Chapter 1
The ARUM Experimentation Platform : an "Open" Tool to evaluate Mobile Systems Applications

Marc-Olivier Killijian, Matthieu Roy, and Gaetan Severac*

Abstract This paper present the ARUM robotic platform. Inspired by the needs of realism in mobile networks simulation, this platform is composed of small mobiles robots using real, but attenuated, Wi-Fi communication interfaces. To reproduce at a laboratory scale mobile systems, robots are moving in an 100 square meters area, tracked by a precise positioning system. In this document we present the rational of such simulation solution, provide its complete description, and show how it can be used for evaluation by briefly explaining how to implement specific algorithms on the computers embedded by the robots.

1.1 Objectives

In this paper, we present the ARUM robotic platform[2] targeted at evaluating performance, resilience and robustness of mobile systems. To obtain an efficient evaluation platform, three specific criteria were considered: **Control conditions** (real time monitoring, repeatability, flexibility, scalability), **Effective implementation** (easiness of configuration, devices autonomy, portability, low cost, miniaturization), and **Realistic environment** (network scale, traffic load, node mobility, positioning, radio broadcast behavior). To our knowledge this platform is the only one to date to integrate all these features in a single environment.

Indeed, current evaluation strategies for distributed and mobile systems can be split in five categories:

Marc-Olivier Killijian, Matthieu Roy
LAAS-CNRS (TSF), 7 avenue du colonel Roche FR-31077 Toulouse, France, e-mail: roy@laas.fr

Gaetan Severac
ONERA (DCSD), 2 avenue E. Belin FR-31055 Toulouse, France, e-mail: gaetan.severac@onera.fr

* First and Corresponding author

[2] ARUM stands for "an Approach for the Resilience of Ubiquitous Mobile Systems"

- **Simulators**. Simulators are cheap and fast to set up, with almost no limitation in the number of nodes. Due to their scalability and simplicity, they are well suited for initial testing. Furthermore, they may speed up development of theoretical researches because since they allow a perfect monitoring and repeatability [21, 5]. Nevertheless simulation is based on models of the running environment, and thus cannot reflect the real complexity of natural environments, particularly for radio communication and mobility pattern[8][6][3] .
- **Emulators**. Emulators are built to physically reproduce connections events using real wired network hardware[16][19]. They provide features interesting for protocol implementation but they still use simulation to reproduce wireless communication behavior and mobility[4].
- **Testbeds**. The ARUM platform we present in this paper can be classified in this category. Testbeds are closer to reality thanks to the use of real hardware. They exist since years now, from the historical MIT RoofNet[1], to the more recent MoteLab[3] service. Ideal to finalize and validate applications before real-life experiementations, testbeds provide much more realistic result than emulators or simulators. But they are also expensive, time consuming and limited by the physical resources/hardware used[14]. Because of those limitations, only a few of them implement real mobility, to the best of our knwoledge two platforms using mobile robots have been develloped recently : MINT[17] and Mobile Emulab[10]. Original solutions that emulate mobility can also be found in the literature, like MOBNET [7] which varies the transmission power levels of fixed access points. It is interesting to notice that most of those platforms have to deal with large variations in the communication noise level because of environment perturbations. Such difficulties can be problematic during applications development phases, but they are representative of conditions encountered in the real life.
- **Hybrid simulators**. They used both simulated networks and real devices, taking advantages and disadvantages of each[18] [23]. They are particularly suited to study, at a low cost, the interconnection of some real devices to a huge network, the latter being simulated.
- **Real live experiments**. This is, obviously, the more realistic kind of experimentation, but they present inherent technical problems which can bring more technical difficulties than scientific benefits[11] [15]. They are absolutely necessary for commercial applications, because it is impossible to truly simulate real environment yet. Yet, they are very expensive, error-prone, and they do not provide repeatability of experiments, due to the wide variability of real environments. As such, such platforms are not used in the context of research and education.

Among all these technologies, there is no good or wrong solutions, the best choice depends on a specific needs and available resources, as shown in Figure 1.1. A survey on the subject will be published soon on the web page of the project[4] to help scientists to choose theire most adapted solution.

