J Intell Robot Syst
https://doi.org/10.1007/s10846-017-0727-y

Dimitri: an Open-Source Humanoid Robot with Compliant Joint

Christopher Tatsch[1] · Ahmadreza Ahmadi[2] · Fabrício Bottega[1] · Jun Tani[3] · Rodrigo da Silva Guerra[1]

Received: 19 December 2016 / Accepted: 6 October 2017

Abstract We introduce Dimitri, an open-software & open-hardware humanoid robot with 31 DOFs, fitted with cost-effective modular compliant joints and parallel link legs, designed for advanced human-robot interaction research, force-informed object handling and intelligent environment discovery. Our main innovation is in the design of a robust full-body biped humanoid robot equipped with very low-cost polyurethane torsional spring fixed to traditional servo motors and a circuit to measure angular displacement, transforming the system into a series elastic actuator (SEA). In order to illustrate the robot's qualities in the field of machine learning applied to robotics and manipulation, a multiple timescale recurrent neural network (MTRNN) is implemented, allowing the robot to replicate combined movement sequences earlier taught via interactive demonstration.

Keywords Humanoid robot · Compliant joints · MTRNN · Neural network

✉ Jun Tani
tani1216jp@gmail.com

Rodrigo da Silva Guerra
rodrigo.guerra@ufsm.br

[1] Centro de Tecnologia, Universidade Federal de Santa Maria, Av. Roraima, 1000, Santa Maria, RS, Brazil

[2] Department of Electrical Engineering, KAIST, Daejeon, 305-701, Korea

[3] Cognitive Neurorobotics Research Unit, Okinawa Institute of Science and Technology (OIST) Okinawa, Japan

1 Introduction

This paper introduces Dimitri, an open humanoid robotic platform equipped with inexpensive modular SEAs joints. This article includes and extends the design presented by the same authors during the 2016 Latin American Robotics Symposium [1]. The main new contributions of this manuscript are: (1) the design of the legs, with parallel joint mechanisms for both upper and lower legs with additional pitch actuators to independently move ankle and hip joints, and (2) the more detailed modelling of the springs employed in our proposed SEA actuators.

Conventional robots used in assembly lines are intended to perform decoupled motions following pre-calculated paths, while neglecting perturbations from the environment. Those characteristics are generally accomplished by utilizing extremely rigid joint mechanisms, massive link designs and potent actuators.

Lately, on the other hand, advances in human-robot interaction have added the urge for reliable and compliant force-informed manipulators. Beside being intrinsically safer, these light and flexible mechanisms support the design of intelligent manipulation with coupling constraints and force sensing.

The emergence of various open hardware humanoid robots caused the development of a large community of researchers and enthusiasts who share and collaborate with each other (see [2] for a review).

We aimed to create a tough humanoid robot suited for experimentation on smart compliant handling of objects with torque sensing and secure human-robot interaction. Dimitri's construction is uncomplicated, modular and cost-effective making it easy to maintain and customize. The

Published online: 24 October 2017

SEAs joints allow the handling of manipulation problems that require compliance and torque feedback. Environmental dynamics can be accounted for, by examining the constraints they infringe to the manipulators, sensed through torque feedback. By sensing and controlling the force applied during manipulation, a model of the movement constraints imposed by the environment is developed.

For assessing the robot in the domain of its intelligent manipulation capacity, we decided to perform an evaluation experiment adopting a multiple timescale recurrent neural network (MTRNN) [12], selected thanks to its capability of learning dynamic movement plans, training lower level primitive abilities and arranging these abilities in goal directed policies, at a higher level of abstraction. To show the robot's sturdiness we carried out an evaluation experiment exploring its robustness against abrupt collisions.

The remaining of this article is arranged in this way: Section 2 introduce technical characteristics regarding Dimitri's software and hardware; Section 3 explains the MTRNN experiment and presents its results; and Section 4 shows the conclusion and discussion.

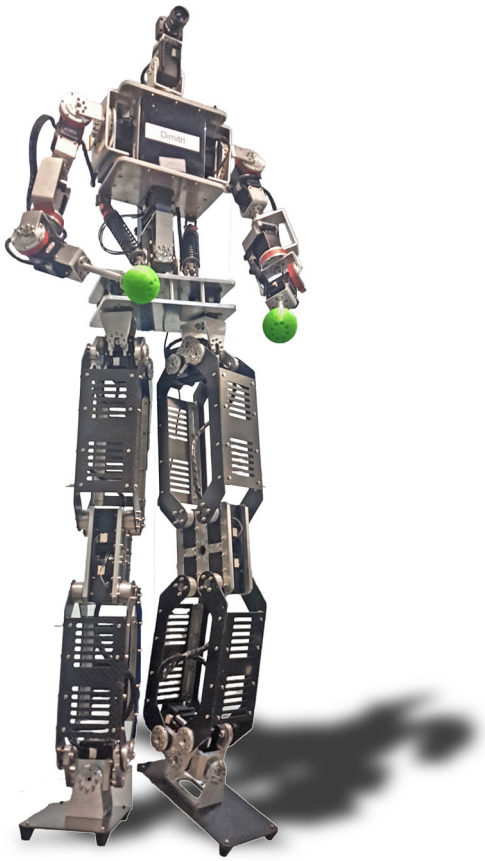

Fig. 1 Dimitri's full body with embedded computer and legs

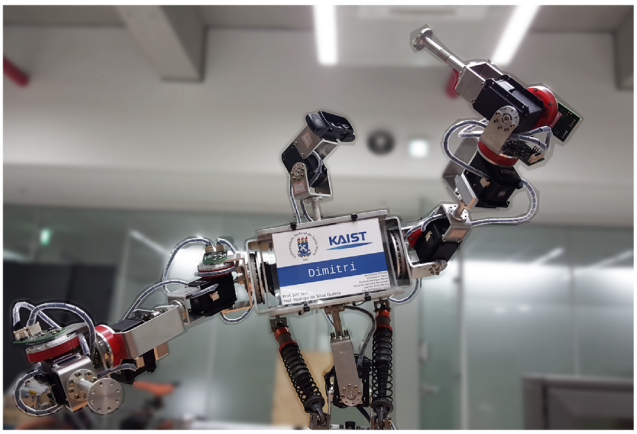

Fig. 2 Upper torso version

2 Methodology

Dimitri was designed with the goal of creating a robust robot that allows studies on fields related to state-of-the-art compliant torque-aware manipulation and human-robot interaction at an affordable cost.

