MM-ulator: Towards a Common Evaluation Platform for Mixed Mode Environments

Matthias Kropff, Christian Reinl, Kim Listmann, Karen Petersen, Katayon Radkhah, Faisal Karim Shaikh, Arthur Herzog, Armin Strobel, Daniel Jacobi, and Oskar von Stryk*

Technische Universität Darmstadt, Research Training Group "Cooperative, Adaptive and Responsive Monitoring in Mixed Mode Environments", 64289 Darmstadt, Germany; WWW home page: `http://www.gkmm.tu-darmstadt.de`

Abstract. We investigate the interaction of mobile robots, relying on information provided by heterogeneous sensor nodes, to accomplish a mission. Cooperative, adaptive and responsive monitoring in Mixed-Mode Environments (MMEs) raises the need for multi-disciplinary research initiatives. To date, such research initiatives are limited since each discipline focusses on its domain specific simulation or testbed environment. Existing evaluation environments do not respect the interdependencies occurring in MMEs. As a consequence, holistic validation for development, debugging, and performance analysis requires an evaluation tool incorporating multi-disciplinary demands. In the context of MMEs, we discuss existing solutions and highlight the synergetic benefits of a common evaluation tool. Based on this analysis we present the concept of the *MM-ulator*: a novel architecture for an evaluation tool incorporating the necessary diversity for multi-agent hard-/software-in-the-loop simulation in a modular and scalable way.

1 Introduction

Mixed Mode Environments cover the range from static and structured to highly dynamic and unstructured environments and consist of a myriad of networked nodes including sensors, robots and possibly humans-in-the-loop. Further, MMEs are characterized by different kinds of heterogeneity with respect to the utilized devices and their capabilities (e.g. communication interfaces, energy resources, sensor data). The scenarios addressed within MMEs may vary from monitoring and surveillance tasks, using heterogeneous sensors, to the coordination of autonomous vehicles. Accomplishing these tasks requires knowledge from four main domains: (1) robotics and control, (2) communication, (3) sensing, and (4) dependable middleware.

In order to respect the multi-disciplinary issues, a common tool is needed to examine the various problems and mutual dependencies. Throughout the last

* This research has been supported by the German Research Foundation (DFG) within the Research Training Group 1362 "Cooperative, adaptive and responsive monitoring in mixed mode environments"

years, the design of such simulation environments has been of significant interest, particularly to the RoboCup community [3]. To the best of our knowledge, however, there exists no evaluation tool covering the diversity of the above named fields. Thus, a concept introducing a holistic validation tool respecting the interdisciplinarity and heterogeneity in MMEs is developed. In the remaining of the paper, we will refer to this concept as the *MM-ulator*.

The paper is organized as follows: Next, we highlight the benefits of a common evaluation tool and define the necessary requirements. In Section 3 we survey relevant simulation tools and discuss their applicability to relevant scenarios. The proposed architecture of the *MM-ulator* is presented in Section 4.

2 Benefits and Challenges of a Common Evaluation Platform

For the purpose of validation and performance analysis, three well known evaluation methodologies can be applied: (1) analytical modeling, (2) simulation, and (3) real experiments. Since analytical modeling is rather impractical and real experiments are expensive and time consuming, a valuable approach is to use simulation. But as only real experiments provide realistic results, they cannot be neglected in general. Hence, validation techniques giving the opportunity to incorporate real systems, would be beneficial. To this end, we focus on *emulation*, a hybrid validation technique combining simulation and real-world experiments, including the known elements of software- and hardware-in-the-loop tests. Figure 1 highlights the conceptual differences to pure simulation.