---

[3] Harvard Sensor Network Testbed - http://motelab.eecs.harvard.edu

[4] ARUM platform - http://projects.laas.fr/ARUM/

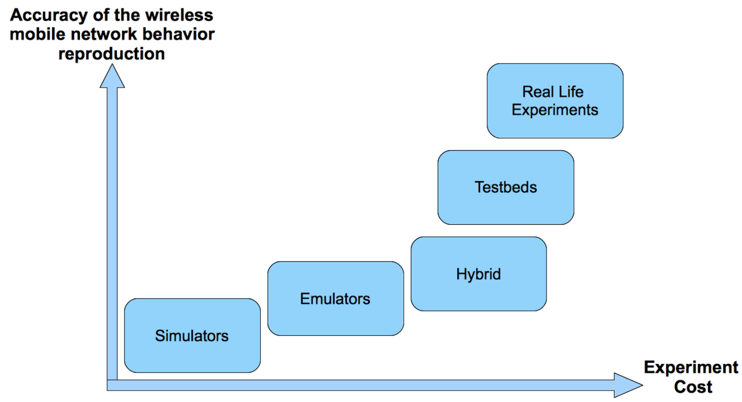

Fig. 1.1 Accuracy of the evaluation solutions for mobile networks depending on their respective costs

In our case, both for scientific and for demonstration reasons, we decided to implement a testbed, the ARUM platform. Indeed our primary goal was to complement simulation and allow realistic evaluation of mobile systems, at a laboratory scale. It finally appeared to be a good platform for demonstration and education, since the platform can be used pedagogically to present the various aspects and problems raised by mobile systems.

1.2 Design

To complete the goals presented in previous section, we had to design an experimental evaluation platform composed of mobile devices. We dispose of a room of approximately $100m^2$ to emulate systems of different sizes, hence we decided to scale every parameter of the system to fit within our physical constrains. Technically speaking, each mobile device is built with : a programmable *mobile hardware* able to carry the device itself, a *lightweight processing unit* equipped with one or several *wireless network interfaces* and a *positioning device*. Hardware modeling required a reduction or increase of scale to be able to conduct experiments within the laboratory. To obtain a realistic environment, all services have been modified according to the same scale factor. For example, if we consider a vehicular ad-hoc network experiment [13], a typical GPS embedded in a moving car is accurate to within 5-20m. So, for our $100m^2$ indoor environment to be a scaled down representation of a $250000m^2$ outdoor environment (a scale reduction factor of 50), the indoor positioning accuracy needs to be $10-40$cm. Table 1.1 summarizes the required change in scale for all peripherals of a node.

Device	Real Accuracy	Scaled Accuracy
Wireless	range: 100m	range: 2m
GPS	5m	10cm
Node size	a few meters	a few decimeters
Node speed	a few m/s	less than 1m/s

Table 1.1 Scale needs

We understand here that to meet those requirements some parts of the development were much more important. The focus was put on the reduced WiFi interfaces, the precise positioning and the node mobility.The different parts of the platform will be detailed in the following section. It is important to remind that we do not perform research on robotic mobility, we simply want an efficient evaluation tool, using existing technological solutions.

1.3 Technical solutions

1.3.1 Mobility

To reproduce mobile systems conditions, the devices used in the platform must be mobile. But when conducting experiments, a human operator cannot be behind each device, so mobility has to be automated. This is why we considered the use of simple small robot platforms in order to carry around the platform devices. The task of these robots is to implement the mobility of the nodes following a movement scenario.

Fig. 1.2 A picture of the the ARUM Platform

A node is implemented in the system using a laptop computer that is carried by a simple robotic platform, that includes all hardware devices, the software under testing and the software in charge of controlling robots movements. Notice that software under testing and control software are totally independent, there are running on the same computer for practical reasons only.

For the mobile platform we use Lynxmotion 4WD rover. We selected it instead of other smaller robot (e.g. Lego Mindstorm) because this rover is able to carry a payload of 2 Kg during a few hours, running at a maximum speed of 1m/s. It is also relatively cheap (cf. table 1.2 and easy to build. We equipped it with infrared proximity sensors to avoid collision and a top deck to support the laptop, a positioning system and a modified Wi-Fi interface.

The motion control software, running on the carried laptop, communicates speeds orders (linear speed and angular speed) to the robot. The mobility patterns are drawn by an operator for each mobile robot, using a dedicated software, that sends it to the mobile nodes control software. This enables flexibility – each node has its own mobility pattern – and repeatability – a pattern can be saved and replayed.