2.1 Robot Details

The humanoid robot presented at Figs. 1 and 2 was conceived to be an open-source and open-hardware, with all its source code, electronic schematics and CAD design of the mechanical parts available on GitHub[1] and other details provided upon request.

The robot's frame is made of carbon fiber and aluminum plates with 3 mm thickness. Aluminum parts were bent using a CNC machine (but other materials and manual bending are also possible). For aesthetic reasons the cylindrical parts used in the arms are milled in a CNC lathe. The arm joints employ series elastic actuators, with torsional springs made of polyurethane milled aluminium fixtures, including an electronic circuit for measuring rotation (the SEA is described in Section 2.2). Table 1 presents Dimitri's main specifications. Figure 3 and Table 2 show the robot's Denavit-Hartenberg parameters.

The popular Dynamixel MX-106R and MX-64R motors, manufactured by Robotis, were chosen for the joints, adopting the first model for legs, arms and waist, and the second model for the neck joint. They are networked together with the SEAs' feedback circuit using a RS-485 bus running the Dynamixel protocol[2]. An embedded NUC computer is used to handle communication with the actuators and SEAs at 1Mbps using a RS-485 to USB adapter. The SEAs support

[1] https://github.com/TauraBots

[2] http://support.robotis.com/en/product/actuator/dynamixel/communication/dxl_packet.htm

Table 1 Robot's technical details

Physical specifications	
Height	1241.79 mm
Reach	542.5 mm
Weight	13.3 kg
DOFs	31
SEA joints	8 (4 in each arm)
Max payload	2.5 kg
Servomotors	MX-64R and MX-106R
Frame material	Aluminium and carbon fiber
Computer	
CPU	Core i5 4th gen. 4250U
Memory	DDR3L 1333MHz 8GB
Storage	SSD mSATA 120GB
Control bus	
Interface	USB 2.0
Channels	4
Bus & protocol	RS-485 Dynamixel 1.0
Baudrate	1Mbps
Camera	
Model	Point Grey FireFly MV
Resolution	1280 × 960
Frame rate	60FPS
Lens mount	CS & C
Energy supply	
Voltage	12.6 Volts DC
Consumption	1960 Watts

torque based modelling and object handling. The motors offer up to 10N.m of stall torque, sufficient to manipulate light items.

The upper and lower legs of Dimitri are both designed with a parallel link mechanism (see Fig. 4). The links were outlined in such way to permit the legs to fold completely, in a crouching position. Furthermore, to enable feet pitch and leg pitch, separate actuators were added to hip and ankle joints. This couples the toughness of the parallel mechanism with the flexibility of full pitch control of legs and feet. Additionally, the modular SEA springs are added to knee joints, allowing studies on flexible and energy efficient walking algorithms.

2.2 Series Elastic Actuator

This humanoid robot's most unique features are its SEA joints. Figure 5 shows how the compliant actuator is assembled[3]. This SEA [9] is essentially composed by classic rigid actuator in series with a torsion spring coupled to the load. This permits the motor to be softly coupled to the load, and the exerted torque to be measured by evaluating the spring's angular displacement. Our SEA design is a torsional spring designed to be attached to the Dynamixel actuators MX-64R and MX-106R, produced by Robotis (for further details see [6] and [7]). This robotics servo actuator was selected due to its good worldwide reputation in the field of robotics, however the same concept can be modified for compatibility with different servo actuators of comparable design. The compliant component is composed of two parts, employing a rotational spring designed using thermoplastic polyurethane (TPU), an elastomer (see Fig. 7). TPU is affordable, resilient, effortless to cut using a CNC router, and it displays rubbery flexibility [3]. This is a very inexpensive architecture since it can be readily fabricated using a simple CNC milling machine. The outline of the spring was conceived so as to support wide angular deformation to both directions, with a linear torque/angle ratio over the full extent of torque provided by its actuators. Figure 6 shows the simulated spring deformation at maximum motor torque.

For estimating spring stiffness, known weights were suspended on a bar of known extent, attached to the output of the spring, and the angular displacement $\Delta\theta$ was evaluated. The torque was calculated for each weight using $\tau = Fl = mgl\cos(\Delta\theta)$. The procedure was reiterated for 20 distinct values of weights and the outcome is displayed in Fig. 8.

The experiment confirmed the spring behaved linearly within the tested range. The coeficients of a line following the equation $y = p_1x + p_2$ were estimated to be $p_1 = 10.49$ and $p_2 = 0$. The slope of this line represents the inverse of the stiffness, so, the value for the torsional stiffness is $k = \frac{1}{10.49} = 0.09Nm/deg$.

In order to evaluate the dynamic behavior of the spring, a step response experiment was performed by releasing a known weight onto the lever and measuring the angular displacement of the compliant element over time. For this experiment, the motor was kept stationary, letting the resulting angular movement to be produced by the compliant element. The step response is shown in Fig. 9.

