2014 5th IEEE RAS & EMBS International Conference on
Biomedical Robotics and Biomechatronics (BioRob)
August 12-15, 2014. São Paulo, Brazil

ModGrasp: an Open-Source Rapid-Prototyping Framework for Designing Low-Cost Sensorised Modular Hands

F. Sanfilippo, H. Zhang, K. Y. Pettersen, G. Salvietti and D. Prattichizzo

***Abstract*— This paper introduces *ModGrasp*, an open-source virtual and physical rapid-prototyping framework that allows for the design, simulation and control of low-cost sensorised modular hands. By combining the rapid-prototyping approach with the modular concept, different manipulator configurations can be modelled. A real-time one-to-one correspondence between virtual and physical prototypes is established. Different control algorithms can be implemented for the models.**

By using a low-cost sensing approach, functions for torque sensing at the joint level, sensitive collision detection and joint compliant control are possible. A 3-D visualization environment provides the user with an intuitive visual feedback.

As a case study, a three-fingered modular manipulator is presented. Related simulations are carried out to validate efficiency and flexibility of the proposed rapid-prototyping framework.

I. Introduction

Building a robotic hand with sufficient dexterity has become one of the most attractive steps in getting a robot to fully mimic the movement of the human hand. However, the development of such a hand is challenging because a high number of degrees of freedom (DOFs) is required to be controlled.

A promising solution consists of using a modular approach [1]. Modular grasping makes it possible to use only the necessary number of DOFs to accomplish a specific task. As such, a trade-off between simple grippers and more complex human like manipulators can be reached. Modularity offers robustness to hardware failures considering that the robot parts are interchangeable. The production cost can also be considerably reduced and the weight of the manipulator can be minimised. Moreover, modularity is advantageous in terms of versatility since the robotic hand can be disassembled and reassembled to form new morphologies that are suitable for new tasks.

From a design point of view, rapid-prototyping can be beneficial when developing modular manipulators with different configurations. Development time can be significantly reduced, the main grasp properties can be analysed and the quality can be assessed. Therefore, rapid-prototyping is a necessary step to validate the design before making a physical prototype.

F. Sanfilippo and H. Zhang are with the Department of Maritime Technology and Operations, Aalesund University College, Postboks 1517, 6025 Aalesund, Norway. [fisa, hozh]@hials.no.

K. Y. Pettersen is with the Centre for Autonomous Marine Operations and Systems, Department of Engineering Cybernetics, Norwegian University of Science and Technology, 7491 Trondheim, Norway. kristin.ytterstad.pettersen@itk.ntnu.no.

G. Salvietti and D. Prattichizzo are with the Department of Advanced Robotics, Istituto Italiano di Tecnologia, 16163 Genova, Italy. D. Prattichizzo is also with the Department of Information Engineering, University of Siena, 53100, Siena, Italy. gionata.salvietti@iit.it, prattichizzo@dii.unisi.it.

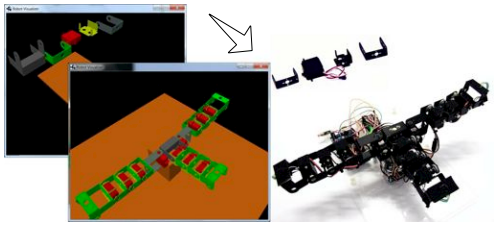


Fig. 1. The idea of realising an integrated virtual and physical rapid-prototyping framework for the design, simulation and control of low-cost sensorised modular hands.

In this paper, *ModGrasp*, a virtual and physical prototyping framework that allows for rapid-prototyping low-cost sensorised modular hands, is presented. The idea is shown in Fig. 1. The rapid-prototyping approach is combined with the modular concept so that different manipulator configurations can be rapidly modelled. This method consists of an immersive design process that involves mechanics, hardware and software. A real-time one-to-one correspondence between virtual and physical prototypes is established. The on-board, low-cost torque sensors provided within each module allow for evaluating the stability of the obtained grasps. An intuitive visual feedback is also provided during the designing phase by means of a 3-D visualisation environment. Moreover, both the virtual models and their physical counterparts can be controlled by using the same input device. As a case study, a three-fingered modular manipulator is presented. Related simulations are carried out to validate efficiency and flexibility of the proposed rapid-prototyping framework.

ModGrasp is an open-source project and it is available on-line at `https://github.com/aauc-mechlab/modgrasp`, along with several detailed class diagrams, all the mechanics, hardware schematics and demo videos.

The paper is organised as follows. In Section II, a review of the related research work is given. In Section III, we focus on the description of the *ModGrasp* architecture, and as a case study, a three-fingered hand is also presented. Related simulations and results are shown in Section IV. In Section V, conclusions and future works are outlined.

