Printable Modular Robot: An Application of Rapid Prototyping for Flexible Robot Design

Dennis Krupke, Florens Wasserfall, Norman Hendrich and Jianwei Zhang
Department of Informatics, Group TAMS, University of Hamburg, Hamburg, Germany

Abstract

Purpose – The Printable Modular Robot (PMR) is a highly customizable, modular, snake-like robot platform for research and education. The robot can be assembled and re-assembled on the fly and automatically detects changes in its topology during operation.

Design/methodology/approach – The robot consists of a number of autonomous modules coupled by magnetic interfaces. Each module combines 3D printed mechanical parts with widely available standard electronic components, including a microcontroller and a single servo actuator. The mechanical and electrical connection is provided by a set of magnets which generate the physical force between the modules and at the same time serve as wires for power and communication.

Findings – The PMR is a fully equipped robotic device, well integrated into a simulation framework, capable to execute common locomotion patterns but still very affordable ($\sim$25 \$ per module). Furthermore, the design is easy to extend and replicate for other research and education groups.

Originality/value – This paper explores a novel approach of connecting robot modules by utilizing very simple magnetic parts. A second focus lies on the concept of closely integrating simulation and hardware development, blurring the edge between the digital and physical world.

Keywords Modular robot; Rapid prototyping; Topology detection; Reconfigurable robot.

Paper type Technical paper

1. Introduction

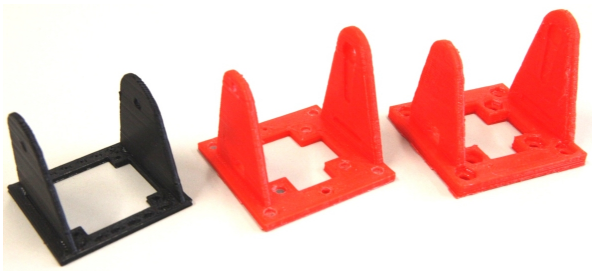
Rapid prototyping and 3D printing in particular have become an ubiquitous topic in the last decade and the technology to generate free-form parts on demand is now available to basically every researcher and engineer. In this paper, we describe our approach to create an open source, easy to prototype modular robot, intended for use in research and education. The modules are optimized for printing and include only standard hardware parts. Module attachment is realised via a novel design that uses magnets for both the mechanical and the electrical interconnection.

All module specifications and design files are open source, allowing students to develop and modify the design and thus learn how to go about creating a robot. The basic design criteria are inspired by the Miniskybot project (Gonzalez-Gomez et al., 2011). The proposed robot is designed for chain-configurations, where the modules can be connected in arbitrary combinations of pitching and yawing joint configurations (Zhang et al., 2009; Hirose and Yamada, 2009). According to the classification of reconfigurable modular robots by Murata and Kurokawa (2012), the proposed robot is positioned between class 2 and class 3, as a semi-self-reconfigurable modular robot. Reconfiguring the physical topology requires manual interaction, but can be performed during operation. The robot automatically recognizes its topology and actively adopts changes to the running locomotion patterns. This concept is similar to the CoSMO modular robot design (Liedke et al., 2013), but focusses on providing only essential functionalities required for locomotion at a fraction of the costs.

2. From simulation to reality and back

Based on previous work (Krupke et al., 2012), we used an extension of the OpenRAVE simulation framework (Diankov, 2010) for the optimization of the modular robot design and locomotion. To simulate the behaviour

of servo motors, the OpenMR (González-Gómez, 2014) plugin is utilized. Realistic robot motion is calculated by the ODE physics engine (Smith, 2014). Arbitrary robot models are easy to integrate into this system, because standardized robot definition files are used to define a robot from 3D model files and joint definitions. The very same 3D models (.stl files) are used for both



