MOTORE++ A Portable Haptic Device for Domestic Rehabilitation

Lucia Saracino[1], Carlo Alberto Avizzano[1], Emanuele Ruffaldi[1], Giovanni Cappiello[2],Zoran Curto[2], Andrea Scoglio[2]

***Abstract*— This paper presents the MOTORE++ system, a Portable, mobile haptic interface with the unique feature of using its wheels to deploy rehabilitation exercises to users. MOTORE++ is a battery powered device, with hot swap batteries, capable to run autonomously for almost an hour.**

MOTORE++ is a technology shift enhancement improvement over the previous version. Its new electronic architecture allows several communication protocols (RS232, WiFi, Bluetooth) that provide link stability in all running scenarios. The device also implements a cutting edge computing architecture and a novel position sensor jointly developed with Anoto®to deliver precise sensor fusion and stability to the localization and force generation algorithms.

In the present papers we will describe the relevant new features of the device, and we will review the major innovation with respect to the existing mobile haptic interfaces.

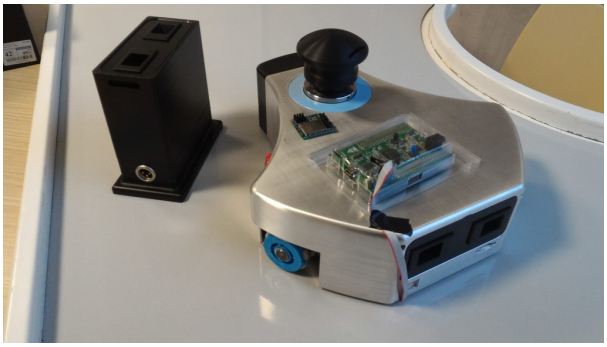

Fig. 1. MOTORE++: The new version of the Portable Haptic Interface. Battery charger with replacement battery (left). The device, with the radio unit and programming board (right). The sliding panel with texturized pattern (bottom).

I. INTRODUCTION

Nowadays robots are widely accepted as a means of rehabilitation tools in several pathologies that may affect the upper limb motor disability. The use of these devices, in combination with Computer Exercises or Virtual Environment, demonstrated higher rehabilitation capabilities with respect to traditional rehabilitation [9]. The most common devices investigated for rehabilitation are robotic exoskeletons(e.g. [4][15]). To date a huge number of exoskeleton like system have been presented by several groups working in the field. In 2007 [5], Brewer presented a systematic review of several existing systems commonly applied to post-stroke therapy. In 2009 Marchal-Crespo [16] provided a systematic review of the control strategies used for the control of these exoskeletons, including assistive motors, impedance and balance control, EMG servos and several different clinical exercises.

Exoskeletons have several benefits, they are wearable, in some cases light, and behave a kinematics that is mostly compliant to human arms. However they also show several drawbacks, among these the weight and the wearing complexity; the power consumption and the autonomy in portable application; the kinematic complexity and the manufacturing costs; the maximum force and the elasticity induced by the serial kinematics; the body attachment when present. As a result, there is no case documented in literature where the use of these systems can be delegated to the patient's family to improve recovery by means of domestic exercises.

The success of MIT-MANUS [8] showed that simpler devices with lesser wearability issues are well accepted as a means of robotic-assisted-rehabilitation into clinics. These devices may provide effective neuro-rehabilitation exercises [13] with a minimal demand for their use.

In 2011 Humanware presented the MOTORE system [3], [2]. MOTORE is a light table top device, fully actuated by three couple of transwheels [21] (a.k.a mecanum or omni-wheels). The wheels are aligned on a circumference with their axes oriented toward the center at 120 degrees in the shape of a standard Killough's platform [18], [12]. MOTORE implements the concept of a mobile haptic interface, as introduced by Nitzsche in [17], but it leverages this concept to a more complex level of interaction thus reducing the typical loss of transparency [7]. Instead than split the motion control and the force generation components, MOTORE uses its wheels for controlling both effects. MOTORE also integrates few technical innovation such as: a precise localization system that exploit the Anoto®technology [10]; a novel sensor fusion policy that optimizes the haptic rendering while minimizing the tracking error; an internal battery pack for up to 40 minutes of autonomous use.

