Seamless Multi-Robot Programming for the People: ASEBA and the Wireless Thymio II Robot

Philippe Rétornaz*, Fanny Riedo*, Stéphane Magnenat†, Florian Vaussard*, Michael Bonani‡, Francesco Mondada*

*École Polytechnique Fédérale de Lausanne
Laboratoire de Systèmes Robotiques
Email: firstname.lastname@epfl.ch

†ETH Zurich
Autonomous Systems Lab
Email: firstname@lastname.net

‡Association Mobsya
Email: firstname.lastname@mobsya.org

***Abstract*—Robots are an ideal tool for introducing programming to young generations. To be accessible to a large public, educational robots must be affordable and easy to use. In a previous work, the authors have developed Thymio II, an educational robot costing about 100 $. Thymio II is programmable using the ASEBA framework, which provides an interactive development experience through real-time compilation and inspection of the internal variables of the robot. However, this solution currently requires a USB cable connection between the robot and a computer, impairing the robot's mobility.**

This paper presents a radio-based wireless interface, allowing to program the Thymio II robot without the hassle of wires. This solution is transparent to the user, and implements the ASEBA protocol in a backward-compatible way. It is built on top of IEEE 802.15.4, costs a fraction of the robot's price, and does not affect its battery life significantly. After discussing the challenges and presenting the design of the interface, this paper shows performance results assessing the suitability of this interface for educational use.

The presented solution opens new perspectives for the use of robotics in schools from the first graders to the universities.

I. Introduction

Robots are an ideal tool for introducing programming to young generations, because they are both objects of fascination and machines with a rich set of interaction possibilities. In the past few years, the Laboratory of Robotics Systems of EPFL, in collaboration with the École Cantonale d'Arts de Lausanne (écal), developed the Thymio II robot (Fig. 1), a small and low-cost educational mobile robot [1], [2]. To be accessible to a large public, Thymio II was designed to be affordable while still providing many features including the ability to be programmed by its users. Thymio II integrates a wide range of sensing capabilities (infrared distance and ground sensors, a three-axis accelerometer, a microphone, touch buttons, etc.) and several actuators (two motors, about 40 LEDs, a speaker, etc.) allowing many different possible behaviours. These range from pre-programmed behaviours directly usable with the robot alone, such as obstacle avoidance, to user-defined behaviours through visual and text programming. Thymio II is an open-source/hardware project[1]. One unit costs about 100 $ and 2500 have been sold so far. This robot was successfully used with children of different age groups to introduce robotics and programming. This success is strongly linked to the low cost of the robot, which was achieved with some limitations in its features. In its sold configuration, a Thymio II robot has no means of communication with other robots, and must always be connected to a computer to be programmed or debugged. From a pedagogical point of view, it would be beneficial that users have a connection with the robot while it is moving around in a test environment, without having any cable connected. Indeed, being able to see and understand what the robot perceives while moving, and debugging its behaviour directly in the problematic conditions, is a key point in the experimental learning process. In addition, this could open new perspectives of multi-robot experiments, for instance in collaborative setups.

Fig. 1: The Thymio II robot (110x112x54mm).

Thymio II is programmed via the ASEBA Studio integrated

This research was supported by the Swiss National Science Foundation through the National Centre of Competence in Research Robotics.

[1] http://www.thymio.org

development environment (IDE), which is part of the open-source ASEBA framework[2]. ASEBA consists of an event-based virtual machine running on microcontrollers, an IDE with an embedded debugger, real-time compilation and visualization of the internal variables, and an easy-to-use scripting language. Compared to alternatives (such as Arduino [3] or ROS [4]), ASEBA allows the development flexibility of a virtual machine under the cost and energy-consumption constraints of a microcontroller [5]. This brings programming to inexpensive robots, which is of paramount importance in an educational context. As the ASEBA framework aims at safe operations of microcontroller-based research and educational robots (software should not be able to harm the robot), it has a strict protocol. Furthermore, being already deployed on a number of target platforms, this protocol can only be updated in a backward-compatible way. Finally, because of its optimised and tightly-integrated design, Thymio II has constraints of its own. In this paper, we show how we took these into account and developed a radio-frequency (RF) solution that is flexible, affordable and compatible with existing ASEBA-enabled robots.