The system behaves as a spring with a high damping constant. We used *Matlab System Identification Toolbox* and our experimental data set in order to identify the transfer function:

$$G(z) = \frac{8.431z}{z^2 - 0.743z + 0.4229} \tag{1}$$

More details regarding modelling and control of the SEA can be found in our previous work [7].

In addition to offering compliance and torque feedback for intelligent manipulation tasks, the intrinsic flexibility of the SEA joints allow the robot to withstand sudden impacts, making it very tough and safe for human interaction.

[3]The figure shows the joint of a knee, however the same design is used in all arm joints.

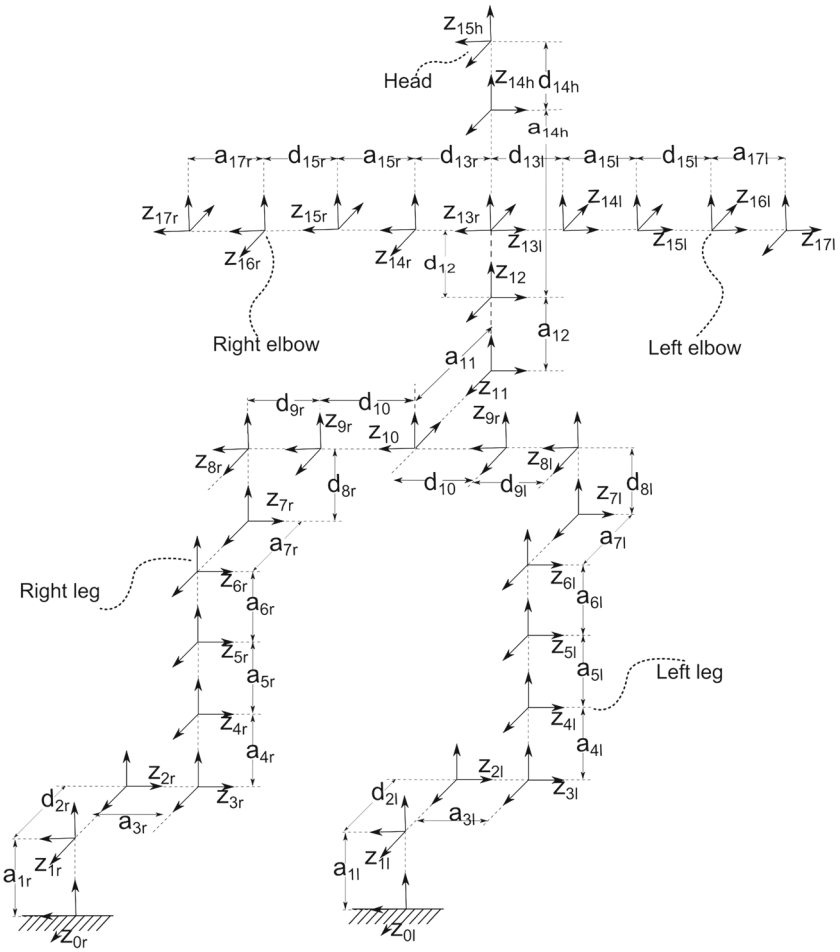

Fig. 3 Denavit-Hartenberg diagram of reference

Table 2 Denavit-Hartenberg parameters right leg and right arm

i	θ_i	d_i(cm)	a_i(cm)	α_i
00	θ_0	0	0	0
01	θ_1	0	4.5	0
02	θ_2	5	0	$\pi/2$
03	$\theta_3 = \theta_4$	0	4	0
04	$\theta_4 = \theta_3$	0	21	0
05	$\theta_5 = \theta_6$	0	10.5	0
06	$\theta_6 = \theta_5$	0	25	0
07	θ_7	0	3.75	0
08	θ_8	3.6	0	$-\pi/2$
09	θ_9	9.6	9.4	$\pi/2$
10	θ_{10}	6	0	$\pi/2$
11	θ_{11}	0	0	$\pi/2$
12	θ_{12}	0	11.5	$-\pi/2$
13	θ_{13}	8	0	$\pi/2$
14	θ_{14}	14.5	0	$\pi/2$
15	θ_{15}	0	18.2	$-\pi/2$
16	θ_{16}	5.5	0	$\pi/2$
17	0	0	9.4	0

2.3 Software Details

The code for controlling Dimitri's regular joints and SEAs is published under the MIT open-source license. There are two versions of the software, one written in C++ and

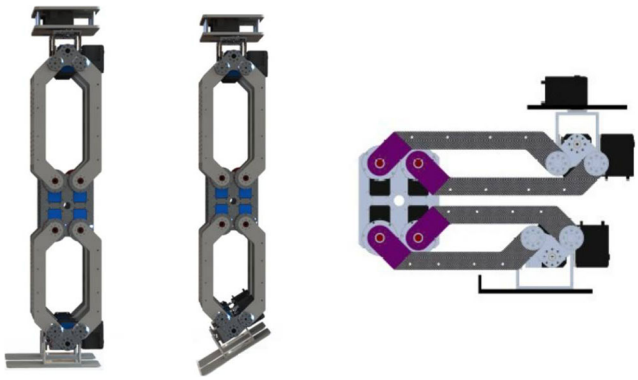

Fig. 4 Left: Legs with parallel link mechanism. Middle: Separate pitch joints for feet and hip allow independent adjustments. Right: Each leg link ends in a 45deg angle, allowing the robot to completely fold its knees, resulting in the sitting position shown in the right