II. Related Research Work

Our preliminary studies started by considering the design of modular grasping grippers capable of adapting to different requirements and situations. In [1], an iterative algorithm that allows for finding a trade-off between a simple gripper model

and more complex human like manipulators was presented. However, a rapid-prototyping framework would make it possible to physically test different manipulator models by linking them with their real counterparts.

Robotic systems have been used as part of a rapid prototyping process [2], [3]. However, the application of rapid-prototyping in robot design has been very limited, especially concerning robotic hands. In [4], prototypes of mechanical joints were fabricated experimentally and then used to build the articulated structure of one 4-DOFs finger on a five-fingered robotic hand. In this case, the joints showed good smoothness and evenness in flat vertical and horizontal surfaces. Indeed, joint compliance can enable successful robot grasping despite uncertainties in target object location. Compliance also enhances manipulator robustness by minimising contact forces in the event of unintended contacts or impacts. In [5], the design, fabrication, and evaluation of a novel compliant robotic grasper constructed by using polymer-based shape deposition manufacturing (SDM) was presented. Afterwards, in [6], the same manifacturing technology was used to build a four-fingered, underactuacted manipulator. This gripper was designed with rapid-prototyping methods to provide performance adequate for general-purpose experimentation, while requiring only off the-shelf components and minimal machining. Simple 3-D printed components were used to make the hand compact and lightweight.

However, most of these previous works mainly focus on the mechanical construction process, while hardware, control and software prototyping are often neglected in the prototyping design. To the best of our knowledge, an integrated mechanical, hardware and software rapid-prototyping framework for designing and testing different configurations of modular grasping manipulators is still missing.

III. Framework Architecture and Case Study

A. ModGrasp Architecture

The main focus of this work is on building a flexible rapid-prototyping framework for different modular manipulator models. A generalised manipulator model is considered that consists of one or more chains of identical modules fixed on a base. Referring to a human-like hand, each chain can be considered as a finger, each module as a phalanx and the base as a palm. The fundamental building module is made by a standard micro servo motor and two metal brackets as shown in Fig. 2-a. This elementary module features one DOF and it is simple to construct and assemble, meeting the requirements of versatility, robustness, low-cost and rapid-prototyping. According to the rotation axis of the motor, each module can be connected to another one in a *pitch-pitch* or in a *pitch-yaw* connection configuration, as shown in Fig. 2-b respectively. By assembling these modules, different finger configurations can be built. The concept of modularity is also applied to the base of the manipulator model. In particular, each finger is attached to a common base by means of two special brackets, which make abduction/adduction and flexion/extension movements possible, as shown in Fig. 2-c. The base modules of the fingers can be connected together using their predefined slots and hooks to form a unique base as shown in [1]. Different base configuration can be achieved allowing for describing the most significant grasp models mimicking the human hand taxonomy, which are presented in [7]. Moreover, to finalise the manipulator, a component, which is made by combining two joint brackets, is used for the fingertips, as shown in Fig. 2-d. To increase friction, a small strip of soft material is taped to the fingertips with vulcanised tape. From a mechanical point of view, even though the design may not be the most efficient for grasping, it still meets the requirements of standardisation, modularisation, extendibility and low-cost.

The concept of modularity is also applied to the system architecture on both the software and hardware sides. In particular, as shown in Fig. 3, a master-slave communication pattern is used. Each finger is controlled by a slave controller board, which communicates with a master controller board. Moreover, the controlled manipulators are simulated in a 3-D visualisation environment that communicates with the master controller. Thanks to all these modular properties, the resulting prototypes are extremely robust to hardware failures. For instance, if one or more modules or even one or more entire fingers break or are disassembled from a prototype, the manipulator keeps working with the remaining functioning joints. In the next subsections, the different components of the framework are described.

1) Control Approach: The possibility of implementing certain control features does not influence the design for the proposed prototyping framework. In particular, because of the modularity properties of the framework architecture, the user can implement different control algorithms for the models according to current needs. In this subsection, some possible approaches are briefly outlined to introduce the reader to the case study discussed later in this paper.