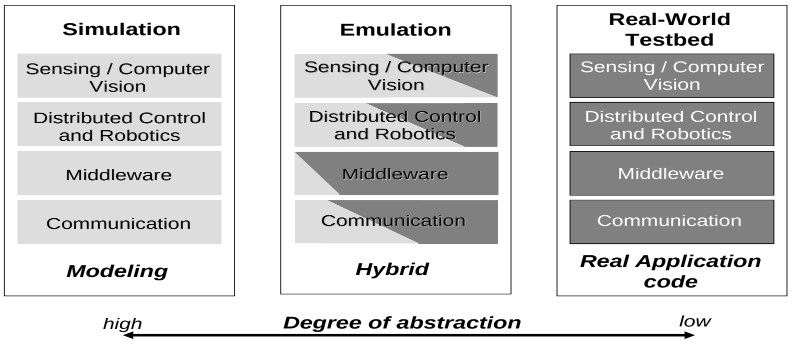
Simulation
Sensing / Computer Vision
Distributed Control and Robotics
Middleware
Communication
Modeling
Emulation
Sensing / Computer Vision
Distributed Control and Robotics
Middleware
Communication
Hybrid
Real-World Testbed
Sensing / Computer Vision
Distributed Control and Robotics
Middleware
Communication
Real Application code
high
Degree of abstraction
low

Fig. 1. Emulation as hybrid approach of simulation and real-world experiments.

Relying on the emulation approach, the developer does not have to cope with simulation time semantics, and the integration of existing sensor and robot hardware to a certain degree is facilitated. This turns emulation into a suitable tool for controlled prototype testing and debugging. Figure 1 also indicates that the degree of abstraction depends merely on the modeled building block: Since only minor parts need to be modeled in detail, the degree of abstraction for the middleware and communication module is low, whereas the sensing and control module require a moderate degree of abstraction.

2.1 Benefits of Validation by Using Multi-Disciplinary Knowledge

The synergetic benefits of tightly coupled multi-disciplinary knowledge is shown in Figure 2. The interconnecting arrows indicate the potential decrease in the level of abstraction regarding the shown dependencies, enabling more realistic results. To have a more thorough understanding of the highlighted challenges,

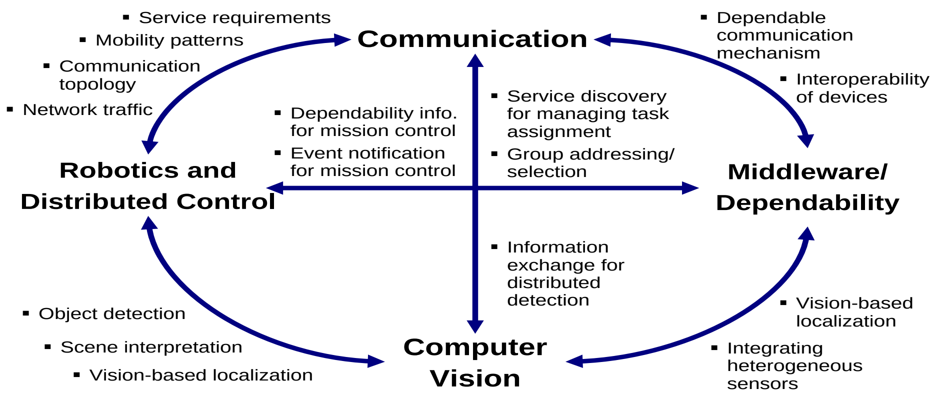
Service requirements
Mobility patterns
Communication topology
Network traffic
Communication
Dependable communication mechanism
Interoperability of devices
Dependability info. for mission control
Event notification for mission control
Service discovery for managing task assignment
Group addressing/ selection
Robotics and Distributed Control
Middleware/ Dependability
Information exchange for distributed detection
Object detection
Scene interpretation
Vision-based localization
Vision-based localization
Integrating heterogeneous sensors
Computer Vision

Fig. 2. Multi-disciplinary knowledge.

let us consider an explosion in a chemical plant and a subsequent spread of fire, evolving into a toxic environment, inaccessible to human operators. In order to support the rescue operations, a team of robots starts exploring the environment. Fundamental tasks are building a map of the environment, locating victims and marking safe exit pathways or unreachable areas.

In the following, we will point out some sparsely tackled research questions from the perspective of cooperative control and mobile communicating, as well as sensing and middleware.

Benefits for mobile communicating teams of vehicles and nodes: In order to use heterogeneous autonomous mobile sensing platforms such as robots within MMEs, it is crucial to combine their control and coordination oriented communication. It has been shown that the information flow among the robots influences the stability of their coordinated movement [10]. Due to this mutual coupling, the communication properties of the environment and the robots need to be respected when applicable control algorithms are being designed. These properties include reflections, fading effects, communication range and packet losses. On the one hand, these effects have a significant impact, e.g. on close loop stability for cooperative control. On the other hand, distributed control may change the network topology, improving routing efficiency or covering a wider area while remaining connected. This combination is obviously bidirectional and very important with respect to cooperative control of robotic groups.