1.3.2 Localization

Positioning is a critical point of the platform. Firstly, we need to reproduce the kind of information produced by actual market solutions such as GPS, pondered by our scale factor. Secondly, we need a precise and real-time position of the mobile node to allow an accurate motion control of the robot. Our specifications required a precision within the centimeter and a minimum refresh of 2 Hz. Several technologies are currently available for indoor location [9]. During the building of the platform, we tried four different solutions.

The first system we tested was the **Cricket solution** [20], developed by MIT. Cricket is based on simultaneous ultrasound/RF messages and triangulation. In theory, this system is very efficient, but in practice we were confronted to important limitations due to ultrasound disturbances. Finally we had to abandon this technology.

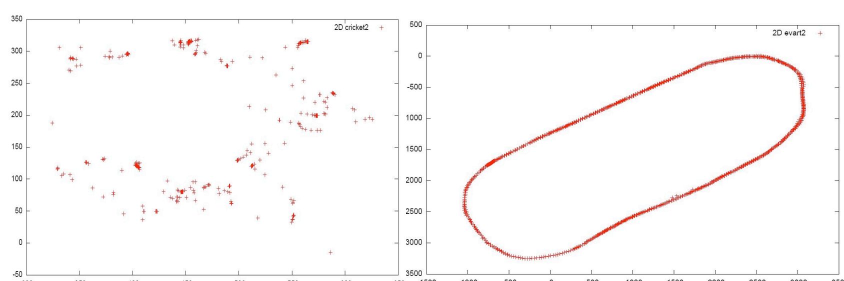

Fig. 1.3 Comparative results of Ultrasounds (left) and Infrared (right) positioning systems. A robot is tracked by the two different systems while following the same circuit drew on the floor

To reach our desired level of accuracy for indoor positioning, we then used a dedicated motion capture technology that tracks objects based on real-time analysis of images captured by fixed infra-red cameras. **The Cortex system**[5] is able to localize objects at the millimeter scale. Although the precision attained was more than enough for our needs, the system has some drawbacks: the whole system is very expensive (in the order of 100k), calibration is a tedious task, and infra-red signals cannot cross obstacles such as humans. Figure 1.3 shows compared results of the ultrasounds and infra-red systems.

The localization system currently used is the **Hagisonic StarGazer technology**[6]. It is also based on infra-red camera but they are small and embedded on-board on the mobile robots. They locate themselves by tracking statically placed infra-red-visible tags. With Hagisonic, a camera needs to see only one single tag to be able to calculate its position and the precision is about a few millimeters, with a frequency of 10 Hz. So this technology, more affordable, was plenty satisfying our requirements.

An **Ultra-Wide-Band-based localization system** (UWB), by Ubisense[7], has also been deployed and used for the experiments. Localization is performed by 4 sensors, placed in the room at each corner, that listen for signals sent by small tags that emit impulses in a wide spectrum. Such impulses can traverse human bodies and small obstacles, so the whole system is robust to external perturbation, but, from our preliminary measurements, attainable precision is about 10cm. The next step will be to couple this technology with the Hagisonic camera system, resulting in a localization system with better properties: it will be relatively cheap, robust to external perturbations such as obstacles, and will have most of the time a precision about the order of a centimeter.

To keep our experimental platform positioning system generic, despite the numerous different technologies used, we developed a **position server**, accessible via the supervision wireless network of the experimentation room. Two kinds of clients can communicate with it, using standard XML messages. A client can be a *position provider* (Cortex, Hagisonic, UWB, ...) and send to the server the position of one or several mobiles or the client can be a *position consumer* (supervision application, motion control software, ...) and ask to the server the position of one or several mobiles. Using this strategy, it is possible to change the technology of one system, provider or consumer of position, and the modification will remain transparent to all the other devices.