For simple prototypes with a low number of DOFs, it may be sufficient to use conventional robotic control design tools and equations [8]. However, when the complexity of the modular model increases in terms of DOFs or when different modular configurations must be controlled independently of their specific morphology, a highly flexible and general control algorithm is needed. A deeper understanding of how the brain exploits the high redundancy of human hands could be an important key in the development of such a control algorithm. In particular, some studies demonstrate that, despite the complexity of the human hand, a few variables are able to account for most of the variance in the patterns of all the possible configurations and movements [9]. These same studies showed that the first two principal components account for most of the variability in the data, more than 80% of the variance in the hand postures. In this context, the principal components were referred to as *synergies* given that, in the *sensorimotor* system of the human hand, combined actions are favoured over individual component actions, with advantages in terms of simplification and efficiency of the overall system. Intuitively, this reduction of DOFs can be used to decrease the complexity of the control algorithm for robotic hands with an anthropomorphic structure that

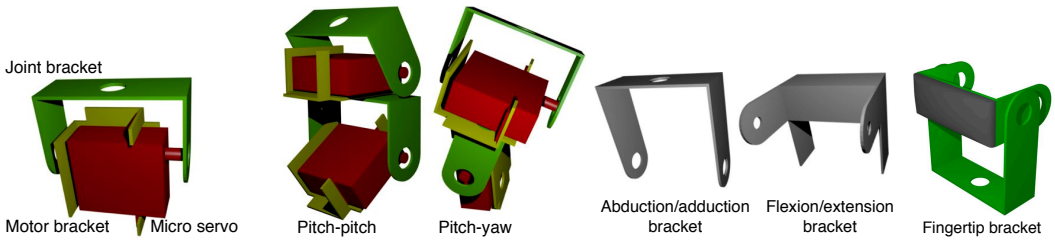


Fig. 2. (a) the fundamental building module that is made by a standard micro servo motor and two metal brackets, (b) the *pitch-pitch* and *pitch-yaw* connection configurations respectively, (c) the two special brackets that allow for abduction/adduction and flexion/extension movements respectively, (d) the component that is used for the fingertips.

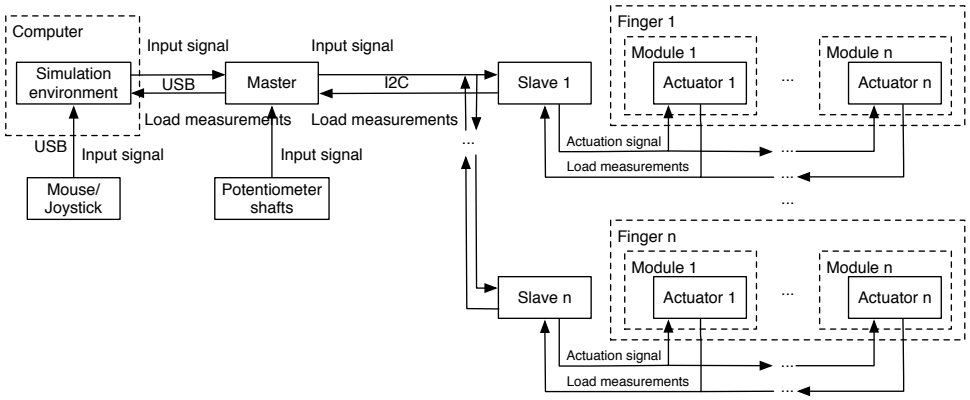


Fig. 3. The proposed control system architecture for *ModGrasp*: a master-slave modular pattern is used for the communication protocol.

closely copies the structure of the human hand. Nonetheless, several approaches for mapping the human hand *synergies* to differently structured robotic hands have been presented [10], [11], showing that this idea is feasible.

The key equations necessary to study manipulators controlled by *synergies* from a kinematic point of view are briefly introduced here. A more detailed presentation of the problem is described in [12]. According to a model inspired by human hand *synergies*, we suppose that the hand is actuated using a number of inputs whose dimension is lower than the number of hand joints. In particular, let the manipulator be described by the joint variable vector $\mathbf{q}_h \in \Re^{n_{q_h}}$, with n_{q_h} representing the number of actuated joints. We assume that the subspace of all configurations can be represented by an input vector of a lower dimension $\mathbf{z} \in \Re^{n_z}$ (with n_z denoting the number of inputs and $n_z \leq n_{q_h}$) which parameterises the motion of the joint variables along the *synergies*. In terms of velocities, one gets:

$$\dot{\mathbf{q}}_h = \mathbf{S}_h \dot{\mathbf{z}}, \tag{1}$$

being $\mathbf{S}_h \in \Re^{n_{q_h} \times n_z}$ the synergy matrix.

2) Controller Boards: On the hardware side, an *Arduino Uno* board [13] based on the *ATmega328* micro-controller is used as the master, while one *Arduino Nano* [13] board is used as a slave to control each finger. *Arduino* is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Using *Arduino* boards simplifies the amount of hardware and software development needed to get a system running. On the software side, *Arduino* provides a number of libraries to make programming the micro-controller easier. The choice of using *Arduino* boards makes the rapid-prototyping framework easy to maintain and makes it possible to add new features in the future.