In [19], Mine Sarac et al. presented the AssistOn-Mobile device. The device is a portable rehabilitation system that can be used at home from patients. Similarly to MOTORE, AssistOn uses actuated omni-wheels (four in this case) to drive the device. The use of such wheels allows to achieve fully actuated holonomic motion, while a dedicated suspension system, which is not present in the case of MOTORE, ensures a stable contact with the rolling table. MOTORE does not need any suspension system since there are only three contact points that ensure a stable contact between the wheels and the supporting plane. Such a shape naturally compensates small

This work was supported by the ECHORD++ EU Program

[1] TeCIP Institute Scuola Superiore Sant'Anna, Pisa (ITALY)

[2] Humanware s.r.l, Pisa (ITALY)

planar variation that can exist in the supporting plane. Like MOTORE, AssistOn is highly back-driveable but requires a more complex control policy to manage the redundancy.

Domestic rehabilitation may have several benefits [20]: they save time for both patients and therapists (and money for the society); the reduce the family burden; they increase the exposition to the therapy. However the same investigation highlights present drawback of domestic robotics as mechanical fragility; difficulty of use; cost and capability of the system to constantly stimulate the user.

In this paper we present the MOTORE++ system, an improvement of the previous version which solves some of the existing drawbacks of the first version and introduces new features to the device to make it more suitable for domestic and portable uses. This updated version has several sensing and control innovations that can be applied to other types of mobile-based haptic interfaces.

The remainder of this paper is organized as follows. First (Section II) we summarize the relevant feature of MOTORE. In section III we analyze the drawbacks of the existing solution as requirements for the new robot. In section IV we provide an introduction to the architecture of the novel system. In section V we revise how the software architecture for the new robot has been organized and in section VI we specify the internal architecture of the software. Finally in section VII the results of preliminary tests are discussed.

II. INTRODUCTION TO MOTORE

MOTORE was conceived to deliver rehabilitation exercises as a table-top device. When placed on its desktop panel (1), the device uses its wheels and the friction with the surface to transfer control forces to the hand depending on the motion of the user arm. With a weight of approximately $10kg$, MOTORE was capable to provide forces up to $25N$ in all planar directions.

MOTORE included a set of ten batteries, for a nominal voltage of 13.5 volts which could be recharged between uses. The device was designed to move over a pad 1080x720 mm in size and printed with the ANOTO®[10] pattern. Thanks to an embedded digital camera, the system computes the absolute position of the device in the table. Due to the nature of the sensor it was not possible to send directly this data to the robot MCU but it was instead necessary to stream them via Bluetooth to a PC and then relayed to the MCU of the mobile robot.

A load cell in handle allows to measure the force exchange with the user,while a 2D accelerometer measure the planar acceleration. These sensors, in combination with the motors' encoders, improve the force rendering, inertia compensation and motion control algorithms.

The MOTORE also included on remote (wired) emergency button, that, in combination with the on-board accelerometer, provides consensus information to enable motor control. Also, if the device is not placed onto a horizontal position or the emergency buttons get pressed, the motor drivers are disabled.

The computational core of the device is a 16bit Micro-Controller-Unit (MCU) running at $150MHz$ (TMS320F28335), and providing partial support for floating point with 32bit precision.

At that time of system development, that MCU was selected being the fastest on the market and directly operable by means of Matlab/Simulink®packages. The Bluetooth communication was established using a serial to Bluetooth transceiver and a PC running a bridging software needed to relay the data positioning.

The major features of the system, its control architecture and the related performances were described in [3]. The analysis of the kinematics instead is available in [6] and [14].

III. MOTORE++ REQUIREMENTS

MOTORE has proven to be effectively simple and easy to operate in domestic environments. However during its clinical tests an amount of technical limitations were highlighted.

Bodyshell The shell of the 1^{st} version has several drawbacks. Its center of mass was not well centered in the geometrical center, thus altering the weight distribution onto the wheels and consequently limiting the maximum force; The wheels cannot be easily changed when consumed; the bottom side was completely opened, the battery pack was fixed; the control electronic cannot be easily accessed.

Sensing The position sensing exploited a standard ANOTO®pen which was using the Bluetooth (BT) protocol to forward device position to the control algorithm. This solution presented few limitations. The Pen required a specific initialization sequence to work, and only provided information relating the absolute position (XY) of the pen's tip on the sheet. Hence there was no simple way to derive the orientation information of the device. The battery charge was estimated using solely voltage information, but due to high current consumption of the motor, this voltage was not quite reliable during operation. The h-bridge drivers (ST-VNH5019) incorporated current sensing architecture which was not reliable for low currents (in the range $5-150mA$).