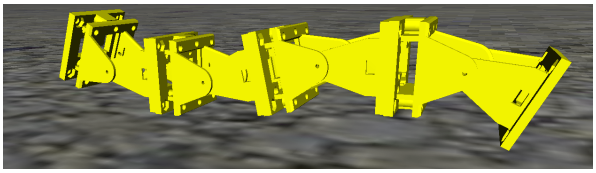


Figure 1 – **Upper:** Evolution of the robot design. Three versions of the lower module. Note the increased wall thickness in the second generation and improved magnet slots in the third generation. **Lower:** Simulation of mechanical assembly and locomotion in the OpenRAVE framework.

the simulation system and the 3D printing. Thus, the simulation automatically captures the mechanical properties of the real housing of a specific robot module. The system has been used to create, test and optimize the generation of locomotion patterns, based on the specific configuration of the robot. Figure 1 shows a simulated robot in locomotion with the same 3D models that have been printed to create the real robot and a series of three iterations of improvement of one part of the robot, with the inner magnets added during the second generation and increased thickness of the base for precise alignment of the magnets in the final design. This concept allows for simulation and reality to be as close as possible. The close coupling of 3D models, simulation and real hardware substantially aids the development process. Robots created only for simulation can be printed if they were successful in simulations. In turn, existing robots can be simulated (if their 3D models are available) and possibly modified to achieve better results for the next production series.

We identified the development cycle drafted in Figure 2 as an abstract, recurring core pattern of this process. The iterations in hardware design and improvement can be verified against the simulation in short intervals. The results are effortlessly printable, bypassing the process of exporting and preparing the data or even waiting for an external supplier. Surprisingly, the bottleneck in this continuous two-stage verification process is not the printing time itself, but the manual assembly of hardware components. Printing an entire set of modules can be done e.g. over night, since it does not require human interaction. Both the iterative process and the estimated amount of time for a single iteration of one part are comparable to the findings in (Macdonald et al., 2014), where the time to part is estimated to be within a range of 7.5-18 hours.

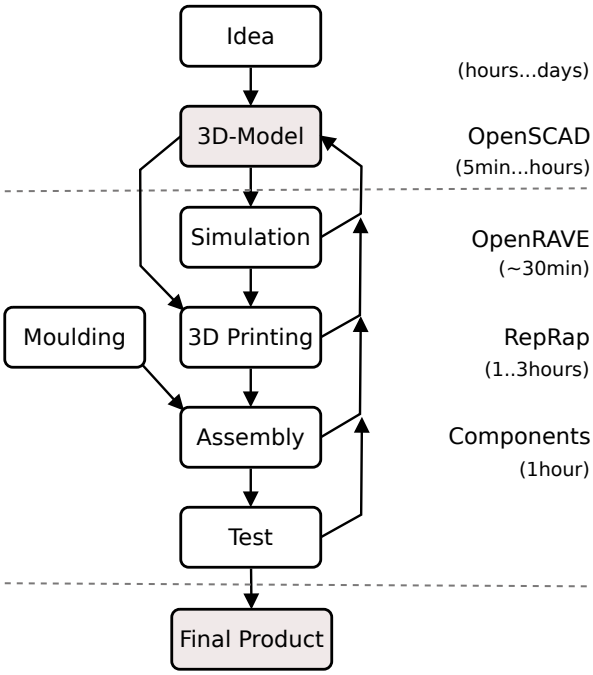


Figure 2 – Iterative hardware development cycle. The initial draft can be tested and optimised against the simulated environment in short iterations. When the virtual parts fulfil the desired requirements, they can be materialised by the 3D printer. Complete modules are then assembled and evaluated in the real world and the development cycle starts over again with extended requirements or remaining insufficiencies. For final product releases, the 3D printing can be replaced by a moulding process.