Typically, field data is provided by sensors. Cooperative data gathering based on aggregated information is closely related to the positions of the robots and viewing angles of their sensors. Thus, the verification of hypotheses in scene interpretation and object detection can be significantly improved by connecting the algorithms to the motion control of robots. Realistic simulated sensor outputs, e.g. including noise, will show the reliability of control algorithms in non-ideal situations and will give rise to increase the robustness of the applied methods. Furthermore, visual servoing [8], dynamic acquisition of navigational data, and distributed cooperative mapping strategies are other representative topics. They incorporate fundamental issues from sensing and motion control, such as the amount of necessary information exchange between multiple unmanned vehicles, mission control, or stability of coordination of partially autonomous robots.

Benefits for sensing and middleware: In the exemplified scenario, robots will have to discover services offered by sensor nodes in radio proximity and therefore, help to re-establish a reliable and efficient communication infrastructure. This smart behavior still imposes several research challenges on communication and middleware concepts. Self-description and self-profiling mechanisms are needed to spontaneously migrate devices into the networked environment, regardless of the given sensor manufacturer or interface. Middleware simplifies the interconnections between sensing, communication, and distributed control. A formal specification of interfaces for these parts leads to an increase of the interoperability of different devices. One challenge is to specify a common representation, to allow hardware independent robot task assignment, actuator control, interpretation of pre-processed sensor data, and robot capability description.

Dependability supporting approaches like multi-path routing require the specification of constraints, which can be provided by the middleware if appropriate interfaces are defined. Finally, several questions in the communication domain are closely linked to information provided by a well-defined middleware concept. For instance, approaches like efficient semantic addressing and routing of sensing and actuation data require certain self-description functionalities on the communication level.

Even by this brief discussion on upcoming research challenges, a fundamental question arises: How will multi-disciplinary performance metrics look like? We believe that having a holistic evaluation tool, available solutions for MME problems can be regarded from new perspectives. For instance, as migrating wireless network constraints into robot control, new metrics like coordination stability will emerge.

To our knowledge, these cross-sectional issues are not supported by any of the existing simulation environments. Based on this analysis we propose the requirements, which are fundamental for such a holistic evaluation tool.

2.2 Basic Requirements

The simulation and emulation of a real physical world requires a flexible approach. A modular architecture is necessary to facilitate scalability and adjustable degrees of abstraction. Furthermore, the *MM-ulator* needs to provide realistic fault and security models as well as efficient analysis and visualization of gathered data. Dependability aspects provide different faults and threat models which can also be considered in the *MM-ulator*.

Modeling node properties: Robots, unmanned vehicles, sensors, actuators, and main servers require heterogeneous 3D models. Besides, the locomotion properties, kinematics and motion dynamics of robots and vehicles are essential to be modeled. Sensor readings, e.g., for laser scanners, cameras, or contact sensors must be considered with an adjustable accuracy. Specific resources like processing power (e.g., for on-board image processing), memory, communication capabilities, energy consumption, and sensing devices with different levels of accuracy have to be modeled properly and comprehensively.

Modeling physical environment properties: The physical environment splits up in static and dynamic properties. The static part consists of a realistic 3D model of the environment, including obstacles, buildings, surface properties, and various objects of interest as well as physical effects like gravity. The dynamic parts of the physical environment include basic radio frequency propagation models for identifying communication links and specific scenario settings like mobility patterns of victims and rescue teams, chemical and physical concentrations (e.g., radioactivity), diffusion process of (toxic) gas, or the spread of fire. The dynamic parts need to be modeled thoroughly. Also interactions with the environment by the nodes, e.g., the distribution of RFID tags, robot driven installment of sensor nodes need to be incorporated in the model of the physical environment.

3 Related Work

Currently available simulation environments for testing algorithmic approaches for the addressed scenarios are either rooted in the area of 3D robot simulation, Wireless Sensor Networks (WSNs) or in Mobile Ad Hoc Networks (MANETs).