ASEBA provides a solution for the programming and debugging of a heterogeneous network of microcontrollers. In particular, it distributes processing locally inside each node, permits their dynamic on-the-fly re-programming, and provides a global view through Studio, its IDE. ASEBA supports a wide range of physical transport protocols such as CAN, UART, USB and Bluetooth. This paper presents the porting of the ASEBA architecture to a wireless RF transport protocol. This is an non-trivial endeavour, because the ASEBA protocol currently makes assumptions, such as no loss of data, that do not hold in wireless networks. This paper presents a robust and transparent implementation of the ASEBA protocol over an RF network.

II. Related Work

The main features of ASEBA are its ability to re-program nodes dynamically, its support of heterogeneous node types, and its real-time event-based programming paradigm. Therefore, its wireless version lies in the field of sensor networks that can be dynamically re-programmed or re-configured. Hence, in this overview of related work, we focus on wireless sensor networks with dynamic programming capabilities and limit ourselves to low-power systems that can fit on a microcontroller with a few kB of RAM and flash memory.

A. Virtual machine

Virtual machines are the most common solutions to dynamically re-program nodes in a deployed wireless network [6], [7]. As these allow to change the running program by sending new bytecodes over the air without flashing the microcontrollers' firmware, they improve the adaptability and, as shown by Levis et al. [8], the general performances of the network.

Some authors demonstrated that using a virtual machine can reduce the local processing performances because of the bytecode interpretation overhead [9]. However, this can be avoided by using a mechanism to access native code directly from inside the virtual machine for the computationally-intensive algorithms without affecting the adaptability of the virtual machine. Because a virtual machine can provide highly-optimized primitives for processing-intensive operations, it can even increase the performances over naive C code in microcontrollers [5]. An additional strong point of virtual machine is their debugger support. As the bytecode is interpreted, it can easily be instrumented and remotely monitored to ease the debugging process. This is a decisive point when inexperienced people are programming the device.

B. Declarative programming

Some sensor networks have a declarative programming (SQL-like) paradigm [10]. This enables the programming of the sensor network as a unique entity and distributes the processing directly to the nodes. This is however best suited for sensor acquisition and is quite limited for actuation. The main limitation with such an architecture is that one cannot easily program a specific behaviour on each node. Such a network is designed to be programmed as a single instance of distributed sensors.

This is thus not entirely fitting our requirements, as each Thymio II robot within a network might demand a different individual behaviour. Moreover, debuging systems using this programming model is difficult for the inexperienced user, because it is declarative rather than imperative [11], and does not support common debugging tools such as breakpoints.

C. Native code generation

Some sensors network architectures focus on code re-usability. They are mainly template-based and output node-specific code which is then compiled into native code [12]. This model does not imply that the firmware in each node is fixed and not remotely updatable. Some dynamic linking can take place in order to rewrite only some part of the firmware [13].

This is an interesting approach if the node behaviours do not change often. However, when the behaviour needs to change often while operating in standard conditions, this approach is sub-optimal: The re-programmability of most low-power microcontrollers is limited (less than 1000 cycles for some) and consumes a significant energy. Moreover, Lombriser et al. [9] showed that using native code to remotely update a node uses more bandwidth than with a virtual machine, leading to lower performances. While it is possible to debug such code, it is quite difficult with microcontrollers to remotely single steps code or debug memory access. Thus, for the

[2] http://aseba.wikidot.com

inexperienced programmer, the learning curve is harder than with the solution based on virtual machines.