3. Module hardware and design

The printable 3D models for the plastic parts of the robot are designed in OpenSCAD. OpenSCAD is an open source solid 3D CAD modelling system for programmers. Objects are created with Constructive Solid

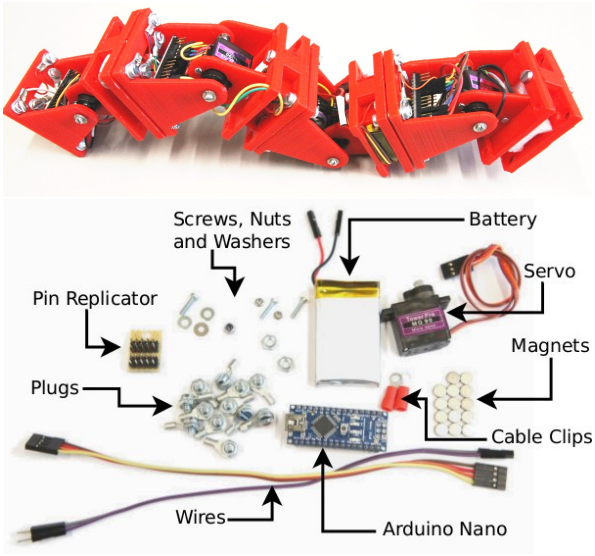


Figure 3 – **Upper**: Assembled robot with 4 modules in all-pitch configuration. **Lower**: Standard components of one PMR module. One servo motor and one Arduino board are needed for each module; batteries are optional but at least one battery is needed for a whole robot.

Geometry (CSG), using a scripting language. Standard components like nuts, servos and microcontroller boards are directly integrated into the module design. They can be easily applied after printing and fit accurately into the custom shaped slots. Parametrization in the scripting language allows for creating generic packages that can be reused for different sized modules and additional hardware.

Each PMR module consists of at least three printed plastic parts. The battery slot is prepared to attach printed tactile feet sensors by a snap-in mechanism. Single modules are mechanically connected by permanent magnets for a fast and easy change of topology. These magnets also provide electrical contacts between adjacent modules on the outer side and to the internal wiring on the inner side, by locking the ferromagnetic screws in place which are attached to the wires. Communication is implemented via serial interfaces which are integrated into Arduino boards in hardware and software.

3.1. 3D printing

The 3D printer used for this project is a custom designed device, based on the Reprap design (Jones et al., 2011). The design has been extended to support simultaneous printing with three extruders for a wide range of extruder characteristics and materials. Objects can be printed from PLA- or ABS-plastic. PVA is used for water soluble support structures. We incorporated experimental adaptive slicing algorithms to achieve high precision prints on affordable hardware. The resolution of the printer's z-axis and extrusion diameter is dynamically modified, depending on the local structure of the object.

3.2. Hardware components

Each module combines the printed mechanical parts, one servo for locomotion, one Arduino Nano V3 board for control and one extension slot, which either holds a battery (lower picture in Figure 3) or other hardware, particularly sensors. The batteries are connected in parallel as a shared power supply. This allows for a subset of modules to operate autonomously but does not require a battery on every module which saves space for sensors or other cargo. The Arduino Nano controller was chosen for its small form factor and affordable price, as well as the extensive collection of open-source software and drivers. It also benefits from easy programmability via USB using the Arduino IDE. All hardware components are open-source (Arduino, CAD-models) or standard parts (servos, magnets). Specifications of the key-components are summarised in table 1.

Control board	Arduino Nano V3 (ATmega328p, 16 MHz, 32 KB flash memory, USB, I2C, RS232 (Hard-/Software)
Servo	Micro servo (Tower Pro MG90, 2 kg/cm torque at 4.8 V)
Weight	$\approx$ 0.12 kg (completely assembled)
Power supply	Low cost two-cell LiPo batteries (7.4 V, 820 mAh)
Magnets	Neodymium (N52, r=3 mm, h=2 mm, $\approx$ 20 N force)
Price	20-25 $ (per module)

Table 1 – Specification of the hardware components.

3.3. Magnetic module interconnection

The key feature of the robot is the combined mechanical and electrical module attachment via a set of eight magnets on each end of the module, as illustrated in Figure 4. We combined the concept of a magnetic connector introduced by Nagy and Abbott (2007) with the approach of a redundant pin layout, comparable to the work of Yim et al. (2000). At least two-times redundant layouts are required in order to realise modules that allow for combinations of pitching and yawing orientations.