USARSim [6] is a 3D simulator for testing robotic applications, especially for search and rescue scenarios. It is based on the *Unreal Engine* by epic games [1], providing plausible physics simulation and high quality visualization. State information is exchanged with the engine using the scripting language *Unrealscript*. USARSim supports a variety of robot models, including legged, wheeled and tracked vehicles, as well as submarines and helicopters, and additionally provides a wide range of sensor models, including cameras, range, touch or odometry sensors. Based on existing classes and adapted scripts new robots/sensors can be added, respectively. Robot control can either be performed by sending text messages via TCP sockets, or by utilizing wrappers for the middleware Player [7], Pyro [5] or MOAST [4], that are already available in USARSim. In most cases, code that was developed within the simulation will also work on the real robots.

The Multi-Robot-Simulation-Framework (MuRoSimF) [12] (cf. Fig. 5(left)) can be used to create simulations for cooperating teams of heterogeneous robots in dynamic environments. MuRoSimF provides models for different legged and wheeled robots equipped with sensors, like cameras and laser range finders. Its modular structure facilitates to assign different algorithms to each part in the simulation (e.g. motion or sensor simulation for individual robots) and provides the option to be extended by the required inter robot communication mechanism.

Other related robot simulation environments are Webots [24], Gazebo [17], Microsoft Robotics Studio [2] and SimRobot [18]. Common to the named tools is their focus on detailed 3D models of the environment, surfaces, robots and physics simulation, while they predominantly lack of components for modeling wireless multi-hop communication, integration of mediating middleware concepts or the incorporation of dependability models for realistic scenario test-runs.

A second category of simulation environments evolves from the area of Wireless Sensor Networks (cf. Fig. 5(right)). TOSSIM [19] is a simulator for wireless sensor nodes which are running the operating system TinyOS. Its dual mode functionality allows to run TinyOS code in a controlled simulation mode as well as on real sensor hardware. In simulation mode TOSSIM models link connectivity by probabilistic models and provides detailed hardware abstraction effects including ADC and battery models. A similar approach is the cycle-accurate instruction level simulator Avrora [26], which operates on sensor node firmware images and provides simulation of fine grained radio models including detailed models to evaluate the energy efficiency of different protocols. A two tier form of WSN heterogeneity is supported by the EMStar framework [13]. It provides simulation and emulation capabilities for constrained motes, as well as more powerful microservers, and therefore focus on middleware mechanisms to provide interoperability.

The most significant drawback of the presented platforms is that they were intentionally designed for static, resource constrained nodes. This disallows the simultaneous integration of more powerful platforms within this setup.

Mobile nodes possessing higher processing/communication capabilities are addressed in the area of Mobile Ad Hoc Networks. Typical emulation environments strongly focus on the evaluation of routing protocols for Mobile Ad Hoc Networks and are shown in [9, 11, 14, 20–23, 25, 27]. However, these approaches address predominantly algorithmic solutions on the network and medium access layer, while mobility and network traffic patterns are predefined in advance of a testrun. As a result, the evaluation of mechanisms for dynamic and cooperative task assignment, motion control under constraints of network connectivity or the interaction of heterogeneous groups of mobile robots are disregarded.

4 Proposed Architecture

The proposed architecture for the *MM-ulator* aims to fulfill two main requirements: (1) reducing the software re-implementation overhead when switching from validation by simulation to a real-world test-run and (2) incorporating real hardware platforms in the evaluation process. To cover a wide range of possible devices, a generic node architecture is proposed that allows to run the same software code either on real embedded systems like robots or sensor hardware, or to instantiate a pure software entity as a virtual node on a common PC platform to increase the scalability of a test-run.