1.3.3 Scenario Drawing Interface

To have adequate experimental conditions, the mobile nodes of the platform need to follow and repeat defined mobility scenario. But first, an operator has to define the mobility scenario. We developed a graphical user interface to draw, configure, visualize and manage mobility patterns. Now the interface is a complete program

[5] Cortex Motion Capture - http://www.motionanalysis.com
[6] Hagisonic - http://www.hagisonic.com/
[7] Ubisense - http://www.ubisense.net/en/products/precise-real-time-location.html

composed of 7 different tabs. It allows to set up passage points on a map, to edit the rovers routes, to simulate the rovers movements, to send the mobility patterns to the different rovers and to control the execution of them. It can be run on any computer connected to the supervision network.

1.3.4 Motion control and trajectory computing

A robot control program receives a scenario description or the movements orders from the scenario drawing interface. This program is running on the embedded computer of the mobile node. The mobility scenarii are then converted into commands and sent to the robot motion control environment. This environment is composed of GenoM[8] modules in charge of computing the final trajectory and controlling the robot speed to follow it. Proximity infra-red captors are continuously polled to stop the robot if an obstacle is detected. We chose the GenoM environment, developed at the LAAS-CNRS laboratory, because it is an open source solution, already functional and still maintained by the robotic community.

1.3.5 Reduced wireless communication

The communication range of the participants (mobile nodes and infrastructure access-points) has to be scaled according to the experiment being conducted. For our first experimentation, the scale factor had to be 50 (cf. table 1.1) but, ideally, the communication range should be variable. Some WiFi network interface drivers propose an API for reducing their transmission power. But the implementation of this feature is often rather limited, or ineffective. A satisfying solution consists in using signal attenuators[9] placed between the WiFi network interfaces and their antennas. The necessary capacity of the attenuators depends on many parameters such as the power of the WiFi interfaces and the efficiency of the antennas, but also on the speed of the robot movements, the room environment, etc. As it is impossible to predict or calculate the WiFi radio wave propagation we conducted empirical experimentation[12] to establish the relationship between signal attenuation and communication range.

[8] GenoM - https://softs.laas.fr/openrobots/wiki/genom

[9] An attenuator is an electronic device that reduces the amplitude or power of a signal without appreciably distorting its waveform.

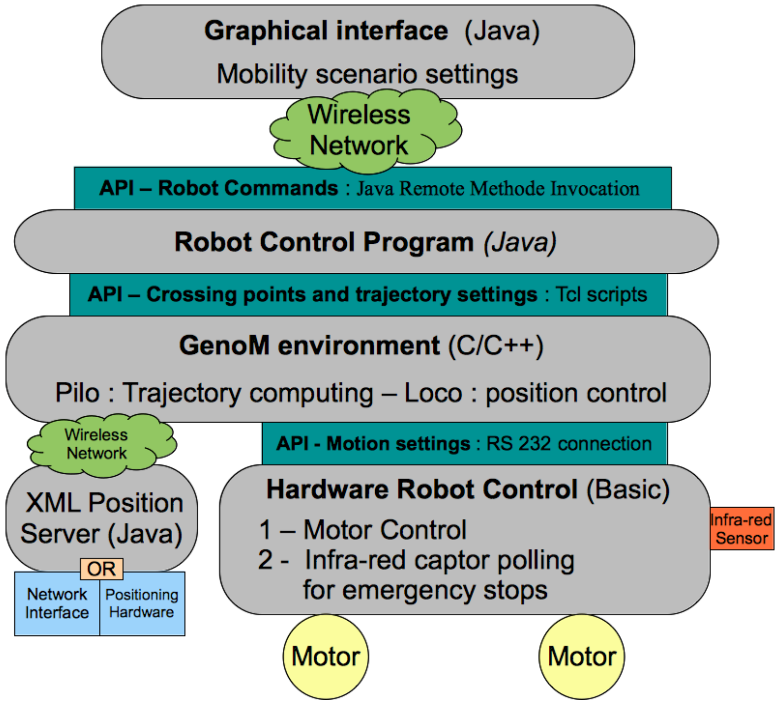

Fig. 1.4 Mobile Node architecture Overview

1.3.6 Supervision network

For the communications between the collaborative algorithms tested on the platform, the attenuated WiFi interfaces previously presented are used. So the internal wireless card of the embedded computers is available for monitoring. Connected to a LAN access point, it provides direct access to each computer without disrupting the current experiment. This system is used to send monitoring information to the robot (e.g. position, commands, ...) and to retrieve data from the mobile nodes in real time, allowing a dynamically overview and analysis of the tested algorithms.