III. ASEBA WIRELESS CHALLENGES

The usefulness of ASEBA to script the behaviour of a swarm of robots has been demonstrated in a previous work [14]. This work was using E-Puck robots [15], connected through Bluetooth. The main drawbacks of that work were the robot itself, as the E-Puck costs more than 800 $, and the Bluetooth protocol, which limits the number of robots in a network to a maximum of 7. Furthermore, all data transited through a central computer, doubling the delay compared to a broadcasting-based approach. Although the main target of the current work is the Thymio II robot, the proposed protocol shall be generic for any ASEBA network. The microcontroller of the Thymio II can only handle simple serial protocols and no complex packets re-transmission or routing because of memory constraints. The added electronics must encapsulate all the RF protocol and provide a simple interface using the native ASEBA protocol.

A. Cost

In the educational context, the cost is a critical factor to have a wide acceptance in schools and families. The wireless addition to the Thymio II robot must therefore be as low cost as possible. As the targeted production volume is moderate (starting with batches of 1000 units), the cost computation is not limited to the different electronic chips but must include the industrialization, the tooling of the printed circuit board (PCB) as well as the certification and needed licenses. Furthermore, this analysis must include any necessary additional hardware to have a functional setup, such as a USB dongle on the computer side.

B. Ease of use

As the Thymio II robot is used by children, the ease of use is very important. A simple configuration step is too much to handle and would discourage the user. Thus the wireless setup must be truly plug and play, without any intermediate steps. If any complex network configuration is needed, such as having different separate networks in the same area, this configuration should be manageable by people unfamiliar with technology. Moreover, as users might use Thymio II both wirelessly and through the USB cable, these two interfaces must behave the same when accessed from a computer. In addition, the user should be able to switch from one to the other during the same programming session, without loosing her work. This use case is realistic, as the user may need to recharge the robot through USB.

C. Compatibility

This requirement is linked to the previous one, but with deeper consequences in the protocol implementation. ASEBA provides multiple software tools, including integration with other frameworks such as ROS. Because the wireless layer must be compatible with all these, and because ASEBA is already employed in several robots [14], [16], [17], [18], [19], the protocol can only change in a backward-compatible way.

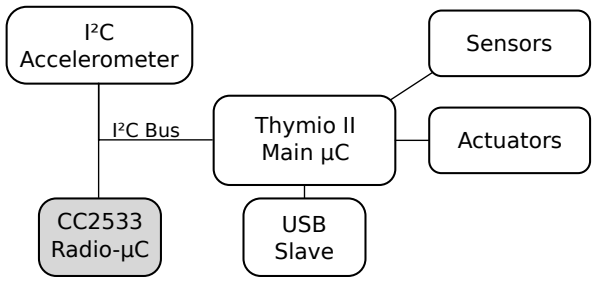

Fig. 2: A diagram of the Thymio II electronics architecture.

D. Low power

The ASEBA protocol is meant to be used on robots with actuators and sensors. The power consumption while the robot is running is thus not a critical factor as long as it stays moderate. For instance, the power consumption of the Thymio II robot is about 1 W while switched on. A wireless module consuming less than 100 mW is therefore perfectly acceptable.

IV. THYMIO II WIRELESS

The Thymio II robot has an internal extension connector providing an I^2C bus and battery power. Any additional module needs to use this connector. A schematic view of the hardware connection between the Thymio II main microcontroller and its various peripherals is shown in Fig. 2. The physical constraints inside the robot are tight, which limits the PCB area available for the wireless module. One additional constraint is the powered-off consumption. As the Thymio II electronics cannot power down the wireless module, an efficient software power down must be implemented. The wireless module must also be able to wake up through an I^2C bus access (as shown in Fig. 2).

A. Hardware

We selected the radio integrated circuit (IC) among the vast choice offered by almost all major semiconductor manufacturers based on the following needs:

- a robust modulation in the 2.4 GHz worldwide ISM band,
- a low total production cost, including bill of materials (BOM), licenses and needed certification,
- a native USB interface (computer side),
- a native I^2C interface (robot side),
- an ultra-low powered-down consumption,
- at least 4 kB of RAM and 64 kB of Flash memory,
- a 3.3 V power supply,
- a C compiler available.