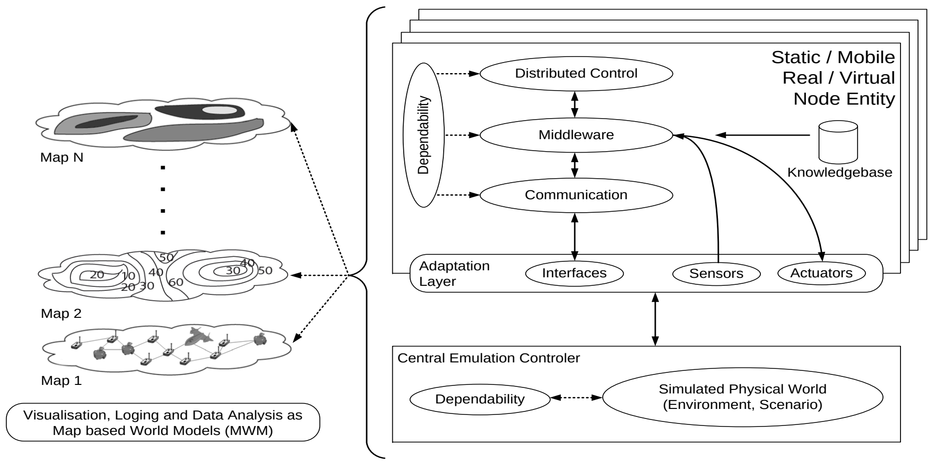

Fig. 3. Architecture of the proposed *MM-ulator* .

4.1 Inner Node Architecture

The inner node architecture describes the functionalities of the node modules and their interconnecting interfaces. The modularity of the architecture allows to model a variety of heterogeneous devices. For instance, while the algorithms encapsulated in the distributed control module model the task planning component on a mobile robot, they might be absent in case the instantiated node entity represents a static, resource limited node, which only supports basic sensing capabilities. The *Knowledge Database* provides information about the node's communication, processing and memory capabilities. It also comprises the node's sensing and actuating resources and provides information about the node's type of locomotion, allowing to easily configure an autonomous vehicle or a static sensor node. Moreover, the knowledge database provides details about a node's energy source and depletion process during operation.

The *Middleware* module provides standardized interfaces to bridge the intra node communication between the sensors, actuators, distributed control- and communication module. It encapsulates algorithms and protocols to provide semantic node addressing and basic Publish/Subscribe mechanisms, facilitating efficient group communication among diverse node groups. Furthermore, the middleware architecture comprises mechanisms for idle sleep cycles to model energy saving algorithms for wireless sensors. Based on information from the knowledge database, the middleware module can generate a generic node description, which can be distributed to neighboring nodes to provide and discover remote sensing capabilities and to coordinate actuation capabilities for distributed task planning. Additionally, the middleware module encapsulates mechanisms for controlling data privacy and security issues.

The *Distributed Control* module comprises the algorithms for distributed task planning, coordinated task assignment and mission control. It holds the control logic for robot movements and deduces possible task goals, depending on the predefined mission statement or the scene interpretation based on sensing information. Predefined mission tasks range from fetching simple sensor readings at a specific location to more elaborated tasks such as exploring the environment and finding injured people.

The *Communication* module encompasses higher level algorithms and protocols for wireless ad hoc communication. To provide an heterogeneous emulation scenario of virtual and hardware nodes simultaneously, network layer functionalities like routing algorithms, service discovery and interface management mechanisms are modeled consistently on node level. For modeling further wireless network mechanisms like the Medium Access (MAC) layer or topology control algorithms, the communication interface at the adaptation layer provides means to specify packet based scheduling policies and transmit power adjustments, which are used in the centralized emulation controller to determine the resulting packet scheduling and network topology.

The inner node core is enriched by the *Dependability* module, which provides the extra-functional abstraction layer (EFAL) for other modules. The EFAL provides fault modeling and injection of faults to ensure the proper execution of application code in the face of failures. For secure execution of applications, the EFAL provides threat modeling and threat injection mechanisms. The EFAL also enables dependability/security evaluation metrics for comprehensive evaluation and debugging of inner node interactions.

4.2 Inter Node Architecture

The connection of the nodes to the simulated world, the so called *central emulation controller*, is crucial to the architecture presented in Fig. 3. Generally, all node-to-environment and node-to-node interactions are exchanged using this connection. The connection is mainly supported by the adaptation layer on the side of the node and by the simulated physical world on the side of the central emulation controller. The former acts as a filter for the exchanged data such that only the information relevant to this node is incorporated and passed to the inner node modules. The latter defines the world model leading to physically correct information. This world model consists of a 3D model of the environment possessing real physical properties (e.g. friction, gravity). Moreover, communication links (basic RF propagation) and scenario settings can be respected. Considering our interest in search and rescue operations, the spread of substances/fire needs to be modeled; also an interaction with the environment is necessary. Such architecture leads to the information flow structure shown in Fig. 4.