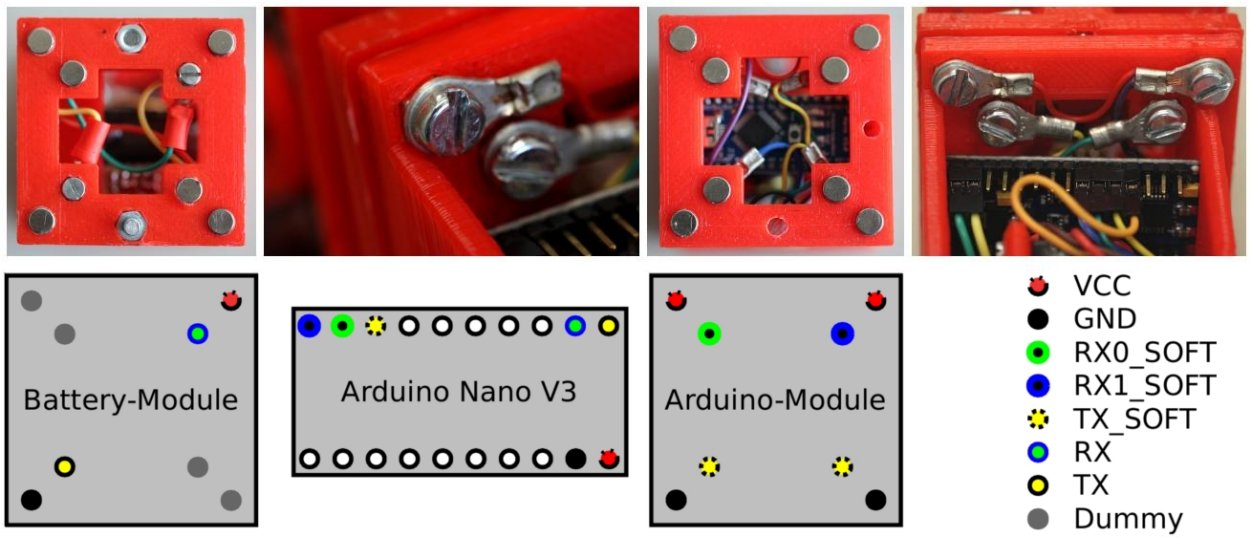


Figure 4 – **Upper (from left to right):** Male connection-face, close-up of the wire connection, female connection-face, back side of a female connection-face. Permanent magnets provide both electrical and physical contacts. The combination of screws, nuts and cable clips build a plug-system that allows for attaching wires from the backside of the connection-faces to the magnets. The male connection-face has a special feature, where contacts of RX and TX are not fixed but can move slightly to establish robust contact. **Lower:** Wiring scheme of the bus interface. Both faceplates at each end of a module are connected to the Arduino board. Two different receive pins at the female faceplate are used to determine the orientation of the neighbouring module.

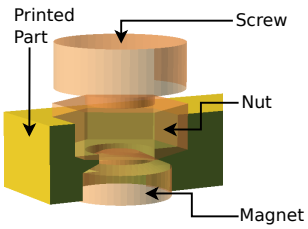


Figure 5 – Cross section of one magnetic connector. The magnet is glued into the narrowing slot, the screw/nut is fixed in its slot by magnetic force only.

The internal wiring is realised with standard ferromagnetic screws and nuts, attached to the wires and held in their slots by the magnets (Figure 5). The batteries are connected in parallel as a shared power supply to allow a subset of modules to operate autonomously without consuming the entire cargo capacity. Due to the precise alignment of the magnets in the printed parts, the contact force generated by the magnets is strong enough to reliably connect the modules during robot movements. On the other hand, the user can connect and disconnect modules easily at any time and without tools. To allow for dynamic changes of the modules' pitch/yaw orientation, a special hardware and software interface was designed. Requirements were a rotatable pin layout and dynamic initialisation of the communication lines. The faceplates are physically genderless, but logically directed (by the UART-assignment). To prevent invalid rotations (shorting VCC and GND), an additional bolt and two corresponding holes are added to the faceplates. This restriction could of course be solved using a more complex concentric magnet layout or more complicated connectors. Note that attaching two faceplates of the same type is impossible anyway due to the magnets' polarization.