Electronics The computing electronic proved to be efficient but quite limited. The computing MCU had not good standardization using a native 16bit *'char'* ; the ADC conversion was limited by the internal sequencer operation to a maximum of 16 conversions per cycle. There was no possibility for the control architecture to store and change some configuration parameters (such as the accelerometer or the load cell offsets); The programming and debugging operations were extremely slow (up to 15 minutes each trial); the F28335 only had two encoders effectively available for use and the third one was implemented in software; finally we missed additional I/O for debugging operations and storing runtime data.

Communication Bluetooth was used as main channel for all device operation. The power of BT connection resulted extremely low when compared to similar wireless communication systems and the stability (in noisy environments like hospitals and fairs) was not good enough for continuous long term operations. The protocol encapsulated a serial

TABLE I

NEW SENSING ELECTRONICS

Measure	Model Type	Features	Notes
Position	New Anoto Pen	delay $2.5ms$ @ $80Hz$	Includes rotation
Load Cell	LPM7704/7701	gain $13.33V/V$	with electric offset stabilized
Motor Currents(1)	ACS711	gain $110mV/A$, Lin 1% BW $100KHz$	Hall Effect Sensor
Motor Currents(2)	INA199	gain $250mV/A$, Lin 1.5% BW $100KHz$	Shunt Resistor ($5m\Omega$) Sensor
Battery Level	DS2483R		1Wire Batt. monitor

stream to handshake all type of signals: exercise control, pen localization, statistics, etc. The protocol bandwidth in bit-per-seconds (bps) and the Round Trip Time (RTT) of the protocol resulted quite limited for real-time operations (115200 bps with 80mS of RTT). To compensate the missing of orientation information coming form the pen, this sort of operation can be well solved through the use of Kalman filters [1], [11]. In MOTORE, developed a modified Kalman filter capable to reconstruct the optimal position by recovering outdated information using a log of the previous state information. However, in the absence of orientation data, the Kalman filter required several steps before converging to the correct orientation[1].

IV. DESIGN OF MOTORE++

The new design of the robot concerned all the component descried above. First we provided the system with a completely new internal/external shape. The completely circular design was changes with a front/back flat surfaces. Beside improving the storage of the device, this new shape increased the effective workspace available for therapeutic exercises. Two emergency buttons were provided directly on the shell in a position easily reachable by the therapist or the user second hand (is viable in case of hemi-paresis). The wheels can now be easily accessed and replaced by removing three covers, and a the battery pack (now $li-ion$ cells) can be replaced on the fly for continuous rehabilitation operation. A cover sheet on the bottom now protects the internal electronics, and reduce risks of device damage or human accidents during transportation. The weight was slightly increased (now $10.5Kg$) and the improved location of the center of mass allowed for higher forces.

Motore++ comes with a completely new sensing architecture (See Table I). First of all, a new integrated position sensing subsystem, includes a new pen developed with the support of Anoto®. This sensor allowed to directly measure position and orientation of the device on the supporting pad. The data detected by the pen were streamed to the control architecture with a direct serial connection running at $115,200kbps$. Pen position and orientation are measured each $12.5ms$ ($80Hz$) and streamed to the control electronics with an overall communication delay now lesser than $2.5ms$. The position accuracy of the pen was the same of the previous model, while the angle accuracy was limited to $1mrad$ by the communication protocol.

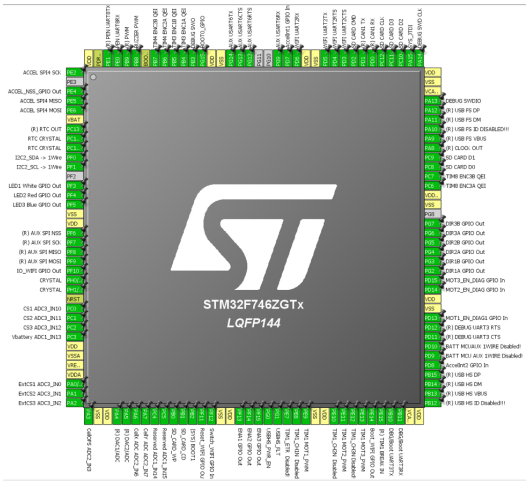

Fig. 2. Pinout of the Motore++ system.