A state space description of each node is applied which, e.g. for a mobile robot, describes its dynamical motion. At time t_k every node computes its own, desired change of state $\dot{\tilde{x}}_n$ using the node's own state $\tilde{x}_n$, control variable u and the relevant parameters θ_n. The relevant information for each node needs to be filtered out of the world information and adapted according to the properties of the node. As already mentioned, this adaptation is performed by the adaptation layer. This layer can work with real hardware or simulated virtual nodes. In the case of a pure simulation, threat, sensor and actuator models for the virtual node mimic the features of real sensors or actuators, resulting in a versatile structure and enabling realistic simulation.

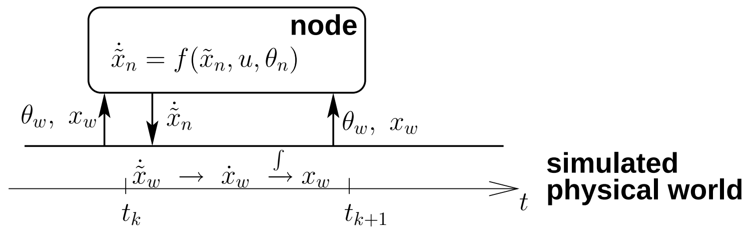
node
$\dot{\tilde{x}}_n = f(\tilde{x}_n, u, \theta_n)$
$\theta_w,\ x_w$
$\dot{\tilde{x}}_n$
$\theta_w,\ x_w$
$\dot{\tilde{x}}_w \rightarrow \dot{x}_w \xrightarrow{\int} x_w$
simulated physical world
t_k
t_{k+1}
t

Fig. 4. Information flow between the node and the central emulation controller.

After the computation of $\dot{\tilde{x}}_n$, each node transmits its desired change of state to the world simulation. Here, the desired changes of state of each node are combined to $\dot{\tilde{x}}_w$, the desired change of state of all nodes. Due to the fact that only local knowledge is available for each node, $\dot{\tilde{x}}_w$ is not necessarily reasonable. Thus, before computing the eventual change of state of each node $\dot{x}_w$, feasibility of $\dot{\tilde{x}}_w$ must be checked. The feasibility study is conducted by physical engines, e.g. PhysX™ by Ageia/nVIDIA or the Open Dynamics Engine ODE. Given an appropriate interpretation of $\dot{\tilde{x}}_w$ due to environmental properties, these engines can compute $\dot{x}_w$, excluding impossible movements this way. Additionally, a dependability interface provides the system with realistic fault/threat models. Similar to the inner node architecture, it investigates which, when and where to inject faults and threats [15] to influence the system behavior.

Dependability can simulate the probability of specific consequences, such as catastrophic failures. As simulation progresses, it is possible to observe 1) how the system evolves, 2) how different failures impact the system, and 3) how well the protocols handle security threats. Provided that some system properties are uncertain, the significance of those uncertainties can be determined. To the authors' knowledge, these dependability models have not been respected in the design of multi-robot system simulators before.

The above described inner node architecture enables real change of state of each node $\dot{x}_w$. A standard integration leads to the new state of all nodes x_w. Including the possibly altered parameters of the simulation and the environment θ_w, the state x_w is subsequently sent back to each node at time t_{k+1} and the simulation can proceed.

4.3 Visualization and Analysis

In general, efficient tracing, analysis, and visualization of log data is one of the main and important aspects of a simulation. Since spatial correlation is common in MMEs, the *MM-ulator* visualization abstractly presents the regions of interest instead of single sensor values. Maps are a natural way to describe the physical real world as well as the network world. The *MM-ulator* provides a Map-based World Model (MWM) [16] consisting of a stack of maps of relevant attributes (e.g., fault/threat map, connectivity map, residual energy map) (cf. Fig. 3).

The MWM abstracts different levels in *MM-ulator* such as communication issues and supports arbitrary applications. It allows efficient event detection, prediction and querying the network. The analysis based on MWM provides efficient mechanisms for predictive monitoring, proactive MME reconfiguration, enhancement of MME functionality, dependability and security.