The only IC family fitting all these needs at the time of development was the CC253x system-on-chip (SoC) from Texas Instruments. It is a IEEE 802.15.4-compatible SoC, its modulation is *direct sequence spread spectrum*, which coexists quite well with other radio equipment such as IEEE 802.11 [20]. This circuit is low-cost and requires little external components: a voltage regulator, an RF impedance matching network, a crystal and some decoupling capacitors. The IC family provides a version of the SoC for mobile devices with an I^2C interface and a 400 nA powered-down current, which is critical in our application. The same family proposes a variant with a USB interface instead of I^2C. This eases the software development as most of the code can be shared between the USB dongle and the Thymio II RF interface.

This SoC is supported by the Contiki [21] embedded operating system and the open-source C compiler SDCC[3], reducing the development time and cost. The only drawback is that this SoC is based on a 8051 core, which is a weak 8-bit microcontroller with only a small available stack space (about 200 bytes usable). This affects the software design as only a limited amount of nested functions is possible on this architecture.

Since this module uses the bare IEEE 802.15.4 protocol, no additional licensing or certification is needed apart from mandatory ones, such as ETSI in Europe or FCC in the United States.

1) Computer side: A desktop computer or a tablet connects to the robots through a USB dongle, based on a CC2531 SoC. The total BOM is less than 5 $ per unit in quantity of 1000. The electronics can be produced on a two-layer PCB with all the components on one face, thus reducing the production cost to a minimum. An integrated PCB antenna is used (MIFA type), reducing the cost of the radio interface to its impedance-matching network.

The SoC is able to directly interface with the USB protocol and encapsulates the RF protocol transparently, providing only a virtual serial port interface to the PC (using standard CDC-ACM USB class). It is completely independent from the Thymio II robot and can be used to interface any ASEBA network over IEEE 802.15.4 without any change on the computer side.

2) Robot side: The Thymio II side of the radio link is similar to the USB dongle. The only changes are a different SoC providing an I^2C communication interface, more suited for embedded systems, and a different PCB antenna (inverted F antenna) because of size constraints. We took special care of the deep sleep power consumption. The Thymio II software power-down current consumption is 120 μA. The RF module consumes less than 1 μA in power-down mode, which has a negligible effect on the robot's power-down battery life.

The total BOM is also less than 5 $ for a production batch of 1000 units. This leads to a total cost of less than 10 $ for the complete solution (USB dongle + robot's RF module).

[3] http://sdcc.sf.net

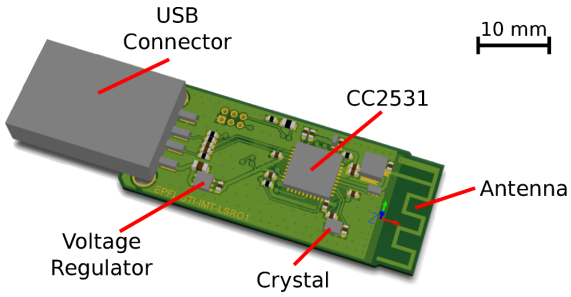

Fig. 3: A 3-D view of the USB dongle (computer side)

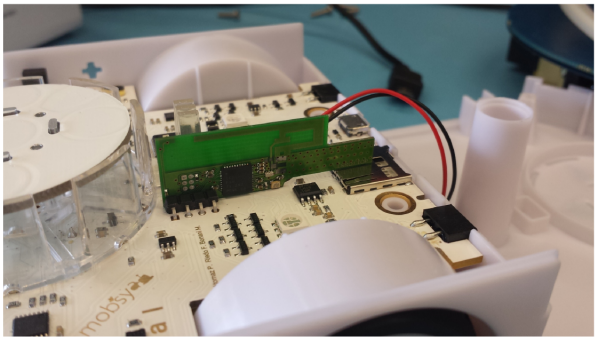

Fig. 4: The RF module (vertical green PCB) fitted inside the Thymio II robot

B. Software

We use the Contiki [21] operating system on both modules. We cannot use the standard IP6 over IEEE 802.15.4 stack 6LowPan [22], because:

- The relative high overhead of the IP protocol would reduce the usable data payload and would produce a lot of fragmentation, leading to poor real-time performances.
- The mesh capability of the 6LowPan is not needed.
- The amount of configuration work needed for a fully-working 6LowPan network is too much a burden for a plug and play radio communication.