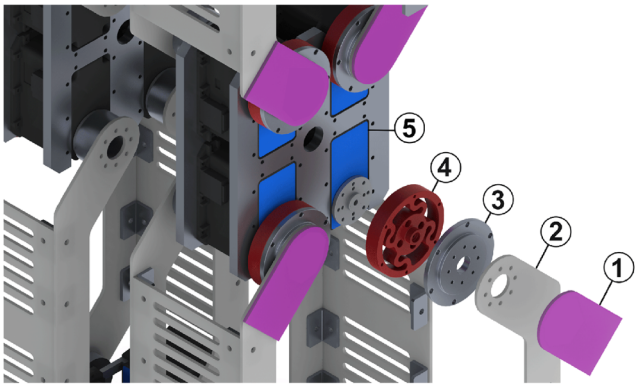

Fig. 5 Knee joint assembly with four SEA, including exploded view. The components are: (1) circuit board, (2) leg link frame, (3) attachment cover, (4) polyurethane torsional spring, (5) Dynamixel MX-106 servo actuator. Illustration adapted from [7]

another in Python, both base codes depending just on standard Linux/Unix system libraries. The exception is for vision processing where OpenCV dependencies are used. Figure 10 shows the class diagram. The class `Joint` handles the interface with the servo actuators. Deriving from it, the `ElasticJoint` adds support to the springs electronic circuit and it applies a PID controller to the actuators. Series of instances of `Joint` are grouped in the class `JointChain` producing the waist, the neck and the arms. The `JointChain` instances are all combined in the class `Dimitri`, representing the whole robot.

3 Experiments and Results

This section reports a MTRNN experiment designed for evaluating Dimitri's potential for cognitive robotics research.

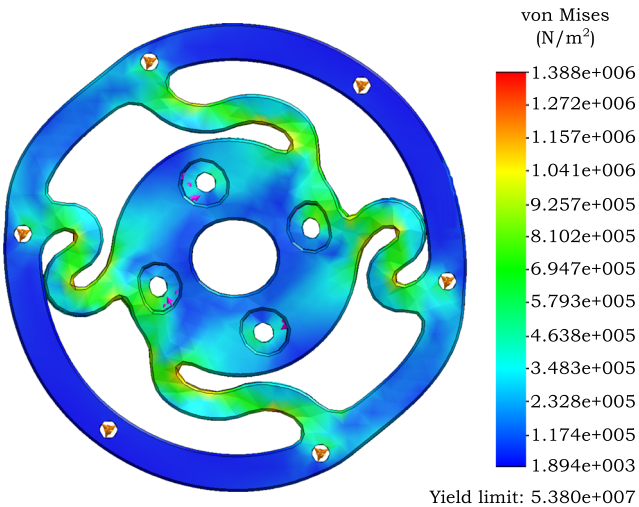

Fig. 6 Finite element analysis of von Mises stress

Fig. 7 Manufactured polyurethane-based torsional spring

3.1 MTRNN Implementation

The multiple timescale recurrent neural network (MTRNN) is a kind of recurrent neural network designed with layers of neurons of distinct time constants, allowing it to self-organize itself in a hierarchy of abstraction levels. Neurons of fast context (FC) have a lower time constant, while neurons of slow context (SC) have a higher time constant. Comparable to the work of [12], input and output layers are exclusively connected to FC units. Every output unit in our model has time constant of 1, without any recurrent connections. Foreign input states are collected by input neurons and their anticipated states are developed by output neurons.

Neuron activations are computed according to a standard firing rate model where the activity of every neuron requires the mean firing rate of other units, and the neuron's decayed inner value depends on the former time step as presented in Eq. 2.

$$u_{i,t+1} = \left(1 - \frac{1}{\tau_i}\right) u_{i,t} + \frac{1}{\tau_i}\left(\sum_j w_{ij} c_{j,t} + \sum_j w_{ik} x_{k,t} + b_i\right) \quad (2)$$

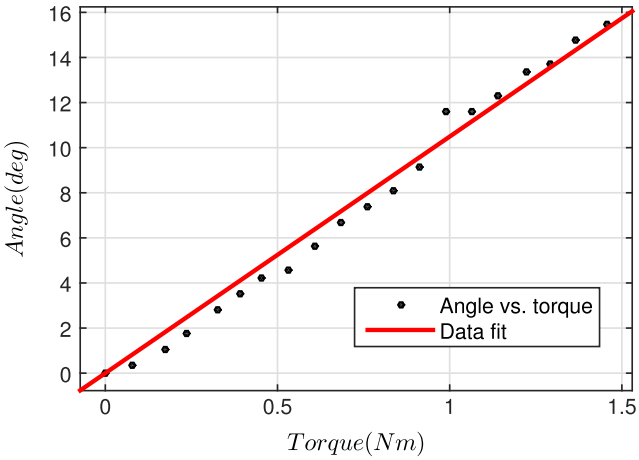

Fig. 8 Estimated stiffness

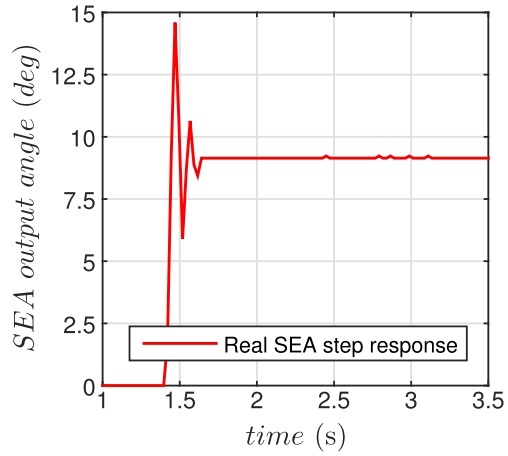

Fig. 9 SEA step response

where $u_{i,t}$ is the i_{th} neuron value at time t, w_{ij} is the connectivity weight from context neurons i_{th} to j_{th}, w_{ik} is the connection weight from i_{th} neuron to k_{th} input, $c_{j,t}$ is the value of the activation of j_{th} context neuron at time t, $x_{k,t}$ is the value of the k_{th} external input unit at instant t, b_i represents the i_{th} neuron's bias, and τ_i is its constant of time. When the neuron is a SC unit, the last summation term disappears, because there aren't any connections to inputs. The weights of the connections between context units (w_{ij}) are bidirectional, and every neuron is connected to one another.