4.4 First Implementation Steps

The screenshot outlined in the left part of Fig. 5 shows our search and rescue benchmark scenario in the MuRoSimF-based simulation [12] environment. Although, MuRoSimF with its origin in robot simulation provides detailed information on the physical environment and on the control/task states during the exploration phase, the aspects of wireless communication for robot interaction and remote sensor reading is not fully supported yet.

The right part of the figure shows the simulation of a homogeneous, static wireless sensor network (e.g. by using [19]) incorporating detailed protocol performance depending on sensor coverage and network connectivity for reliable event reporting. Based on the design proposed in 4.1 - 4.3 it is possible to integrate the communication characteristics of wireless multi-hop networks to MuRoSimF's dynamic environment models, providing more realistic radio propagation models as well as scenario dependent packet flows.

5 Conclusion

A novel architecture for a simulation environment has been proposed for emulation and validation of fundamental research topics from the diverse fields

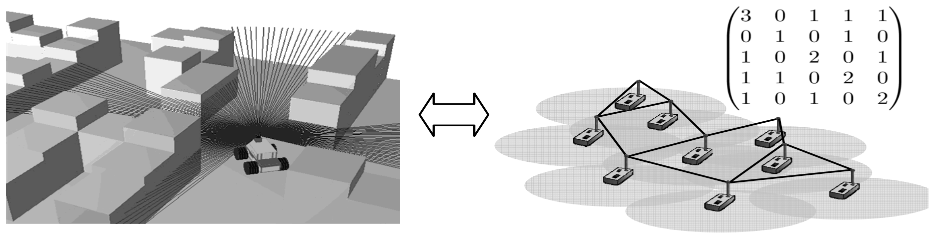
$\begin{pmatrix} 3 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 2 & 0 & 1 \\ 1 & 1 & 0 & 2 & 0 \\ 1 & 0 & 1 & 0 & 2 \end{pmatrix}$

Fig. 5. Robot simulation of a wheeled vehicle equipped with a laser scanner exploring an urban area (left). WSN simulation without locomotion properties (right).

involved in using heterogeneous networks of sensors and mobile robots in mixed mode environments. Motivated by various benefits of such a tool, a modular architecture has been presented to meet the different requirements and levels of realism in simulation. The architecture itself is comprised of a central emulation controller acting as the physical world and independent modules, incorporating the node specific characteristics, that are connected to this physical world emulation. Resulting in a highly scalable approach, this architecture respects issues that have not been considered before and is designed such that every node instance may either be simulated or real hardware equipment.

Future work will primarily deal with the implementation of this architecture as a stand-alone simulation tool extending existing simulators.

Acknowledgements. The authors thank J. Meyer, P. Schnitzspan, M. Andriluka, M. Friedmann, and A. Khelil for helpful discussions.

References

1. Epic games, unreal engine, www.epicgames.com. 2007.
2. Microsoft Robotics Studio, msdn.microsoft.com/robotics/, 2007.
3. S. Balakirsky, C. Scrapper, S. Carpin, and M. Lewis. USARSim: providing a framework for multi-robot performance evaluation. In *Proc. of PerMIS*, 2006.
4. S. Balakirsky, C. Scrapper, and E. Messina. Mobility open architecture simulation and tools environment. In *Proc. of the Intl. Conf. on Integration of Knowledge Intensive Multi-Agent Systems*, 2005.
5. D. S. Blank, D. Kumar, L. Meeden, and H. Yanco. Pyro: A python-based versatile programming environment for teaching robotics. *Journal of Educational Resources in Computing (JERIC)*, 2004.
6. S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper. USARSim: a robot simulator for research and education. In *Proc. of the 2007 IEEE Intl. Conf. on Robotics and Automation*, 2007.
7. T. H. J. Collett, B. A. MacDonald, and B. Gerkey. Player 2.0: Toward a practical robot programming framework. In *Proc. of the Australasian Conf. on Robotics and Automation*, 2005.