3) Support for Different Input Devices: The modular manipulators can either be controlled directly from the simulator environment by means of a computer mouse/joystick or they can work stand-alone and be controlled by means of a set of potentiometer shafts that are used as input controllers. In detail, in the first case, a real-time one-to-one correspondence between virtual and physical prototypes is established and the mouse/joystick DOFs are used as inputs. In the second case, a set of potentiometers is connected to the master board and is used to generate the input signal allowing the manipulator to be used without running the simulation environment. In both cases the input signal can be multi-dimensional. Thanks to the modularity of the framework architecture, any other input device can be used without influencing the effectiveness of the system.

4) Communication Protocols: The standard I^2C [14] is used as a communication protocol between the master and the slaves. This protocol is chosen because it is relatively easy to set up and it also supports slaves having different addresses, thereby meeting the requirements of the framework architecture. In addition, the physical manipulator models communicate with the simulation environment through the serial interface of the master controller board.

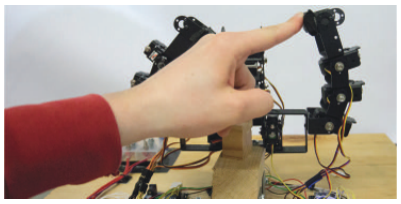


Fig. 4. The sensitive collision detection approach: one of the fingers stops moving when an external force is applied.

```
1: connect to communication network as the master
2: while true do
3:     read inputs
4:     send input values to slaves
5:     receive current loads from slaves
6: end while
```

Algorithm 1: Pseudocode of the program running on the master controller.

```
1: define current threshold, step-back value
2: connect to communication network as a slave
3: while true do
4:     send current values to the master
5:     if input values are received from the master then
6:         for each joint do
7:             calculate actuation value
8:             actuate joint
9:             if current load ≥ current threshold then
10:                calculate step-back value
11:                actuate joint
12:            end if
13:        end for
14:    end if
15: end while
```

Algorithm 2: Pseudocode of the program running on the slave controllers.

5) Low-cost Torque Sensing and Joint Compliance: In order to monitor the load of each joint actuator, the current is continuously measured from each slave controller. This feedback signal is very important in order to improve the manipulator dexterity. In particular, the current sensing at the joints level allows for a more accurate grasping of objects with different stiffness without squeesing or damaging them. By measuring the input current to each servo motor, the servo torque can be calculated and adjusted according to the task to be performed. Moreover, crucial functions like sensitive collision detection and compliant control actions are possible. In Fig. 4, the sensitive collision detection is highlighted: one of the fingers stops moving when an external force is applied.

The basic motor equation is:

$$T = K_t I sin(A), \tag{2}$$

where T is torque, K_t is the motor torque constant, I is the measured current and A is the angle between rotor and stator magnetic fields. In a properly-operating servomotor control scheme, A is held at 90 degrees and I is varied to meet the torque demands. More implementation details on how to measure the servo current by using simple resistors are provided in Section IV. It should be noted that this approach still meets the requirements while being very economical when compared to the use of traditional torque sensors.

6) Logic of the Master and Slave Programs: The master controller takes care of the communication between the simulation environment and the physical prototypes. The program that runs on the master board is very simple and it is shown in the Algorithm 1 box. When the program starts running, it connects to the I^2C network as the master. Then the main iteration loop starts. If the manipulator is directly controlled from the simulator environment by means of a computer mouse/joystick, the master reads the input signal through the serial channel. Alternatively, if the manipulator is used as a stand-alone device, the master reads the input signal from the set of potentiometers that are used as inputs. In both cases, the input signal is forwarded to the slave controllers. At the end of each iteration, the master receives the current loads from each slave.

In accordance with the concept of modularity, each slave controller runs exactly the same program, shown in the Algorithm 2 box. This program stores a current threshold value and a step-back value for the motors. After connecting to the I^2C network as a slave, the program starts the main iteration loop. At each iteration, the current values are read and sent back to the master. Then, every time a new message from the master is received, the new actuation value is calculated for each joint and applied to the motor according to the selected control algorithm. Moreover, if a servo load reaches the predefined threshold, the motor is programmed to step back slightly (according to the predefined step-back value) in order to reduce the torque.

7) Simulation Environment: A basic simulation environment is used to simulate different manipulator prototypes on the computer side for free-hand motions. The virtual models to be controlled are added to the system simply by defining their corresponding standard Denavit-Hartenberg (D-H) tables [15]. In particular, this simulation environment is written in *Java*, thus making cross-platform support possible. To visualise the simulated behaviour, the *Open Graphics Library* (OpenGL) [16] is used on a low level by making raw *OpenGL-API* calls via the *Lightweight Java Game Library* (LWJGL) [17]. Moreover, to make dynamic creation and testing of different models possible without having to recompile the program each time, the simulator core is bound to the powerful, fast, lightweight and embeddable scripting engine *Lua* through the use of the *Kahlua* library [18].