Load cell electronics now have differential input that allow the system to remove the bias induced by electrical changes in the power supply. The input are forwarded to the MCU analog to digital converters (ADC) for being sampled at a $33,600Hz$ (e.g. five time faster than the previous system). Two types of current sensors can now be used on the device in addition to the current sensing mechanism provided by the h-bridges. Both of them proved to be much more accurate than the previous solution. The present implementation uses ACS711 that provide a better linearity all over the current range. Current sensing is sampled at a frequency of $15,900Hz$ then averaged between samples before being processed by the current control loop (running at $5KHz$).

In terms of electronics, MOTORE++ adopts the new cortexM7 processor family recently introduced on the market. This device does allow very high on-board computation capabilities (1082 CoreMark / 462 DMIPS) and internal hardware support to double precision floating point operation.

The selected MCU was an STM32F756 having 144 I/O pin package. The figure 2 shows the present pinout of the system. Almost all the pins have been assigned to specific I/O function. The increased number of pins for this new electronic allowed us to reserve one additional port and two USB ports for debugging operations. In addition we have now full use of SD interface which is employed to record exercise data to a memory card inserted in the backward slot (see figure 1).

[1]The Kalman filter estimated the orientation by comparing the on-board velocity information provided by wheels to the linear motion detected with the pen. During clinical tests this policy demonstrated to be unsafe if for some physiological reasons (e.g. a spasm of the patient), the wheels are dragged onto the pad.

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	SPI3_RX	SPDIFRX_DT	SPI3_RX	SPI2_RX	SPI2_TX	SPI3_TX	SPDIFRX_CS	SPI3_TX
Channel 1	I2C1_RX	I2C3_RX	TIM7_UP		TIM7_UP	I2C1_RX	I2C1_TX	I2C1_TX
Channel 2	TIM4_CH1	-	I2C4 _RX	TIM4_CH2	-	I2C4 _TX	TIM4_UP	TIM4_CH3
Channel 3	-	TIM2_UP TIM2_CH3	I2C3_RX	-	I2C3_TX	TIM2_CH1	TIM2_CH2 TIM2_CH4	TIM2_UP TIM2_CH4
Channel 4	UART5_RX	USART3_RX	UART4_RX	USART3_TX	UART4_TX	USART2_RX	USART2_TX	UART5_TX
Channel 5	UART8_TX	UART7_TX	TIM3_CH4 TIM3_UP	UART7_RX	TIM3_CH1 TIM3_TRIG	TIM3_CH2	UART8_RX	TIM3_CH3
Channel 6	TIM5_CH3 TIM5_UP	TIM5_CH4 TIM5_TRIG	TIM5_CH1	TIM5_CH4 TIM5_TRIG	TIM5_CH2	-	TIM5_UP	-
Channel 7	-	TIM6_UP	I2C2_RX	I2C2_RX	USART3_TX	DAC1	DAC2	I2C2_TX

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	ADC1	SAI1_A	TIM8_CH1 TIM8_CH2 TIM8_CH3	SAI1_A	ADC1	SAI1_B	TIM1_CH1 TIM1_CH2 TIM1_CH3	SAI2_B
Channel 1	-	DCMI	ADC2	ADC2	SAI1_B	SPI6_TX	SPI6_RX	DCMI
Channel 2	ADC3	ADC3	-	SPI5_RX	SPI5_TX	CRYP_OUT	CRYP_IN	HASH_IN
Channel 3	SPI1_RX	-	SPI1_RX	SPI1_TX	SAI2_A	SPI1_TX	SAI2_B	QUADSPI
Channel 4	SPI4_RX	SPI4_TX	USART1_RX	SDMMC1	-	USART1_RX	SDMMC1	USART1_TX
Channel 5	-	USART6_RX	USART6_RX	SPI4_RX	SPI4_TX	-	USART6_TX	USART6_TX
Channel 6	TIM1_TRIG	TIM1_CH1	TIM1_CH2	TIM1_CH1	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	-
Channel 7	-	TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3	SPI6_RX	SPI5_TX	TIM8_CH4 TIM8_TRIG TIM8_COM

Fig. 3. DMA basic map of the Motore++ system. The green color represents the DMA currently programmed; the red one the DMA possibly useful but in conflict with other resources; the yellow highlights available streams currently unused.