3.1.1 Input and Output Mapping

A softmax transform is applied to map every input $y_{i,t}$ to a space of higher dimension $y_{ij,t}$ conform the adjacent receptive fields of same length. The transformation is described in Eqs. 3 and 4 [4].

$$y_{ij,t} = \frac{\exp\left(\frac{-||k_{ij}-j_{i,t}||^2}{\sigma}\right)}{\sum_{j\in Z}\exp\left(\frac{-||k_{ij}-j_{i,t}||^2}{\sigma}\right)} \tag{3}$$

where k_{ij} represents the the i_{th} dimension of the j_{th} neuron from the actual input, $y_{i,t}$ is the i_{th} dimension of the value of the input before conversion at time t, σ sets the silhouette of the distribution (we used 0.05), and $y_{ij,t}$ is the transformed vector. The following equation calculates vectors of reference

$$k_{ij} = B_i^{Min} + \frac{B_i^{Max} - B_i^{Min}}{l(i)-1}(j-1) \tag{4}$$

where B_i^{Max} and B_i^{Min} represent the maximum and the minimum values for the i_{th} component of the input, respectively, and $l(i)$ represents the dimension of the reference vector.

Each real input was transformed into the higher dimension of 11 after conversion. This was done independently for each input, which in our experiment resulted in 110 softmax input dimensions because 10 real input dimensions were used.

For calculating the i_{th} component of the output unit, we use the inverse softmax transformation as shown in Eq. 5

$$y_{i,t} = \sum_{j\in Z} y_{ij,t}k_{ij} \tag{5}$$

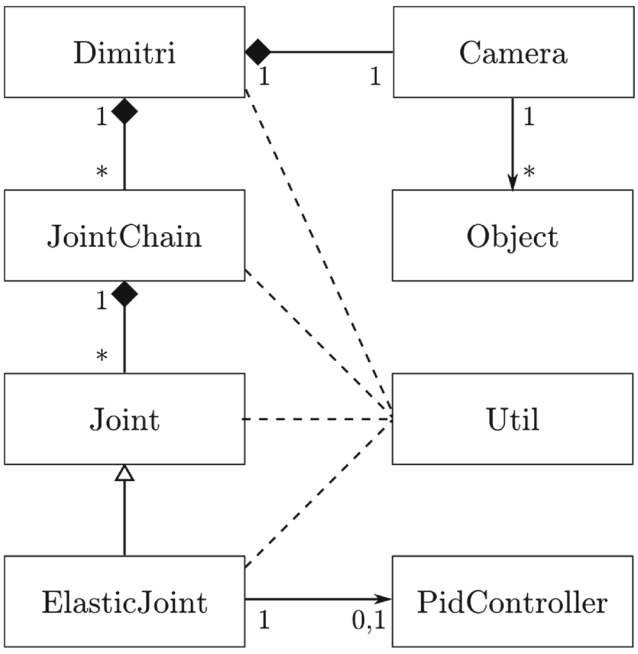

Fig. 10 Simplified UML class diagram

3.1.2 Generation and Training Methods

The context neurons internal dynamics demonstrated in Eq. 2 are achieved according to typical firing rate. In order to compute the forward dynamics context output values, we used the activation function $c_{i,t+1} = 1.7159\tanh(0.667u_{i,t+1})$. Eqs. 6 and 7 compute the output unit's forward dynamics

$$u_{ij,t+1} = \sum_l w_{ijl}c_{l,t} + b_{ij} \tag{6}$$

$$y_{ij,t+1} = \frac{\exp\left(u_{ij,t+1}\right)}{\sum_k \exp(u_{ik,t+1})} \tag{7}$$

where $u_{ij,t+1}$ is the inner state of the j_{th} softmax output matching to the i_{th} output, w_{ijl} is the connection weight from the l_{th} FC neuron to the j_{th} softmax output matching to the i_{th} output, $c_{l,t}$ is the activation of the l_{th} context neuron, b_{ij} is the bias of the j_{th} softmax output matching to the i_{th} output, and $y_{ij,t+1}$ is the j_{th} softmax output matching to the i_{th} output at time $t+1$.

We used an ordinary back-propagation through time (BPTT) training method for the neural network [10]. The learned attributes are updated so as to minimize the Kullbak-

Leibler divergence (noted E) among wanted and real activation values of the softmax output neurons (respectively $\bar{y}_{i,t+1}$ and $y_{i,t+1}$), following the Eq. 8

$$E = \sum_{t} \sum_{i \in O} y_{i,t+1} \log(\frac{\bar{y}_{i,t+1}}{y_{i,t+1}}) \tag{8}$$

Biases, weights and initial states, labelled θ, approach their optimal values following the direction opposite to the gradient $\frac{\partial E}{\partial \theta}$, and are updated according to

$$\nabla\theta(n+1) = \mu\nabla\theta(n) - \alpha\frac{\partial E}{\partial \theta} \tag{9}$$

$$\theta(n+1) = \theta(n) + \nabla\theta(n+1) \tag{10}$$

where α is the learning rate, defined as 0.00003, and μ is the momentum, defined as 0.9. Equations regarding gradients of initial biases, weight and states can be checked in [8].