B. Case Study: a Three-Fingered Modular Manipulator

A three-fingered modular hand is considered as a case study, as shown in Fig. 5. The hand consists of one finger having 3 DOFs (the thumb) opposing the two other fingers with 4 DOFs each. Table I is the D-H table of the thumb, while Table II is the D-H table of the other two fingers.

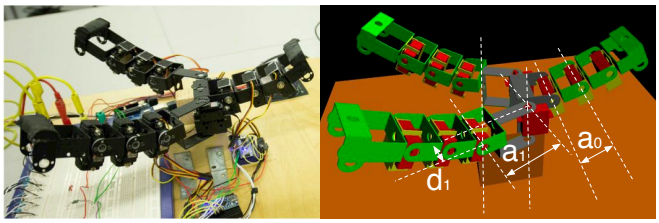


Fig. 5. The three-fingered modular manipulator and the corresponding virtual model. The parameters $a_0 = 3.2cm$, $a_1 = 5.9cm$ and $d_1 = 1.5cm$ are used to determine the D-H tables reported in Table I and Table II.

TABLE I

D-H TABLE OF THE THUMB, WHERE $a_0 = 3.2cm$ IS SHOWN IN FIG. 5

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	a_0	0	θ_1
2	0	a_0	0	θ_2
3	0	a_0	0	θ_3

TABLE II

D-H TABLE OF THE OTHER TWO FINGERS, WHERE $d_1 = 1.5cm$ AND $a_1 = 5.9cm$ ARE SHOWN IN FIG. 5

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
2	$\frac{\pi}{2}$	a_1	0	θ_2
3	0	a_0	0	θ_3
4	0	a_0	0	θ_4

For the sake of simplicity, the fingers are directly attached to a wooden plate that is used as a base. This particular configuration is chosen as a heuristic design in order to describe the most significant grasping models mimicking the human hand taxonomy, as outlined in [7].

Since the hand features 11 DOFs, a *synergistic* control approach is chosen to simplify the control algorithm. The *synergy* matrix, which is determined by following the mapping approach proposed in [11], is:

$$\mathbf{S}_h = \begin{bmatrix} -0.7 & 0 \\ -0.2 & 0 \\ -0.1 & 0 \\ 0 & -1.6 \\ -0.7 & 0 \\ -0.2 & 0 \\ -0.1 & 0 \\ 0 & 1.6 \\ -0.7 & 0 \\ -0.2 & 0 \\ -0.1 & 0 \end{bmatrix} \begin{matrix} \} \, Thumb \\ \\ \\ \} \, Finger1. \\ \\ \\ \\ \} \, Finger2 \\ \\ \\ \end{matrix} \quad (3)$$

In this particular case, an input vector $\mathbf{z} \in \Re^2$ is used to select the first two principal *synergy* components.

It should be noted that thanks to the modularity of *ModGrasp*, any other control algorithm can be implemented without influencing the effectiveness of the system.

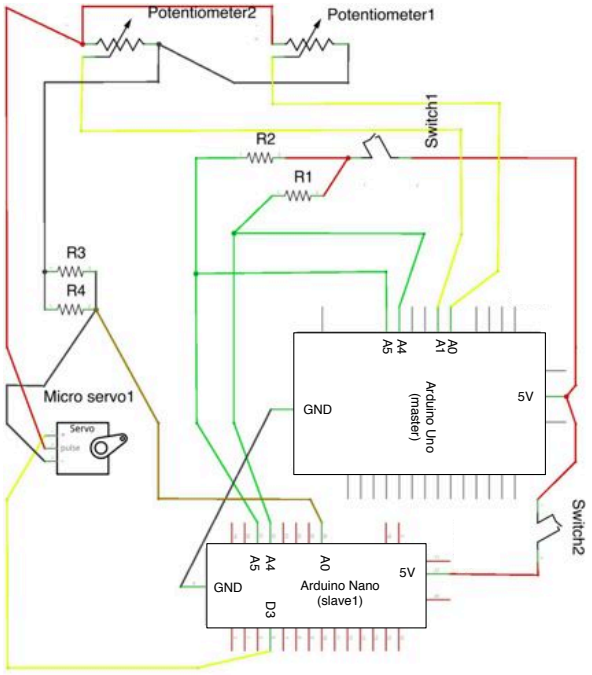


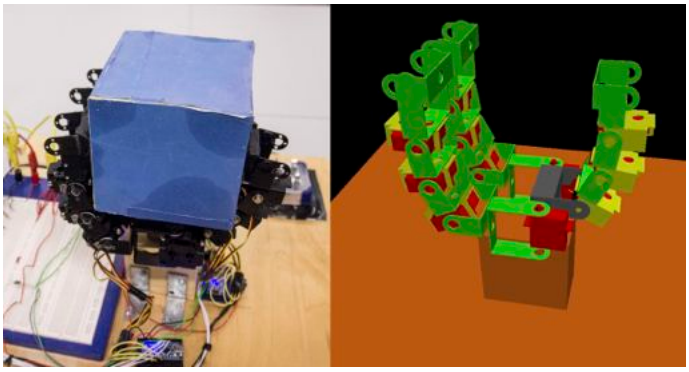
Fig. 6. A section of the wiring schematics for the low-cost sensing circuit and for the communication between master and slave.