1.3.7 Implementation and Price

It is interesting to consider that all the different parts of a mobiles nodes (localization, trajectory planning, robot control, communication, ...) are connected thought clearly defined and documented interface, so it easy and fast to change one of this part to make the platform evaluate, without re-designing everything (cf. figure 1.4). For example we envisage buying a new localization system and changing the Lynxmotion robot for a Roomba[10] development mobile platform. Anyone interested in reproducing our evaluation platform in a laboratory can reuse some parts of interest modify others. The full documentation and sources of the platform is available at this web address: http://projects.laas.fr/ARUM/. As an indication you can see in the table 1.2 the actual price of the different parts of a mobile node.

Device	Price ($)
Linxmotion mobile Platform Kit	1 000
Hagisonic IR Camera	1000
Wireless WiFi interface	50
Attenuators	70
Laptop	1200
Serial-USB Adaptors	30
Total	3350

Table 1.2 Platform devices Costs

1.4 Experimentation and Lessons Learnt

To evaluate our ARUM platform, we experiment the **Distributed Black-Box application**, or DBB for short. This work was conducted in the course of the european project HIDENETS[11]. The application developed provides a virtual device, whose semantics is similar to avionics black-boxes, that tracks cars history in a way that can be replayed in the event of a car accident. It ensures information is securely stored using replication mechanisms, by means of exchanging positions between cars. This architecture is a partial implementation of the HIDENETS architecture and has been detailed in the project deliverable [2].

The ARUM platform was used to emulate the network of communicating cars. Through this work, the global performance of the evaluation platform was validated. The modularity and repeatability of the mobility patterns was used to test and improve the DBB algorithms in controlled situations. The use of real, power reduced WiFi interfaces allowed realistic results; we monitored during the experiment wireless signal variations similar to real wireless network behavior in difficult conditions (maximum range limit, noise perturbation, obstacle, ...).

A very precise positioning system was used both by the tested cooperative algorithms and the robot motion control software, without disturbances. With hindsight we have to admit that, even if we get positive results, we had to deal with a lot of contingencies. The total labour cost of the platform development was more consequent than expected. Some parts of the development would have been impossible to

[10] Roomba Devel - http://www.irobot.com/images/consumer/hacker/roombascispecmanual.pdf

[11] HIghly DEpendable ip-based NETworks and Services - http://www.hidenets.aau.dk/

reduce - attenuated Wi-Fi scaling, positioning systems tests - but if we had to rebuild all from scratch we would probably choose a mobile robotic platform that already has motion control implemented - such as the Roomba Devel platform.

However, now that the platform is finished and validated, it can be used as a tool ”out of the box”; you can contact us and come to our laboratory to implement your algorithms on the mobile nodes. As showed in Section 1.3, it is possible to interface any code with the different parts of the mobile node - communication, positioning, monitoring, ... - and to easily program a mobility pattern. All the parts of the platform are segmented by software interfaces, defined in the documentation[12], so it can quickly be handled and adapted by anybody interested. And if you already have your own hardware, of if you want to buy it, it is possible download the sources of each software part to rebuild the same platform at your laboratory.

Even if it is not the primary function of this platform, we noticed that the versatility and the easiness of use of this platform makes it an interesting educational tool. All the different parts of it, presented in section 1.3, can be used, studied and replaced by students. The localization, the mobility scenario computing, the motion control and trajectory calculation or the reduces wireless communication, could support interesting university work.

1.5 Conclusion

This article started by pointing out the difficulties of evaluation of application for mobile devices systems. It presented the difficulties encountered to emulate a realistic mobile network environment at a laboratory scale. Those observations motivated the development of a testbed platform designed to evaluate distributed applications. This platform, ARUM, appears to provide an interesting compromise between resources consumption (in terms of manpower) and accuracy of results, appropriate to complement simulation. The whole architecture is described part by part to ease reuse by researchers or in an educational context, while reducing the waste of time and money in development and tests.