The MTRNN uses an open-loop generation for training, where the neural network gets the current inputs from the environment and generates several subsequent prediction steps of the output. Initial states, biases and weights are randomized from an uniform distribution on the interval [$-\frac{1}{M}$, $\frac{1}{M}$] at the start of the training, where M represents the count of context neurons in the experiment. The closed-loop generation mode referred as the mental simulation of actions [5, 11] is different because it uses the present predicted output on the next time input. Figure 11a and b are schematics for the open-loop and closed-loop generations respectively, the l is the count of sequential predictions of the output that the MTRNN generates for each presented input.

3.2 MTRNN Experiment

The MTRNN allowed the robot to learn and generate combined movement patterns. Three combined gestures were trained by holding the robot's hand and demonstrating the desired order of movements. A colored cube of dimensions 60×60×60 mm was placed on a table in front of the robot. The object was tracked by adjusting the neck's angles of pitch and yaw – this job was performed using an OpenCV framework. The robot always began to acquire the training data from the same initial pose. During training, the torques of both arms joints (8 DOFs) were set to low values while the waist joints were fixed with a high torque to allow the arms to move easily when a movement pattern was being demonstrated. The two vision inputs we used were the neck pan and tilt angles.

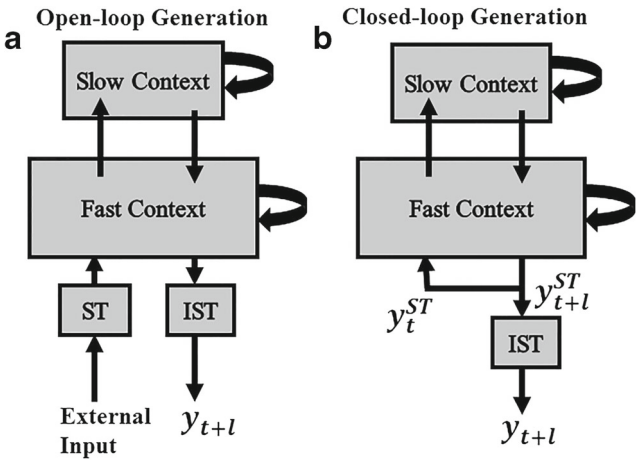

Fig. 11 Plans for (**a**) open-loop and (**b**) closed-loop generation. ST and IST are abbreviations for softmax transform and its inverse, respectively

Three standard movement sequences were determined: (1) Pushing an object using one hand (PUSH); (2) Touching an object with one hand (TOUCH); and (3) Hitting the table taking turns with both hands (HIT). Rather than training each sequence independently, these were trained in combinations of two successive sequences. The combined patterns were:

(A) HP[4] / PUSH / HP / HIT / HP
(B) HP / TOUCH / HP / HIT / HP
(C) HP / TOUCH / HP / PUSH / HP

The three sequences of combined gestures were performed, and their corresponding data series were recorded, always beginning with the colored object on one of three distinct locations in the y coordinate (center; 9 cm up; 9 cm down), adding up to 9 training sequences. Afterwards, we used the the data of the recorded patterns for trainning the MTRNN. The model consisted of 20 SC units, 40 FC units, 110 softmax output units and 110 softmax input units (softmax units were produced from 10 components, 8 representing arm SEAs and 2 representing neck angles, as mentioned in 3.1.1). The constants of time were defined as 5 and 100 for FC and SC units, respectively. Only the initial states of SC units were allowed to be changed during training, whereas the initial states of FC units were kept equal to zero. Initial states of SC neurons had distinct values at every training sequence. This means MTRNN could be trained to generate multiple sequences by correlating initial states with their corresponding sequences. Training ran for 85000 epochs generating 5 forward prediction steps ($l = 5$) of the input series, generated in open-loop.

3.3 MTRNN Results

The MTRNN was used to control Dimitri's arm after the training was successfully finished. The joints of the