IV. SIMULATIONS AND EXPERIMENTAL RESULTS

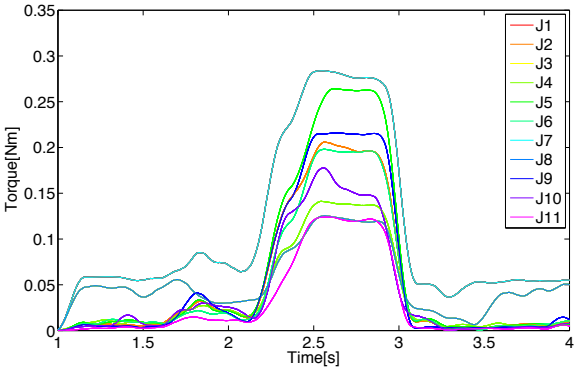
In order to discuss our simulations and experimental results, some implementation details must be provided first. To actuate each module of the manipulator presented in the case study, the *HiTEC HS-85MG* micro servo is used. This particular micro servo features a torque constant $K_t = 0.50Nm/A$ and a maximum torque of $0.294Nm$. A section of the wiring schematics for the hardware is shown in Fig. 6. To measure the current load, two 1Ω resistors connected in parallel are used for each servo. These resistors are connected from the servo ground to the common ground, such that the current will run through that point. Since the electrical resistance is very small, the current can be read as the change in voltage over these resistors. Therefore, wires are connected from these resistors to the analogue input on the respective slave controller board.

To implement the I^2C protocol, two analogue inputs, the A_4 and A_5, are connected together for all the controller boards. Additionally, a $1.5k\Omega$ resistor is placed between A_4 and $5V$ and another is placed between A_5 and $5V$ to ensure that the *Arduino* boards do not interpret noise as an actual high or low value. Two switches are also added in order to make it easy to turn on the power to the master and the slaves.

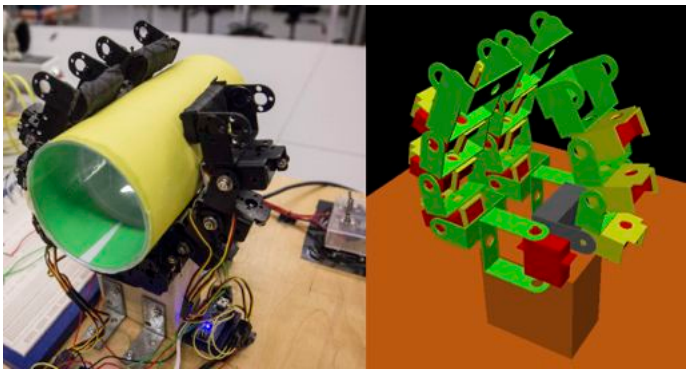
In this case, an I^2C version of the *EasyTransfer* [19] library is used for a fast and easy implementation. The library, besides offering a good support for the I^2C protocol, also provides some extra interfaces and features a wide range of data types that can be easily transferred. Moreover, the *RXTX Java library* [20] is employed to implement the



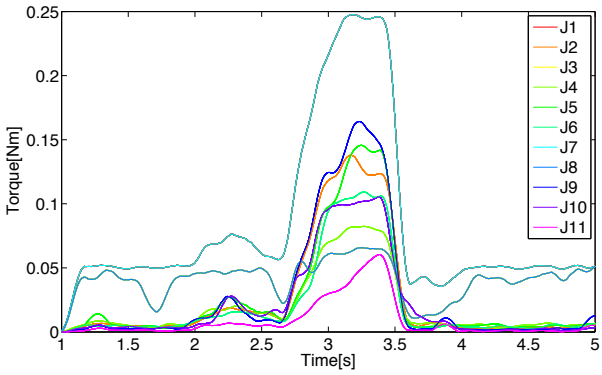
(a)