References

1. D Aguayo, J Bicket, S Biswas, and D De Couto. Mit roofnet: Construction of a production quality ad-hoc network. *MobiCom Poster*, 2003.
2. J. Arlat and M. Kaâniche(editors). Hidenets. revised reference model. *LAAS-CNRS, Contract Report nr. 07456*, Septembre 2007.
3. Elyes Ben Hamida, Guillaume Chelius, and Jean-Marie Gorce. Impact of the Physical Layer Modeling on the Accuracy and Scalability of Wireless Network Simulation. *Simulation*, 85:574–588, 09 2009.

[12] ARUM platform - http://projects.laas.fr/ARUM/

4. Razvan Beuran, Lan Tien Nguyen, Toshiyuki Miyachi, and Junya Nakata. Qomb: A wireless network emulation testbed. *reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings*, 2009.
5. Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING*, 2:483–502, 2002.
6. D Cavin, Y Sasson, and A Schiper. On the accuracy of manet simulators. *POMC '02: Proceedings of the second ACM interna- tional workshop on Principles of mobile comput- ing, pages 38–43, New York, NY, USA*, 2002.
7. Kun chan Lan, Eranga Perera, and Henrik Petander. Mobnet: the design and implementation of a network mobility testbed for nemo protocol. 2005.
8. Kwan-Wu Chin, John Judge, Aidan Williams, and Roger Kermode. Implementation experience with manet routing protocols. *ACM SIGCOMM Computer*, 2002.
9. J. Hightower and G. Borriello. A survey and taxonomy of location systems for ubiquitous computing. *IEEE Computer*, 2001.
10. David Johnson, Tim Stack, Russ Fish, Daniel Montrallo Flickinger, Leigh Stoller, Robert Ricci, and Jay Lepreau. Mobile emulab: a robotic wireless sensor network testbed. *IEEE INFOCOM*, 2006.
11. Wolfgang Kiess and Martin Mauve. A survey on real-world implementations of mobile ad-hoc networks. *Ad Hoc Networks*, (5):324 – 339, 2005.
12. M.O. Killijian, D. Powell, M. Roy, and G. Séverac. Experimental evaluation of ubiquitous systems: Why and how to reduce wifi communication range. *DEBS'08*, 2008.
13. M.O. Killijian, M. Roy, G. Séverac, and C. Zanon. Data backup for mobile nodes : a cooperative middleware and experimentation platform. *DSN'09 WADS*, 2009.
14. Matthias Kropff, Tronje Krop, Matthias Hollick, Parag S. Mogre, and Ralf Steinmetz. A survey on real world and emulation testbeds for mobile ad hoc networks. *Proceedings of 2nd IEEE International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM 2006)*, 2006.
15. K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. *Parallel and Distributed Processing Symposium, International*, 0:155, 2006.
16. Michael Matthes, Holger Biehl, Michael Lauer, and Oswald Drobnik. Massive: An emulation environment for mobile ad-hoc networks. *Proceedings of the Second Annual Conference on Wireless On-demand Network Systems and Services (WONS'05)*, 2005.
17. Christopher Mitchell, Vikram P. Munishwar, Shailendra Singh, and Xiaoshuang Wang. Testbed design and localization in mint-2: A miniaturized robotic platform for wireless protocol development and emulation. *Proc. First International Conference on Communication systems and Networks (COMSNETS)*, 2009.
18. Fredrik Osterlind, Adam Dunkels, and Thiemo Voigt. Sensornet checkpointing: Enabling repeatability in testbeds and realism in simulations. 2008.
19. Matija Pužar and Thomas Plagemann. Neman: A network emulator for mobile ad-hoc networks. *Technical Report 321, ISBN 82-7368-274-9 Department of Informatics, University of Oslo*, 2005.
20. A. Smith, H. Balakrishnan, M. Goraczko, and N. B. Priyantha. Tracking moving devices with the cricket location system. *2nd International Conference on Mobile Systems, Applications and Services (Mobisys 2004), Boston, MA*, June 2004.
21. Ivan Stojmenovic. Simulations in wireless sensor and ad hoc networks: matching and advancing models, metrics, and solutions. *IEEE Communications Magazine*, 46(12):102–107, 2008.
22. Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. Motelab: A wireless sensor network testbed (http://motelab.eecs.harvard.edu). *ISPN/SPOTS*, 2005.
23. J Zhou, Z Ji, and R Bagrodia. Twine: A hybrid emulation testbed for wireless networks and applications. *IEEE INFOCOM, 2006*, 2006.