[4]HP stands for Home Position

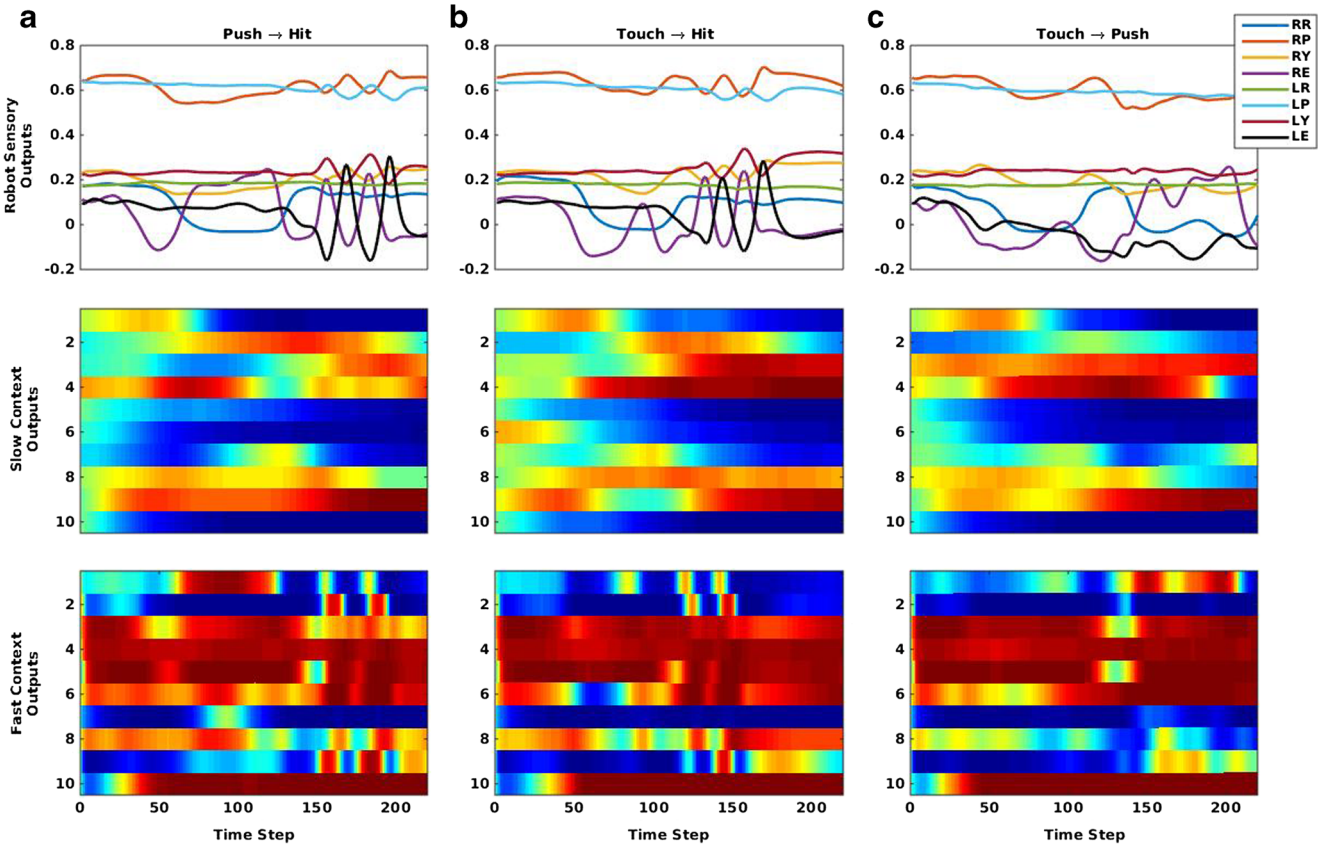

Fig. 12 Outcome of the MTRNN experiment for combined prototypical movement patterns. RR: Right Arm Roll, RP: Right Arm Pitch, RY: Right Arm Yaw, RE: Right Arm Elbow, and likewise for left arm with LR, LP, LY and LE

robot were fixed and the neck angles were assigned to track the object, returning the values as network's external visual inputs. The MTRNN computed the reciprocal arms joint angles carrying out a closed-loop generation. In other words, the system was in a state of semi-closed loop (arms joints were in closed-loop, but vision inputs were in open-loop). The MTRNN was evaluated for objects in 3 different initial positions and vision inputs for all 9 movement schemes. The robot managed to successfully generate all movement schemes. Figure 12a, b and c show the results of a single initial position in the y coordinate. The transitions among primitive sequences can be noted on the graphs of the first row, which represent the dynamics of the angles of the arms joints. The middle row represents the activities of a sample of 10 SC neurons, and the last row represents the activity in the FC units of a sample of 10 neurons. For the sake of clarity, vision inputs (pan and tilt angles) are not shown. A self-organized functional hierarchy can be noticed to emerge, since the frequency differs so that activations in SC neurons showed slower dynamics enabling us to see the broader modulation regulating the commutation among different movement sequences, while activities in FC units had faster dynamics, encoding the specific servo paths for every primitive movement sequence. A mental simulation of the combined models was possible to be performed by setting initial states of the SC neurons, with closed-loop generation, disconnected from any inputs. Dimitri successfully produced all 9 sequences.

3.4 Robustness Experiment

In order to demonstrate Dimitri's robustness, provided by its SEA joint's passive compliance, a brick of 2.57 kg was dropped from a 50 cm over the robot's left arm end-effector, as it can be seen in Fig. 13. The brick's impact on the arm, caused flexion on the springs of the SEA, absorbing the impact. With the feedback of the rotation of the spring, the PID controller managed to command the servo motors to rapidly restore the arm to initial posture.

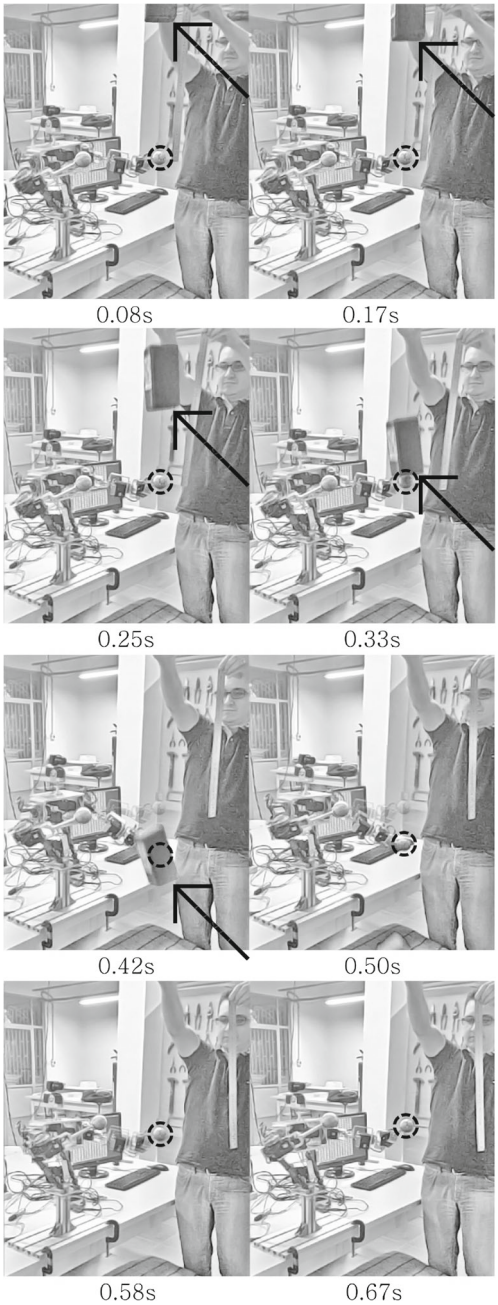

Fig. 13 A brick of 2.57 kg was let go from a 50 cm of height over the robot's left arm, producing no damage. Collision is shown at 0.33s, and the original pose is restored in less than 1s

