a mobile sensor network. It additionally highlights the effective integration of software, hardware, and swarm management components described in Section IV. Specifically, we perform a scalar field estimation of light intensity using multiple Wolf Bots to distribute the workload.

The test area was prepared as shown in Fig. 9 (left), where a flashlight and an area lamp were arranged around the perimeter of the OptiTrack viewing space. These light sources were chosen to produce two distinct light intensity gradients within the field. The OptiTrack camera system was used to provide localization information to each WolfBot via the Packmaster.

Using the WolfBot's hardware interface library, a short Python script was written to autonomously direct each WolfBot along a path using this explicit localization data. At each target along the path, the script reads and logs the value of the onboard light sensor. For the path, a single set of coordinates was generated which describe a simple back-and-forth survey pattern.

The above script and survey path file were loaded onto the Packmaster for deployment. Using the platform's management tools, the files were synced to two WolfBots configured with OptiTrack markers. A subsequent command from the Packmaster instructed each WolfBot to begin execution of the survey.

The pair of WolfBots divided the predetermined path in two, each surveying half the field as seen in Fig. 9 (top-right). The recorded light intensity data was collected by the Packmaster for offline processing. The resulting estimated light intensity field, after some Gaussian spatial filtering, is shown in Fig. 9 (bottom-right).

It should be stressed that the described experiment could be scaled to any reasonable number of WolfBots with no additional effort.

VI. Conclusion

In this paper, we introduced an open-design mobile sensing platform for research and education capable of real-time computer vision processing. The performance of the platform is analyzed by testing its computer vision capabilities, power consumption and system integration.

Future software versions will be integrated with the Robot Operative System (ROS) library [18] for easy integration with other robotic platforms. To ensure open-design availability for the WolfBot platform, additional steps are being taken to

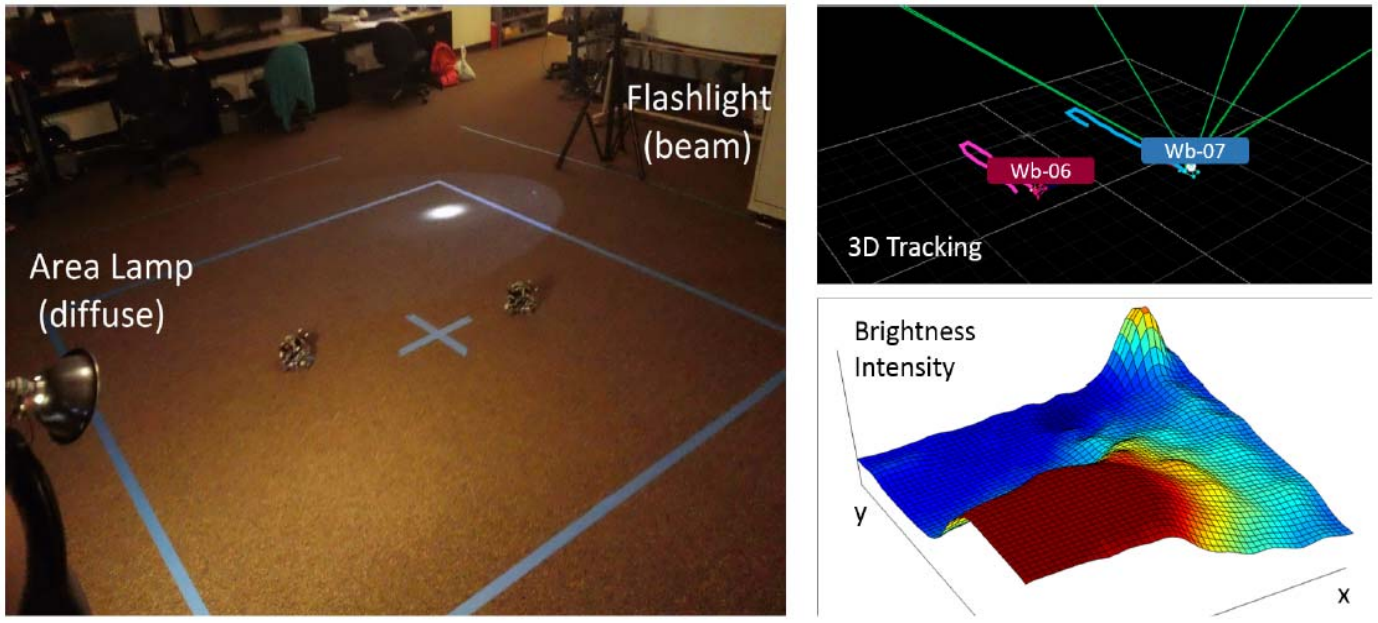

Fig 9. Light intensity field estimation: setup (left), partial robot paths (top-right) and estimation results (bottom-right).

ensure that all aspects of hardware and software design are included within a public online repository. As the project expands, researchers can amend or add to the existing documentation as necessary.

Acknowledgment

The authors would like to thank Alexander Youngs, Anthony Arazo, Benjamin Cancel, Matt Green, Snigdha Kotta, Ekdeep Rajpal, Charles Teague, and Sachin Patel for assistance in design, testing, and debugging for this platform.