Therefore, we developed an ASEBA network protocol directly on top of the IEEE 802.15.4 layer.

1) Protocol: The ASEBA framework requires two communication modes: broadcast and point-to-point. The first one is used when two robots exchange events. In this mode, events embed an identifier and a variable-length payload. The second one is used between the debugger (running on a desktop computer) and a specific node. It is assumed that only one debugger is active on the network at any given time. If no desktop computers are present, no such communication can happen. Therefore, using unicast and relying on the IEEE 802.15.4 acknowledgment mechanism is sufficient to ensure a safe delivery of each packet.

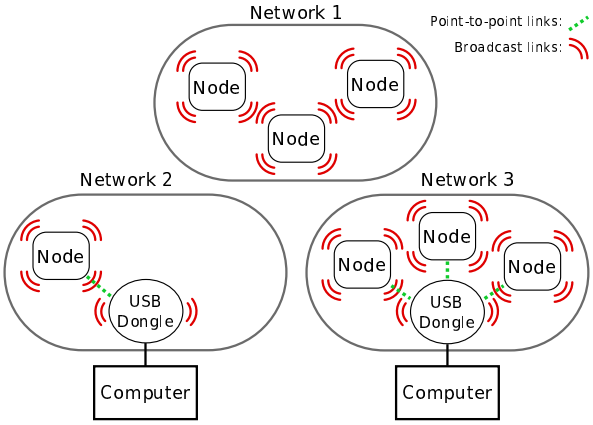

Fig. 5: Different ASEBA networks configurations

2) Multiple co-existing networks: One main target use of the Thymio II RF interface is a classroom, with several distinct networks in the same room (Fig. 5). We implement separation between different networks using the IEEE 802.15.4 personal area network identifier (PAN ID) in addition to a separate radio channel. Currently only three different channels are used to minimize cross-talk with main Wi-Fi channels, but our system can be extended to the full sixteen channels supported by IEEE 802.15.4. In the example of Fig. 5, a first network is completely standalone, with multiple nodes exchanging events without any central authority. A second network is formed by just one node and a computer debugging/programming this node. A third network is a mixture of the first and second cases, where the computer is programming a whole set of nodes. The ASEBA framework enables the user to switch on-the-fly between the different network topologies just by powering on and off the different elements. This network is based on unslotted IEEE 802.15.4, a detailed performance analysis of such networks can be found in [23].

The user can change the PAN ID and the radio channel of the USB dongle using a simple configuration running on a desktop computer. This tool puts the dongle in a presence-beacon–broadcasting mode. The user can then put the robot in a "pairing mode" by approaching it to the dongle and by holding two buttons for 5 seconds. Doing so will make the robot scan every IEEE 802.15.4 channel for presence beacons. Using the received strength of the beacon signal, the robot will automatically re-configure itself to join the closest network. Two LEDs blinking in a synchronized way on the USB dongle and the Thymio II robot give a feedback to the user. This procedure has been designed to be easily performed by people unfamiliar with technology, such as first grade teachers.

3) Compatibility: As the goal of the presented extension is to give wireless communication capabilities to existing ASEBA robots, its compatibility with the ASEBA framework was a pre-requisite. The developed solution is fully compatible and does not require any interface change on the computer side. The USB dongle transparently emulates the ASEBA serial protocol used by the physical USB connection on the Thymio II. It encapsulates the full RF stack and performs packet reassembly when needed. Therefore, all existing installations of ASEBA can directly use the RF module. This decreases the maintenance burden, especially in schools where users do not have administrative rights on the computers to install additional software, drivers or change network settings.