4 Conclusion

This work introduced Dimitri, a cost-effective, open humanoid robot conceived for studies on human-robot interaction, intelligent force-aware manipulation and other applications of mechanical compliance. Regardless its restrictions, the simplistic arrangement easily demonstrates its advantage when the task demand joint compliance and safe human-robot interaction. The results of the MTRNN experiment confirmed that distinct combined patterns could be trained and generated by using initial states of the SC units both in closed-loop and semi-closed loop. Results also confirm that the intention could be manipulated in an abstract sense, via the superior neural states trickling their effects to the output neurons, in a top-down approach. Research involving online learning and exploration are accomplished safely, and in addition it provides torque response.

The torsional springs performed satisfactorily, allowing torque sensing with manageable decline in joint speed. A noticeable downside is the propensity for oscillatory behaviour frequently observed on the shoulder roll joints. This oscillations were decreased by adjusting PID gains, with the disadvantage of decreasing the system's dynamics.

For subsequent work we are working on increasing Dimitri's capacities, including studies on bipedal walking using the SEA equiped legs and the deisgn of an animatronic face for human social interaction. We are also testing wrist and end-effector designs for more practical service robotics tasks.

References

1. Ahmadi, A., Tatsch, C., Montenegro, F.J.C., Tani, J., Guerra R.D.S.: Dimitri: a low-cost compliant humanoid torso designed for cognitive robotics research. In: Proceedings of the 2016 IEEE Latin American Robotics Symposium (LARS) (2016)
2. Allgeuer, P., Farazi, H., Ficht, G., Schreiber, M., Behnke, S.: The igus humanoid open platform. KI-Künstliche Intelligenz **30**(3–4), 315–319 (2016)
3. Ashby, M., Johnson, K.: Materials and desing: the art and science of material selection in product design. Elsevier Editora Ltda, Rio de Janeiro (2011)
4. Bishop, C.M.: Pattern recognition. Machine Learning (2006)
5. Jeannerod, M.: The representing brain: neural correlates of motor intention and imagery. Behav. Brain Sci. **17**(02), 187–202 (1994)
6. Martins, L.T., Guerra, R.D.S., Tatsch, C., Maciel, E.H., Gerndt, R.: Polyurethane-based modular series elastic upgrade to a robotics actuator. In: Proceedings of the Robot Soccer World Cup XVIII (2015)
7. Martins, L.T., Tatsch, C.A.A., Maciel, E.H., Gerndt, R., Guerra, R.D.S.: A polyurethane-based compliant element for upgrading conventional servos into series elastic actuators. IFAC-PapersOnLine **48**(19), 112–117 (2015)
8. Park, G., Tani, J.: Development of compositional and contextual communicable congruence in robots by using dynamic neural network models. Neural Netw. **72**, 109–122 (2015)
9. Pratt, G., Williamson, M.: Human robot interaction and cooperative robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, pp. 399–406 (1995)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. Tech. rep., DTIC Document (1985)

11. Tani, J.: Model-based learning for mobile robot navigation from the dynamical systems perspective. IEEE Trans. Syst. Man Cybern. B Cybern. **26**(3), 421–436 (1996)
12. Yamashita, Y., Tani, J.: Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. PLoS Comput. Biol. **4**(11), e1000220 (2008)

Christopher Tatsch has a B.S. degree in Electrical Eng. from UFSM and has worked several robotics related in research projects, such as application of neural networks to robotics and the design of the humanoid robot described in this paper.

Ahmadreza Ahmadi graduated in Electrical Engineering at Shiraz University of Technology, Class of 2007. He finished his master degree at University Technology Malaysia in 2012, where he received "Best Student Award" and since 2013 he is a Ph.D. student at the Cognitive Neurobotics Lab at KAIST under the supervision of Prof. Jun Tani. He recently won "Excellent Paper Award" at ICONIP2016.

Fabrício Bottega is a Computer Engineering undergraduate student, working in several robotics projects under the supervision of Prof. Rodrigo da Silva Guerra at UFSM, including university's humanoid robot soccer team. He has recently been involved in founding a start-up company on the field of internet of things, where he and his team have won a two national awards.

Jun Tani received a doctor of engineering degree in electrical engineering from Sophia Univ. in Tokyo in 1995. He worked for Sony Corp. and later for Sony Computer Science Lab as a researcher from 1990 to 2001. Then, he worked at Riken Brain Science Institute from 2001 to 2012 where he has been a PI of Lab. for Behavior and Dynamic Cognition. He became a full professor of Electrical Engineering in Korean Advanced Inst. of Science and Technology in Korea in 2012. Then, he moved to Okinawa Institute of Science and Technology for starting Cognitive Neurorobotics Research Unit in 2017. He is an author of Exploring Robotic Minds: Actions, Symbols, and Consciousness as Self-Organizing Dynamic Phenomena." published from Oxford Univ. Press in 2016.

Rodrigo da Silva Guerra has a B.S. in Control and Automation Eng. (PUCRS 2001) with a M.S. degree in Electrical Eng. (UFRGS 2004), a Ph.D. in Robotics (Osaka Univ. 2008) and a post-doc in the same field (UFRGS 2012). He is currently a professor at UFSM, pursuing projects in fields related to service robotics, involving learning and cooperation among robots and humans. He works on subjects related to robotics, such as computer vision and machine learning, telepresence, robot soccer, educational robotics and humanitarian robotics.