8. N. Cowan, G. Lopes, and D. Koditschek. Rigid body visual servoing using navigation functions. In *Proc. of the 39th IEEE Conf. on Decision and Control*, 2000.
9. M. Engel, B. Freisleben, M. Smith, and S. Hanemann. Wireless Ad-Hoc Network Emulation Using Microkernel-Based Virtual Linux Systems. In *Proc. of 5th EUROSIM Congress on Modeling and Simulation*, pages 198–203, 2004.
10. J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Trans. on Automatic Control*, 49(9):1465–1476, 2004.
11. J. Flynn, H. Tewari, and D. O'Mahony. JEmu: A Real Time Emulation System for Mobile Ad Hoc Networks. In *Proc. of the SCS Conf. on Communication Networks and Distributed Systems Modeling and Simulation*, pages 115–120, 2002.
12. M. Friedmann, K. Petersen, and O. v. Stryk. Adequate motion simulation and collision detection for soccer playing humanoid robots. In *Proc. 2nd Workshop on Humanoid Soccer Robots at the 2007 IEEE-RAS Intl. Conf. on Humanoid Robots.*
13. L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer. A system for simulation, emulation, and deployment of heterogeneous sensor networks. In *Proc. of the 1st Intl. Conf. on Embedded Networked Sensor Systems*, 2004.
14. R. He, M. Yuan, J. Hu, H. Zhang, Z. Kan, and J. Ma. A Real-time Scalable and Dynamical Test System for MANET. In *Proc. of 14th IEEE Conf. on Personal, Indoor and Mobile Radio Communications*, pages 1644–1648, 2003.
15. A. Johansson, B. Murphy, and N. Suri. On the impact of injection triggers for os robustness evaluation. In *The 18th IEEE Intl. Symp. on Software Reliability Engineering*, 2007.
16. A. Khelil, F. K. Shaikh, B. Ayari, and N. Suri. MWM: A map-based world model for event-driven wireless sensor networks. In *The 2nd ACM International Conference on Autonomic Computing and Communication Systems (to appear)*, 2008.
17. N. Koenig and A. Howard. Gazebo - 3D multiple robot simulator with dynamics, website http://playerstage.sourceforge.net/gazebo/gazebo.html. 2003.
18. T. Laue, K. Spiess, and T. Röfer. SimRobot - a general physical robot simulator and its application in RoboCup. In A. B. et al., editor, *RoboCup 2005: Robot Soccer World Cup IX*, number 4020 in Lecture Notes in AI, pages 173–183. Springer, 2005.
19. P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: accurate and scalable simulation of entire tinyos applications. In *Proc. of the 1st Intl. Conf. on Embedded Networked Sensor Systems*, pages 126–137, 2003.
20. T. Lin, S. F. Midkiff, and J. S. Park. A Dynamic Topology Switch for the Emulation of Wireless Mobile Ad Hoc Networks. In *Proc. of the 27th Annual IEEE Conf. on Local Computer Networks*, pages 791–798, 2002.
21. P. Mahadevan, A. Rodriguez, D. Becker, and A. Vahdat. MobiNet: A Scalable Emulation Infrastructure for Ad hoc and Wireless Networks. In *Proc. of Intl. Workshop on Wireless Traffic Measurements and Modeling*, pages 7–12, 2005.
22. S. Maier, D. Herrscher, and K. Rothermel. On Node Virtualization for Scalable Network Emulation. In *Proc. of Intl. Symp. on Performance Evaluation of Computer and Telecommunication Systems*, pages 917–928, 2005.
23. M. Matthes, H. Biehl, M. Lauer, and O. Drobnik. MASSIVE: An Emulation Environment for Mobile Ad-Hoc Networks. In *Proc. of IEEE 2nd Annual Conf. on Wireless On-demand Network Systems and Services*, pages 54–59, 2005.
24. O. Michel. Cyberbotics ltd. - webots(tm): Professional mobile robot simulation. *Intl. Journal of Advanced Robotic Systems*, 1(1):39–42, 2004.
25. M. Puzar and T. Plagemann. NEMAN: A Network Emulator for Mobile Ad-Hoc Networks. In *Proc. of 8th Intl. Conf. on Telecommunications*, 2005.

26. B. L. Titzer, D. Lee, and J. Palsberg. Avrora: Scalable sensor network simulation with precise timing. In *Proc. of 4th Intl. Conf. on Information Processing in Sensor Networks*, Los Angeles, 2005.
27. P. Zheng and L. M. Ni. EMWIN: Emulating a Mobile Wireless Network Using a Wired Network. In *Proc. of 5th ACM Intl. Workshop on Wireless Mobile Multimedia*, pages 64–71, 2002.