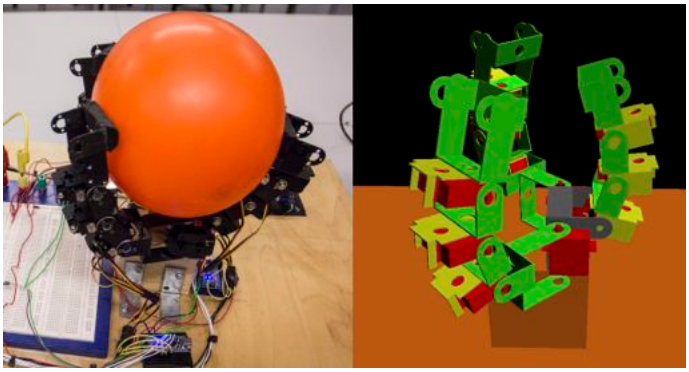
(b)



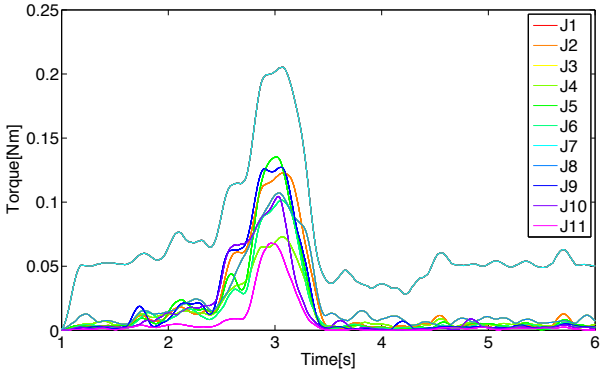
(c)



(d)



(e)



(f)

Fig. 7. Obtained "stable" grasps for (a) a cube, (c) a cylinder and (e) a balloon. Corresponding time plots showing the estimated torque values while grasping and releasing the same set of objects in (b), (d) and (f) respectively.

TABLE III

CURRENT MEASUREMENTS AND CORRESPONDING TORQUES

		Thumb			Finger1				Finger2			
Object	Measurement	$Joint_1$	$Joint_2$	$Joint_3$	$Joint_4$	$Joint_5$	$Joint_6$	$Joint_7$	$Joint_8$	$Joint_9$	$Joint_{10}$	$Joint_{11}$
Cube	Current	0.564	0.405	0.238	0.286	0.523	0.397	0.564	0.238	0.438	0.323	0.245
	Torque	0.282	0.202	0.119	0.143	0.262	0.199	0.282	0.119	0.219	0.161	0.123
Cylinder	Current	0.468	0.249	0.130	0.141	0.193	0.193	0.468	0.130	0.234	0.193	0.082
	Torque	0.234	0.124	0.065	0.071	0.097	0.097	0.234	0.065	0.117	0.097	0.041
Balloon	Current	0.438	0.252	0.212	0.156	0.212	0.208	0.438	0.212	0.252	0.141	0.052
	Torque	0.219	0.126	0.106	0.078	0.106	0.104	0.219	0.106	0.126	0.071	0.026

The current is expressed in A, while the torque unit is Nm.

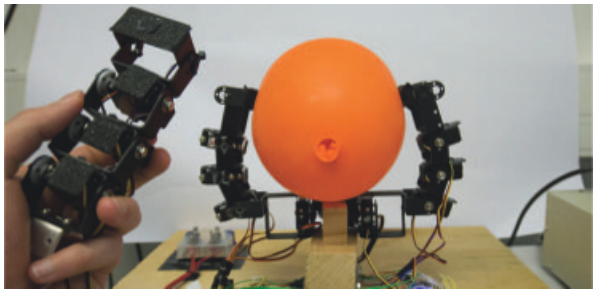


Fig. 8. Possibility of controlling different modular configurations and robustness to hardware failures: the thumb is physically detached and the grasping experiment with the balloon is successfully repeated.

communication link between the simulation environment and the physical manipulator prototypes.

Related simulations are carried out in order to test the framework architecture within the particular case study of the three-fingered modular manipulator. In detail, a set of daily objects which consists of a cube, a cylinder and a balloon, is selected for use in performing some grasp and release experiments, as shown in Fig. 7-a-c-e. In Table III, the corresponding current measurements and torque loads for grasps that a human would consider "stable" are listed. It should be noted that none of the joint actuators are overloaded. Moreover, a time plot of the estimated torque values for the joints while grasping and releasing the same set of objects is shown in Fig. 7-b-d-f. Note that a low-pass filter is applied to reduce the noise from the collected data. As expected, the torque values increase when the contact is made with the object to be grasped. In addition, symmetric patterns in the torque values can be seen between symmetric joints while executing the grasps. Special emphasis should be placed on the balloon grasp because this task is particularly challenging for a robotic hand. With this experiment, our framework demonstrates effectiveness in designing hands that are capable of performing such a kind of task with sufficient dexterity. Finally, to prove the effectiveness of the proposed framework in controlling different manipulator configurations and also to show the robustness to hardware failures of the resulting hands, the thumb is physically detached from the prototype and the grasping experiment with the balloon is successfully repeated as shown in Fig. 8.