4. Control architecture

A complete robot consists of a master-module, located at the male end of the chain, and a number of 0-14 slave modules. The number of modules is limited by the current protocol's address space of 4 bit. A higher number of modules is possible with a modified protocol. Technically, every module is able to act as master, depending on the software configuration. Due to the Arduino's limited computing power, and due to easier programming and teleoperation, it is necessary to connect an external controller for remote control or real-world applications. This occupies one of the two possible UART-interfaces and therefore requires the master to reside at the end of the chain. An external controller can be carried inside the master's cargo-slot for autonomous operation, or connected wireless by Bluetooth.

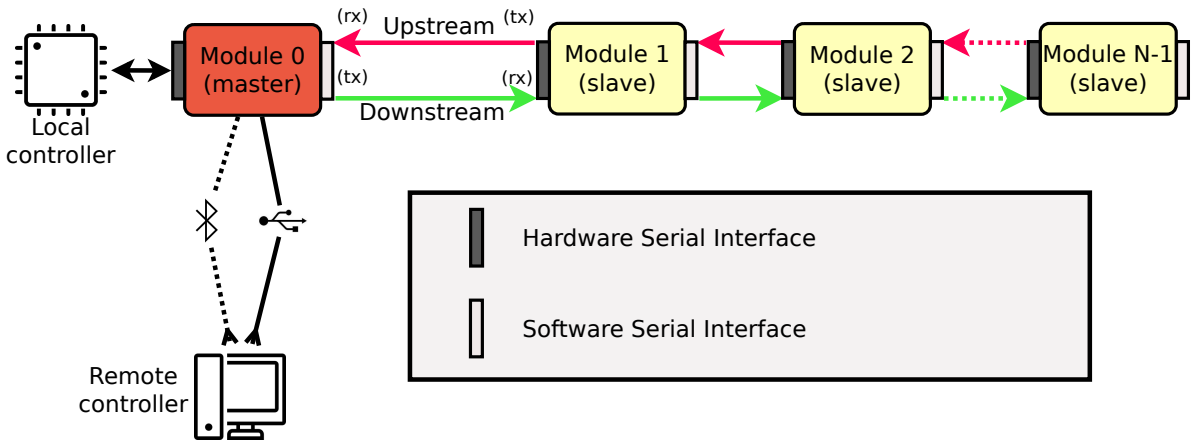


Figure 6 – Diagram of the robot's hardware- and communication topology. The master module is required to be at the end of the chain. It is usually connected to an external controller by serial interface: either directly to an on-board micro-controller (*Local controller*), or via Bluetooth/USB to stationary hardware (*Remote controller*). Every pair of modules is connected by a combination of one hardware- and one software-UART, to allow for dynamic pin assignment and pitch/yaw topology detection.

4.1. Topology management

The robot is organized in a semi-dynamic master-slave-architecture. Every node has a unique address, assigned in software during the programming process. The master node stores and manages current information about order and orientation of every attached module.
The modules are wired in a daisy-chain configuration. Inter-module communication is realised via serial interfaces (UART). Modules are connected pairwise by one hardware UART and one software UART (Figure 6), with the hardware UART facing in the master's direction. A module first initializes the hardware UART and waits for an incoming connection request. After successfully establishing the connection, it alternately probes the two software UART pins for messages from a successor module, implicitly determining its relative orientation. The established connections are periodically probed by heartbeat messages between adjacent modules to determine breakups and changes in topology. Every change in topology gets propagated upstream to the master, triggering an adoption of the locomotion pattern. A loss of connection to the master module gets propagated to every downstream module as well. Every detached module disconnects completely and waits for a new incoming connection request at the upstream UART. The connection chain always extends downstream, beginning at the master module. Removing and adding modules can be performed at any time during operation, without stopping the currently executed locomotion.
The current version of the Arduino software serial library allows for dynamic pin assignment but only supports half-duplex communication. The interrupt-routines are disabled during a write operation, and incoming data may be lost. To overcome this problem, important messages have to be acknowledged and are retransmitted when lost.