Moreover, our solution provides a smooth transition from programming through a USB connection, then switching to the RF one, and finally going back to the USB one for recharging the battery. These transitions are done by plugging and unplugging the dongle or the robot's USB connection, and do not require any configuration change.

V. Experimental Validation

This section shows the performances of the ASEBA wireless interface. These are mainly limited by the current implementation on the Thymio II robot. Indeed, the 400 kHz I^2C communication bus between the robot's main microcontroller and the RF module is the weakest part in terms of bandwidth. This communication bus is shared with others sensors, further limiting the bandwidth dedicated to the radio communication. This I^2C bus is based on a master/slave architecture and does not provide any interrupt line, forcing the main microcontroller to constantly poll the radio microcontroller to check if some data are available. The chosen polling frequency is 10 ms because the main use case of the Thymio II robot is interacting with humans. As the next section shows, this has a strong influence on the latency of the exchanged messages.

A. Latency

Fig. 6 shows an histogram over 600'000 measurements of the latency between two Thymio II robots at a distance of 30 cm. Since the robot is the limiting factor we performed the evaluation on the worst case scenario of a communication between two robots. The latency between a PC and a robot, while not experimentally evaluated, should be better while staying in the same magnitude order. One robot was emitting a "Ping" event while the second robot was emitting a "Pong" event immediately after receiving the "Ping" event. This experiment measures the latency of the whole communication stack: from the main microcontroller down to the RF layer and back. The achieved latency is 20 ms, which is fully expected because the main delay in the system is the 10 ms polling latency on each robot. Therefore, this relatively large latency is specific to the Thymio II robot and would be lower in a robot with an available interrupt channel.

B. Events rate

Fig. 7 shows the event throughput given the event's payload size, between two robots. The performance between one robot and a PC would be the same as the weakest link limit the performance of the whole communication chain. The

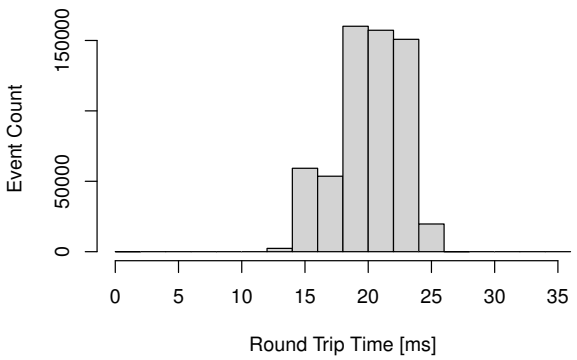

Fig. 6: The latency between two Thymio II robots.

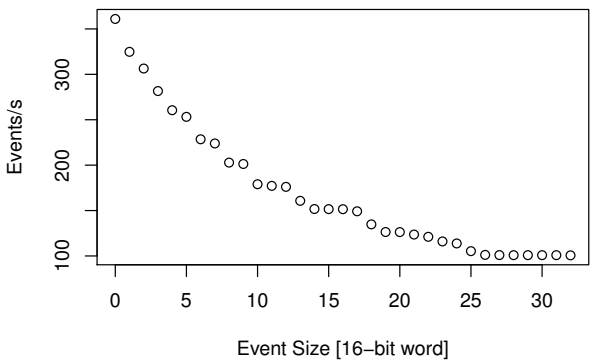

Fig. 7: The event throughput in function of payload size.

maximum size that a Thymio II robot can send is 32 16-bit words. The minimum achieved throughput is 100 events/s, which given that the fastest internal event-generation rate in the robot is 100 Hz, is sufficient. Moreover, in most situations fewer than 32 words are required. As we demonstrated in [5], the bandwidth and the bus utilization can be dramatically lowered by adopting an event-based control policy, therefore the throughput is not a limiting factor in an ASEBA network. Thus the network is limited by the events rate and not by the number of robots.