V. Conclusion and Future Work

In this paper, *ModGrasp*, an open-source rapid-prototyping framework that makes it possible to design and test low-cost sensorised modular grasping manipulators, is presented. It should be noted that the production cost of a simple model can be easily kept below 100 US dollars given the metal brackets, the micro servo motors, the controller boards and the resistors that are used for sensorising the physical prototypes. In the future, the production cost could be even further reduced by using a 3-D printing approach or a shape deposition manufacturing (SDM) method to produce the mechanical brackets.

This work highlights the potential of the modular grasping approach. It is the authors intend this work to be an open platform for the open-source research community. However, the simulation environment is still in the early stages of development and currently, only free-hand motions are possible. In the future, integration with a physics engine would allow for the simulation of controllable forces, object displacements, manipulability analysis and the addition of other grasp quality measures.

VI. Acknowledgments

The authors gratefully acknowledge the contribution of the students Stian Sandviknes, Ole Jonny Varhaugvik and Andreas Bull Enger in the realisation and implementation of this work.

References

[1] F. Sanfilippo, G. Salvietti, H. Zhang, H. Hildre, and D. Prattichizzo, "Efficient modular grasping: An iterative approach," in *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE, 2012, pp. 1281–1286.

[2] H. Lipson and J. B. Pollack, "Automatic design and manufacture of robotic lifeforms," *Nature*, vol. 406, no. 6799, pp. 974–978, 2000.

[3] G. Reshko, M. T. Mason, and I. R. Nourbakhsh, *Rapid prototyping of small robots*. Carnegie Mellon University, The Robotics Institute, 2002.

[4] J. Won, K. J. DeLaurentis, and C. Mavroidis, "Fabrication of a robotic hand using rapid prototyping," in *Proceedings of the 2000 ASME Mechanisms and Robotics Conference*. Citeseer, 2000, pp. 10–13.

[5] A. M. Dollar and R. D. Howe, "A robust compliant grasper via shape deposition manufacturing," *IEEE/ASME Transactions on Mechatronics*, vol. 11, no. 2, pp. 154–161, 2006.

[6] R. R. Ma, L. U. Odhner, and A. M. Dollar, "A modular, open-source 3d printed underactuated hand," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 2737–2743.

[7] D. Prattichizzo and J. C. Trinkle, "Grasping," *Springer Handbook of Robotics*, pp. 671–700, 2008.

[8] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[9] M. Santello, M. Flanders, and J. F. Soechting, "Postural hand synergies for tool use," *The Journal of Neuroscience*, vol. 18, no. 23, pp. 10 105–10 115, 1998.

[10] G. Gioioso, G. Salvietti, M. Malvezzi, and D. Prattichizzo, "Mapping synergies from human to robotic hands with dissimilar kinematics: an approach in the object domain," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 825–837, August 2013.

[11] M. T. Ciocarlie and P. K. Allen, "Hand posture subspaces for dexterous robotic grasping," *The International Journal of Robotics Research*, vol. 28, no. 7, pp. 851–867, 2009.

[12] D. Prattichizzo, M. Malvezzi, and A. Bicchi, "On motion and force controllability of grasping hands with postural synergies," in *Robotics: Science and Systems*, 2010.

[13] Arduino. (2014, February) Arduino, an open-source electronics prototyping platform. [Online]. Available: http://arduino.cc/

[14] F. Leens, "An introduction to I^2C and *SPI* protocols," *Instrumentation & Measurement Magazine, IEEE*, vol. 12, no. 1, pp. 8–13, 2009.

[15] J. Denavit, "A kinematic notation for lower-pair mechanisms based on matrices," *Trans. of the ASME. Journal of Applied Mechanics*, vol. 22, pp. 215–221, 1955.

[16] OpenGL. (2014, February) The industry's foundation for high performance graphics. [Online]. Available: http://www.opengl.org/

[17] LWJGL. (2014, February) The lightweight java game library (lwjgl). [Online]. Available: http://www.lwjgl.org/

[18] Kahlua. (2014, February) Kahlua, a lua implementation for java. [Online]. Available: https://github.com/krka/kahlua2

[19] B. Porter. (2014, February) Easytransfer arduino library. [Online]. Available: http://www.billporter.info/2011/05/30/easytransfer-arduino-library/

[20] Arduino. (2014, February) Arduino and java. [Online]. Available: http://playground.arduino.cc/Interfacing/Java