4.2. Communication protocol

The communication protocol consists of two layers: one for internal communication and one for communication between master node and external controller. The instruction set of the internal protocol is a subset of the external ones.

Internal 5-byte messages: 4 bit address, 2 bit flags, 6 bit command type, 12 bit message value and start/end sequence. The address-field is evaluated depending on the messages' direction. In a downstream message (read on the hardware UART), it denotes the receiver. Since upstream messages are always sent to the master, the address field can be used for the sender-address. Operations simultaneously performed on every module can be transmitted as broadcasts.

External External messages are ASCII encoded. The master implements a number of high-level commands, as `run` or `stop` with parameters to modify phase, amplitude etc. To provide full external control, every low-level command is accepted and translated to the internal format by the master.

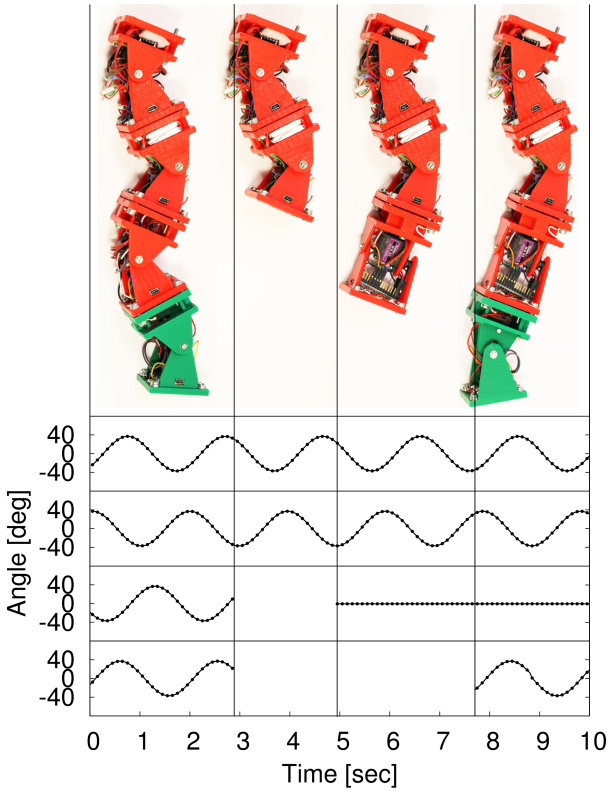


Figure 7 – Recorded joint angles for sinusoidal locomotion generated by the master module during execution on the actual hardware. The plotted values describe the actual joint angles of the four modules. The corresponding topology configuration is depicted by the images above. The lower two modules are detached at $t = 3$ sec. When the third module is reattached in yaw configuration around $t = 5$ sec. it is no longer incorporated into the sinusoidal movement by the master. When the fourth module is reattached in pitch orientation at $t = 7.6$ sec. it starts moving immediately after attaching to participate in the robots movement.

4.3. Performance of the protocol

The serial connections were tested to work reliably at transfer rates up to 19200 baud, while higher rates require improvements to the software serial library. The length of one message is $5 \cdot 8\,\text{bit} + 5 \cdot 2\,\text{bit} = 50\,\text{bit}$ (Start/Stopbit in 8N1-mode). Up to $\frac{19200}{50} = 384\,\text{msg/s}$ are possible in half-duplex transfer. Computing and timing overhead reduces this to about 200 msg/s, measured on a single link between 2 modules. Setting the joint-angles of 8 modules requires 7 messages via the connection between master and slave 1. Given a certain amount of further communication on the bus, e.g. heartbeat messages, this results in a maximum update rate of the travelling wave of $\sim$20 Hz.