References

[1] BeagleBone Black. http://beagleboard.org/Products/BeagleBone%20Black
[2] Intel Integrated Performance Primitives (Intel IPP) 7.1. http://software.intel.com/en-us/intel-ipp.
[3] K-Team Mobile Robotics. www.k-team.com
[4] LEGO Mindstorm EV3. http://mindstorms.lego.com/en-us/default.aspx
[5] NumPy. http://www.numpy.org/
[6] Open Source Computer Vision (OpenCV) Library. http://opencv.org.
[7] OptiTrack – Optical Motion Capture Systems and Tracking Software. http://www.naturalpoint.com/optitrack/
[8] Pololu: Robotics and Electronics. http://www.pololu.com/catalog/product702.
[9] Raspberry pi. http://www.raspberrypi.org
[10] Robot Operating System. http://ros.org.
[11] SaltStack. http://saltstack.org.
[12] SensorDrone. http://www.sensorcon.com/sensordrone/
[13] The Python Programming Language. http://www.python.org/.
[14] TIOBE Programming Community Index Oct 2013. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
[15] UC Berkeley Drifters. http://float.berkeley.edu/.
[16] WolfBot. http://research.ece.ncsu.edu/aros/project/wolfbot/.
[17] YAML Ain't A Markup Language. http://yaml.org
[18] Robotic Operating System. http://www.ros.org/
[19] M. Bonani, V. Longchamp, S. Magnenat, P. Retornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada. The marxbot, a miniature mobile robot opening new perspectivese for the collective-robotic research. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on,* pages 4187-4193, 2010.
[20] Chen, P.; Ahammad, P.; Boyer, C.; Shih-I Huang; Leon Lin; Lobaton, E.; Meingast, M.; Songhwai Oh; Wang, S.; Posu Yan; Yang, A.Y.; Chuohao Yeo; Lung-Chung Chang; Tygar, J. D.; Sastry, S.S., "CITRIC: A low-bandwidth wireless camera network platform," *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on* , vol., no., pp.1,10, 7-11 Sept. 2008
[21] P. DeLima, G. York, and D. Pack. Localization of ground targets using a flying sensor network. In *Sensor networks, Ubiquitous, and Trustworthy Computing, 2006. IEEE International Conference on*, volume 1, pp. 194-199, 2006.
[22] H. Durrant-Whyte and T. Bailey. Simultaneous Localization and mapping (SLAM): Part I the essential algorithms. *Robotics and Automation Magazine*, 13(2):99-110, 2008.
[23] Z. He. Energy-efficient integrated camera and sensor system design for wildlife activity monitoring. In *Multimedia and Exp, 2009. ICME 2009. IEEE International Conference on*, pages 1580-1581, 2009.
[24] T.-S Lin, H.-C Lu, J.-H Liu, J.-A Jiang, T.-H. Wen, C.-H Sun, and J.-Y. Juang. Application of a reliable mac protocol for the urban air quality monitoring system based on the wireless sensor network. In *Southeastcon, 2012 Proceedings of IEEE,* pages 1-6, 2012.
[25] A. Lindgren, C. Mascolo, M. Lonergan, and B. McConnell. Seal-2-seal: A delay-tolerant protocol for contact logging in wildlife monitoring sensor networks. In *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on, pages 321-327, 2008.*
[26] Y. Liu, J. Zhu, R. W. II, and J. Wu. Omni-directional mobile robot controller based on trajectory linearization. *Robotics and Autonomous Systems,* 56:461-479,2008.
[27] J. McLurkin, A. Lynch, S. Rixner, T. Barr, A. Chou, K. Foster, and S. Bilstein. A Low-Cost Multi-Robot System for Research, Teaching, and Outreach. In *Proc. Of the Int. Symp. On Distributed Autonomous Robotic Systems,* 2010.
[28] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys. Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision. *Autonomous Robots,* pages 1-19. 10.1007/s10514-012-9281-4.
[29] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptoez, S. Magnenat, J. Christophe Zufferey, D. Floreano, and A. Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions,* pages 59-65, 2009.
[30] R.J. Mullen, D. N. Monekosso, S. A. Barman, and P. Remagnino. Autonomous control laws for mobile robotic surveillance swarms. In *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA),* number Cisda, pages 1-6, IEEE, July 2009.
[31] A. Purohit, Z. Sun, F. Mokaya, and P.Zhang. Sensorfly: Controlled-mobile sensing platform for indoor emergency response applications. In *Information processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 223-234, 2011.
[32] J. Röning, J. Haverinen, A. Kemppainen, H. Mösäri, and I. Vallivaara. Smart System for Distributed Sensing. In *International Biennial Baltic Electronics Conference (BEC),* pages 1-10, 2008.
[33] 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*, pages 3293-3298. IEEE, May 2012.
[34] M. Stealey, A. Singh, M. Batalin, B. Jordan, and W. Kaiser. Nims-aw: A novel system for autonomous sensing of aquatic environments. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on pages 621-628, 2008.*
[35] P. Tokekar, E. Branson, J. Vander Hook, and V. Islar. Tracking aquatic invaders: Autonomous robots for monitoring invasive fish. *Robotics Automation Magazine, IEEE,* 20(3):33-41, 2013.
[36] A. Tovar, T. Friesen, K. Forens, and B. McLeod. A dtn wireless network for wildlife habitat monitoring. In *Electrical and Computer Engineering (CCECE), 2010 23rd Canadian Conference on*, pages 1-5, 2010.
[37] P. Volgyesi, A. Nada, X. Koutsoukos, and A. Ledeczi. Air quality monitoring with sensormap. In *Information processing in Sensor Networks, 2008. IPSN '08. International Conference on,* pages 529-530, 2008.