We exploited the three independent ADC converters of the device to acquire separately the information related to the current sensing (ADC1), the load cell (ADC3) and the battery voltage (ADC2). PWM signals handled with internal timer hardware were designed at a base frequency that is prime with the ADC conversion thus minimizing the aliasing interference.

Four serials provided to serve for the Anoto®Pen communication (USART8), The remote PC communication (USART2, USART3) and a spare service on-board (USART6). DMA was programmed in order to get almost all these information available without charging the processor load.

As shown in table 3, few DMAs could not be programmed. However these DMAs only related slow peripheral that could be used using polling strategies (as for the USART2 TX port) or the internal sequencer of the ADC2 converter (injected channels).

Finally the internal configuration of the MCU used 9 internal timers out of the 15 effectively available on this type of micro-controller:

- SYSTICK was used for running the HAL kernel (see below);
- TIM1 had enhanced capabilities for driving complex PWM and was used to driver the h-bridges;
- TIM3, TIM4 and TIM8 were respectively used as encoder interfaces;
- TIM6 and TIM7 were used as high speed interrupt generators for driving regular and high speed computational control steps as generated by the Simulink®embedded coder (described below);
- TIM10 and TIM11 were used and reserved for the on-board buzzers.

The communication interface MOTORE++ was completely redesigned while maintaining compatibility with the previous BT interface. We decided to remove the BT interface in favor of a more stable Wi-Fi subsystem. An Internet of Things (IoT) modules (MURATA SN8205) integrating a cortexM3 MCU allowed to implement the MOTORE++ device as a network node which can both connect to extisting framework infrastructure as well as provide an infrastructure by itself. The connection modality for the device is 802.11g running at 54Mbps. Two full-duplex serial ports where mapped onto two TCP or UDP streams (mutually exclusive). The device provide to encode all serial information to and from the wireless connection.

As a safe strategy (in critical environment) the device can also be accessed by means of two USB or serial connections.

V. DEVELOPMENT FRAMEWORK

The original MOTORE firmware was developed using the TI toolkit for Matlab that allowed the Simulink's embedded-coder to design, generate code, compile and deploy software for the assigned target.

When switching to the new architecture we decided to maintain a similar architecture. This choice allows us to preserve most of the complex algorithms for localization, force rendering and trajectory control which were developed during the original MOTORE project [3].

However there are a number of difficulties to solve for this approach:

- ST-Microelectronics provides a Matlab package for the STM32F4xx architecture, but no package currently exist to develop and dignose programs on the novel STM32F7xx;
- Most of the software for the electronic hardware requires specific configuration and control programs to be set on the target. However Matalb does not allow sophisticated mechanisms to integrate its own generated code with user one;
- Interrupt procedure call within the ARM architecture are preemptive only when two interrupts having different priority levels are raised. In the case of a multirate Simulink schematics, with a scheduler triggered with a single timer this behaviour prevents faster loops to execute when a slower loop has not yet completed.

Hence in order to port the pre-existing software on the new architecture we proceeded as follows:

First we collected open tools to compile, flash and debug programs for the target architecture: the gcc-arm-embedded toolkit (available at launchpad.net) provided us a fully cross platform gnu toolchain; as flashing hardware we used an STLinkV2 interface which is commonly available at low cost; the OpenOCD debugger, with minimal script modification allowed us to program and low-level debugging of the target; and the eclipse-cdt toolkit (with add-on for the arm-gcc) allowed us to debug programs at higher level.

Second we edited the target language compilers to allow Matlab deploying programs for any 'gnu like' toolchain. In particular we created a new target file and a new template

makefile that simply use environment variables to control compiling, linking and flashing process.

Third we wrote few Matlab scripts that intercept the load of the schematics and pre-configure required environment variables in order to properly build and flash programs for the selected target.

Finally we configured all the device characteristics using the STMcube32MX (briefly cube in what follows) software from ST microelectronics. Such interface allowed us to configure all I/O assignment, peripheral parameters, and DMA settings from a graphical interface and thus limiting code to be integrated with Simulink. cube also generate code for use with the STM provided libraries (HAL). The use of these libraries, minimizes the amount of code required to manage I/O operation and the risk of unproper memory access consequently to compile optimization (through the use of adequate memory barriers).

On the Wi-Fi module we used the Broadcom WICED toolkit that is available to users adopting hardware with the Broadcom chips on-board (as the MURATA SN8205). Even in this case we implemented few changes in the OpenOCD debugger to allow the device being programmed and debugged through a STLinkV2 SWD interface.