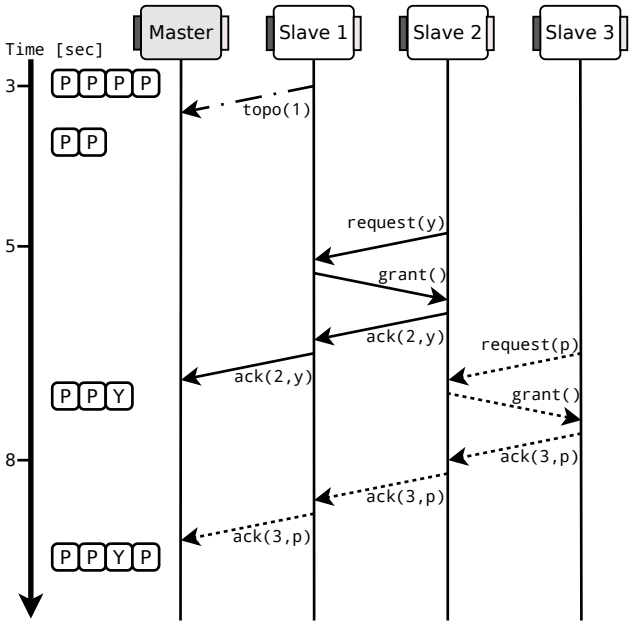


Figure 8 – Sequence of the handshaking protocol to handle the changes in topology as illustrated in Figure 7. The master module stores the current topology. Disconnecting the last two modules in second 3 causes the last remaining slave module, slave 1, to send a topology update to the master. Reattached module slave 2 connects in yaw orientation at second 5 and slave 3 respectively in pitch orientation.

To overcome this limitation, we implemented an experimental mode for decentralized parametric oscillation. In this mode, every module locally generates its own sine wave based on the parameter frequency, amplitude, phase and offset which can be modified by the master during operation to achieve a consistent motion sequence. To prevent drifting, the master node propagates a broadcast `CLOCK_RESET` message every $\frac{1}{freq}$ s downstream, synchronizing the modules to their initial phase once in every full oscillation.

5. Experimental results

To evaluate the functionality of the hardware and software, some basic locomotion algorithms for chainlike modular robots have been implemented. An illustration of a sinusoidal locomotion pattern is shown in Figure 7. The master uses these values to actuate the robot's joints. Changes of the robot's topology are integrated smoothly into the movement. The handshaking-based process that adds modules to the currently stored topology of the robot is illustrated in Figure 8. The incremental procedure starts at the direct neighbour of

the master module and continues with its successors. Acknowledges are passed to the master that stores the change in topology. By taking only several milliseconds the initialization process allows for very fast integration of newly connected modules into the current locomotion pattern. Smooth locomotion patterns are achieved even if further modules are attached during locomotion. Figure 7 illustrates a situation where module 4 connects after the three preceding modules are already initialized and involved in locomotion. The fourth module instantly joins the harmonic locomotion pattern after initialization.

A lateral rolling gait has been implemented as a second common locomotion pattern. A chain of alternating pitch- and yaw-oriented modules is used to create a rolling movement. Experiments with this locomotion pattern pushed the magnetic connectors to their limits, revealing that sudden impacts on hard surfaces can cause short disconnects.

6. Conclusions and future work

This work demonstrates the successful design and creation of a modular robot by using rapid prototyping with FDM printers and standard components. It can be regarded as a case study that shows that the techniques for designing new robots have changed significantly with the increased availability of 3D printers. The presented result is a very flexible, open source and low-cost robot platform that is suitable for research and education in the field of intelligent modular robots. Since every part of the robot is open source, or at least an easy to purchase standard component, it perfectly fits the needs of educational purposes or low-cost research. As a first application of the robotic platform, dynamic detection of the topology has been implemented in hard- and software.

Future work will focus on improving the bandwidth of the upstream bus and the decentralized motion generation in each module to reduce communication overhead. The next steps include extensive testing of various locomotion techniques from literature and integration of different sensors, to enable adaptive locomotion and intelligent behaviour. The use of other microcontrollers with more hardware serial controllers would provide the system with the capability of dealing with more complex topologies.

7. Building a PMR

We explicitly encourage researchers and teachers to copy and enhance our work. The sourcecode for hardware design and operating software is released under a GPLv3 license and can be found on our website `http://tams.informatik.uni-hamburg.de/research/3d-printing/pmr`.