While Fig. 7 shows results with a single network of two robots, but several will be used concurrently in a classroom. Because of the limited speed of the I^2C bus, the current system uses only about 25 % of the physical bandwidth (250 kbps) of the IEEE 802.15.4 physical layer. Therefore, 3 to 4 robots can be used on a single channel without much effect on the usable bandwidth. Hence, using three different channels accommodates up to 12 robots without any loss of performance. With more robots, performances will decrease but will most-probably still be sufficient for educational use. If full speed in required for more than 12 robots, additional IEEE 802.15.4 channels can be used, at the price of increased collisions with Wi-Fi.

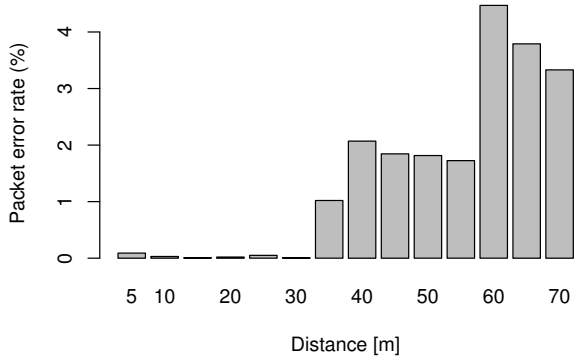

Fig. 8: The packet error rate in function of distance.

C. Range

Fig. 8 shows the dropped event rate in function of the distance. The measurement was performed in an obstacle-free environment between two robots over 10'000 events. We see that the rate is below 1 ‰ up to 35 meters, which is beyond the target range of a classroom. The dropped packet count was in the measurement noise for the first 30 meters. Therefore, the measured range completely fulfils the requirements.

In addition, we performed a test with a Thymio II enclosed inside a freezer. Indeed, the robot is fitted with a temperature sensor, allowing an interesting educational experiment in which the robot measures the temperature in different places. We verified that the robot was still able to communicate smoothly with the USB dongle in this situation.

D. Ease of use

To deploy the wireless interface, the only adaptation needed to the existing tools is to update the Thymio II's firmware. All other tools will automatically be compatible, validating that the presented solution is truly plug and play. In summary, the system is usable in a wide variety of configurations:

- computer to one robot (Fig. 5, network 2), when developing a behaviour for a single robot;
- computer to multiple robots (Fig. 5, network 3), for instance when several robots are used to animate a LEGO structure;
- many to many robots, no computer (Fig. 5, network 1), for instance to demonstrate bio-inspired collective behaviour, such as flocking [24].

VI. Conclusion

Using the presented solution, a user unfamiliar with technology can program, debug, and monitor a network of wireless robots easily, without any prior technical knowledge. Thymio II and ASEBA are open-hardware/source projects, and ASEBA runs on all major operating system, including Android tablets. We plan to industrialize the wireless module and release it under an open-hardware license. Therefore, its diffusion will not be encumbered by expensive fees or

restrictive licenses. This is important for the adoption in public schools, which run on tight budgets and cannot afford non-sustainable, restrictive solutions. Therefore, we believe that the combination of Thymio II, ASEBA and the wireless interface is a significant progress to the diffusion of robotics and programming activities in schools.

REFERENCES

[1] S. Magnenat, F. Riedo, M. Bonani, and F. Mondada, "A programming workshop using the robot "thymio II": The effect on the understanding by children," in *Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on*, pp. 24–29, IEEE, 2012.

[2] F. Riedo, P. Rétornaz, L. Bergeron, N. Nyffeler, and F. Mondada, "A two years informal learning experience using the thymio robot," in *Advances in Autonomous Mini Robots*, pp. 37–48, Springer, 2012.

[3] R. Balogh, "Educational robotic platform based on arduino," in *Proceedings of the 1st international conference on Robotics in Education, RiE2010. FEI STU, Slovakia*, pp. 119–122, 2010.

[4] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, 2009.