VI. SOFTWARE ARCHITECTURE

The above configuration allowed us to deploy simple configuration and use of the device. In order to render matlab fully interoperable with the cube generated code, we created few inline functions to manage all peripherals and provide very high level diagnostic tools similar to the matlab external mode. The following block were generated:

Timer s-function: the basic timer block can have a Simulink parameter to program the counting period. At each reset, the interrupt procedure provides to generate a Simulink function call.

Scheduler s-function: this is the most important block. The block provides to sync the Simulink scheme with a timer. Whenever it receives the function call the Simulink basic step loop is called (rtOneStep). Additionally it provides to properly call initialization procedure for cube and Simulink in the proper order. The scheduler block is also responsible to implement the Reentrant Interrupt Procedure Call (RIPC) as preemption method.

Digital IO s-function: these blocks provide to convert Simulink logical/integer signals into electrical values on the digital IO ports (and viceversa).

Encoder s-function: this block provides I/O access to the timers programmed as encoders interface. All the configuration of the timer should be managed in cube, and only the counter value will be available as block output.

PWM s-function: this block provides control of the timer generating PWM waves for the control of the three h-bridges. It takes up to four values from Simulink and programs them in the timer compare registers. The remaining configuration operation are performed in cube. PWM is also used with a variable period in order to control the tune of the buzzer.

ADC two types of ADC s-function were generated. The first type uses DMA to program conversion transfer into an array of shared memory which is provided as output; the second provides specific outputs as direct readings of the ADC injected channels. Both types of s-function do not request any particular computational load to the MCU.

USART block manages to handle serial communication on any USART device. It constructs IO buffers in memory, programs DMA, and regularly use send and receive commands to exchange data with a remote node. The type and the size of data is dynamically determined at time of build. Optionally to render the communication more robust, an header, a tail and a CRC32 checksum can be added to the protocol.

IDLE s-function: implement a vector of function call blocks which are executed at best effort only when the realtime operation are completed. It concept is similar to non-realtime tasks versus realtime. Using such block we implement different new features on the MCU such as: a console manager that listen on one serial (TCP/UDP) port and provides a shell to control the operation of the MOTORE++ device; a diagnostic tool that provides realtime monitoring of all internal parameters/viarables; a flash tool that provides an interactive configuration API to manage the device specific parameters (e.g. load cell/accelerometer offset).

USB It is similar to the USART block but handles USB communication on the FULL speed or HIGH speed interface of the chip.

A. CONTROL LOOPS

There are two major control loops in the device: and internal current controller which runs at 5KHz and the force position controller which runs at 1KHz. The two loops are triggered by different timers (TIM6 and TIM7).

The current controller loop, reads and averages the currents stored by DMA at high frequency, and uses the estimated value to regulate the appropriate PWM duty cycle to the h-bridges. The controller structure includes two elements: a feedforward controller whose value is determined on the ideal switching model; and a feedback controller (a PI regulator) that uses the sensors' values to cancel the current error.

The force position controller is composed of different components: first an extended Kalman filter (EKF) is employed to reconstruct the robot location combining PEN and encoder readings; second a velocity controller is used as an internal loop for the haptic impedance/admittance controller. In the same loop two feedforward component allow to improve the controller performance: an inertia compensator and a handle torque compensator which compensate for the location where the user force is applied. The inertia compensator uses the accelerometer input and the force compensator uses the force information provided by the load cell.

At slower frequencies we have other loops: 500Hz the USB handler; 100Hz the audio controller which manages the buzzer; 2-10Hz the battery.

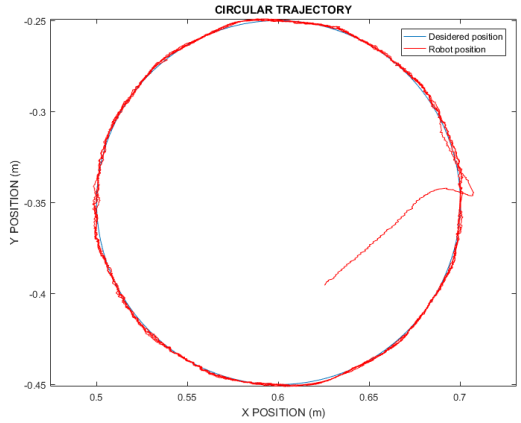

Fig. 4. Trajectory tracking performed by Motore++ system. The maximum error is 0.0029 m, calculated as the difference between the maximum radius obtained and the desidered radius.