For educational use, besides the actual replication and modification of the hardware, the PMR can potentially be used as a platform for introduction into robotic concepts. The fact that every module comes with its own processor makes it particularly interesting for student projects. To give an inspiration, we developed a simple project scenario:

- Each group gets a single module without operating system and a specification of the communication protocol.
- Each group implements its own protocol against the specification. Local oscillation or integration of sensors as additional tasks.
- Final assembling of the robot together with test of the students designs together.

References

Diankov, R. (2010). *Automated Construction of Robotic Manipulation Programs*, PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

González-Gómez, J. (2014). OpenMR: Modular robots plug-in for openrave, `http://www.iearobotics.com/wiki/index.php?title=OpenMR:_Modular_Robots_plug-in_for_Openrave`. visited: 2014/01/03.

Gonzalez-Gomez, J., Valero-Gomez, A., Prieto-Moreno, A. and Abderrahim, M. (2011). A new open source 3D-printable mobile robotic platform for education, *Proc. of 6th Int. Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2011)*.

Hirose, S. and Yamada, H. (2009). Snake-like robots [tutorial], *Robotics Automation Magazine, IEEE* **16**(1): 88 – 98.

Jones, R., Haufe, P., Sells, E., Iravani, P., Olliver, V., Palmer, C. and Bowyer, A. (2011). RepRap - the replicating rapid prototyper, *Robotica* **29**: 177–191.

Krupke, D., Li, G., Zhang, J., Zhang, H. and Hildre, H. P. (2012). Flexible modular robotic environment for research and education, *26th European Conference on Modelling and Simulation. ECMS2012*, pp. 243–249.

Liedke, J., Matthias, R., Winkler, L. and Worn, H. (2013). The collective self-reconfigurable modular organism (cosmo), *Advanced Intelligent Mechatronics (AIM),*

2013 IEEE/ASME International Conference on, pp. 1–6.

Macdonald, E., Salas, R., Espalin, D., Perez, M., Aguilera, E., Muse, D. and Wicker, R. B. (2014). 3D printing for the rapid prototyping of structural electronics, *IEEE Access* pp. 234–242.

Murata, S. and Kurokawa, H. (2012). *Self-Organizing Robots*, Vol. 77 of *Springer Tracts in Advanced Robotics*, Springer.

Nagy, Z. and Abbott, J. (2007). The magnetic self-aligning hermaphroditic connector a scalable approach for modular microrobots, *Advanced intelligent mechatronics, 2007 IEEE/ASME international conference on*, pp. 1–6.

Smith, R. (2014). Open dynamics engine, `http://www.ode.org`. visited: 2014/01/03.

Yim, M., Duff, D. and Roufas, K. (2000). Polybot: a modular reconfigurable robot, *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, Vol. 1, pp. 514–520 vol.1.

Zhang, H., González-Gómez, J. and Zhang, J. (2009). A new application of modular robots on analysis of caterpillar-like locomotion, *Mechatronics, 2009. ICM 2009. IEEE International Conference on*, pp. 1 –6.

About the authors

Dennis Krupke received his Diploma degree in September 2014 and is now a PhD-student at the University of Hamburg. His research interests are modular robot locomotion control and simulation.
Contact: krupke@informatik.uni-hamburg.de

Florens Wasserfall is an expert in 3D printing technology. He received his master degree in April 2014 and works as a PhD-student at the University of Hamburg.
Contact: wasserfall@informatik.uni-hamburg.de.

Norman Hendrich is a lecturer at the Department of Informatics at the University of Hamburg. His research interests are system simulation and software for autonomous robots, focussing on grasping and manipulation with high-DOF robot systems.

Jianwei Zhang is a full professor and the director of the multi-modal systems group (TAMS) at the Department of informatics at the University of Hamburg. He is a Administration committee member of the IEEE Robotics and Automation Society and the coordinator of several international research projects. His research interests include all aspects of service robotics and cross-modal interactions in natural and artificial cognitive systems, where he has published over 300 peer-reviewed papers.