[5] S. Magnenat, P. Rétornaz, M. Bonani, V. Longchamp, and F. Mondada, "Aseba: a modular architecture for event-based control of complex robots," *Mechatronics, IEEE/ASME Transactions on*, no. 99, pp. 1–9, 2010.

[6] P. Levis and D. Culler, "Maté: a tiny virtual machine for sensor networks," in *ACM Sigplan Notices*, vol. 37, pp. 85–95, ACM, 2002.

[7] R. Muller, G. Alonso, and D. Kossmann, "A virtual machine for sensor networks," *ACM SIGOPS Operating Systems Review*, vol. 41, no. 3, pp. 145–158, 2007.

[8] P. Levis, D. Gay, and D. Culler, "Active sensor networks," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pp. 343–356, USENIX Association, 2005.

[9] C. Lombriser, D. Roggen, M. Stager, and G. Troster, "Titan: A tiny task network for dynamically reconfigurable heterogeneous sensor networks," in *Kommunikation in Verteilten Systemen (KiVS)*, pp. 127–138, Springer, 2007.

[10] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "Tinydb: an acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems (TODS)*, vol. 30, no. 1, pp. 122–173, 2005.

[11] P. Pichler, B. Weber, S. Zugal, J. Pinggera, J. Mendling, and H. Reijers, "Imperative versus Declarative Process Modeling Languages: An Empirical Investigation," in *Business Process Management Workshops*, vol. 99 of *Lecture Notes in Business Information Processing*, pp. 383–394, Springer Berlin Heidelberg, 2012.

[12] C. Buckl, S. Sommer, A. Scholz, A. Knoll, and A. Kemper, "Generating a tailored middleware for wireless sensor network applications," in *Sensor Networks, Ubiquitous and Trustworthy Computing, 2008. SUTC'08. IEEE International Conference on*, pp. 162–169, IEEE, 2008.

[13] A. Dunkels, N. Finne, J. Eriksson, and T. Voigt, "Run-time dynamic linking for reprogramming wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, pp. 15–28, ACM, 2006.

[14] S. Magnenat, P. Rétornaz, B. Noris, and F. Mondada, "Scripting the swarm: event-based control of microcontroller-based robots.," in *SIMPAR 2008 Workshop Proceedings*, 2008.

[15] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. 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*, vol. 1, pp. 59–65, 2009.

[16] M. Bonani, V. Longchamp, S. Magnenat, P. Rétornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada, "The marxbot, a miniature mobile robot opening new perspectives for the collective-robotic research," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 4187–4193, IEEE, 2010.

[17] M. Bonani, S. Magnenat, P. Rétornaz, and F. Mondada, "The handbot, a robot design for simultaneous climbing and manipulation," in *Intelligent Robotics and Applications*, pp. 11–22, Springer, 2009.

[18] J. Fink, F. C. Vaussard, P. Rétornaz, A. Berthoud, F. Wille, F. Mondada, and P. Dillenbourg, "Motivating Children to Tidy up their Toys with a Robotic Box," in *HRI pioneers workshop*, 2013.

[19] F. Rochat, P. Schoeneich, O. Marti, H. Bleuler, and F. Mondada, "Cy-mag3De: magnetic climbing inspection robot," in *Field Robotics: Proceedings of the 14th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, pp. 407–414, World Scientific, 2011.

[20] D. Yang, Y. Xu, and M. Gidlund, "Wireless coexistence between ieee 802.11-and ieee 802.15. 4-based networks: A survey," *International Journal of Distributed Sensor Networks*, vol. 2011, 2011.

[21] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pp. 455–462, IEEE, 2004.

[22] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of ipv6 packets over ieee 802.15.4 networks," *IETF RFC*, vol. 4944, September 2007.

[23] B. Lauwens, B. Scheers, and A. Capelle, "Performance analysis of unslotted CSMA/CA in wireless networks," *Telecommunication Systems*, vol. 44, no. 1-2, p. 109–123, 2010.

[24] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, pp. 25–34, ACM, 1987.