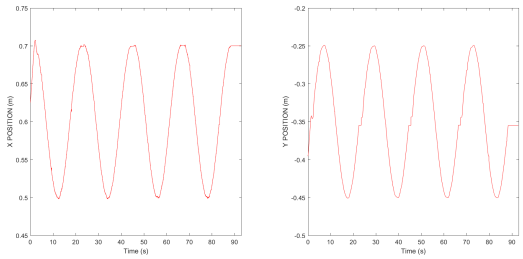

Fig. 5. Temporal evolution of individual XY axes.

VII. Experimental results

Preliminar experiments has been carried out with the aim of testing and characterize the robot performance in terms of path following, load on the system and relations between forces excerted, velocity and position.

In the following analisys we present the robot execution of the tracking of a circular trajectory with radius of 10cm. The first image in Figure **??** shows the performed trajectory (red lines) overlapped on the reference path (blue line) and the second image points out the individual XY axes temporal evolution. The robot moves in a completely autonomous mode, so all the control is focused on the force excerted by the device itself. The control equations used:

$$F = M_a a \tag{1}$$

$$F = k(Pd - P) \tag{2}$$

Equation(1) is the base of the mechanism of impedance control, where F is the force M_a is the apparent mass and a is the acceleration. Equation(2) refers instead to the relation that generates the force, with stiffness gain k and direction of motion given by the difference between the desidered position Pd and actual position P in the 2D space. The stiffness has been set to 150 N/m to obtain a smooth and precise tracking. The results denote a position error always under 3 mm and an optimal repeatability of the path.

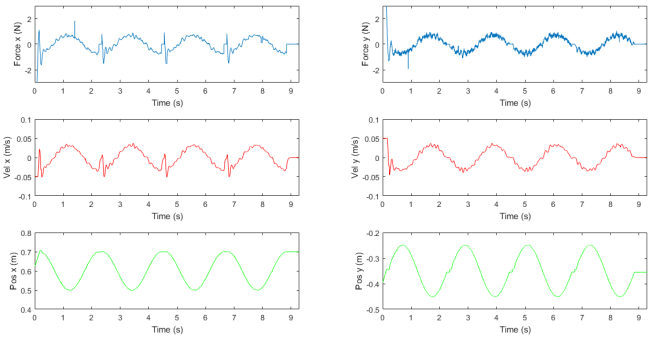

Fig. 6. Comparison between force excerted,velocity commanded and actual position while performing circular trajectory.

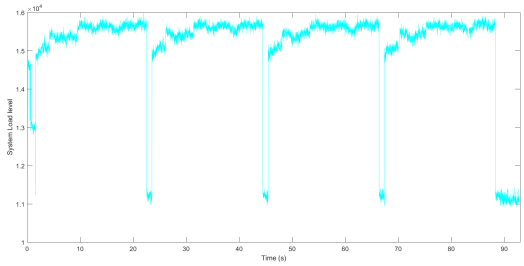

Fig. 7. Load Level of the Motore++ system. Target Timer Count threshold for the scheduler : 49999.

In order to inspect the performance of the inner velocity control loop, during the execution of the tracking we have registered and plotted data relative to force and velocity commanded in addition to actual robot position (see figure 6). As the results suggest the force command is properly integrated and scaled to reach a velocity command, considering a viscosity component.

Moreover to evaluate the persistence and the efficience of the scheduler gesture we have analyzed the system load level in figure 1, reached during the circular path following. A Reentrant Interrupt Procedures (RIP) is implemented on the device and Target Timer Count (PV) is 49999. As clear from the results the values of the free time is keept hugely under the threshold PV, ensuring a smooth behaviour of the robot for all the task.

VIII. Conclusion

In this paper we presented the MOTORE++ system, a new generation of electronic/software system that introduces novelties both at mechatronics and computing levels. MOTORE++ provided to be more efficient, performing and robust of previously existing device; the novel sensing subsystem (position, currents and force) allowed more stable control loops even in presence of external disturbances.

The ergonomy of the device has been optimized for clinical and domestic use, its shape optimized for transportation and its power scheme optimized for continuous use.

The device is currently under clinical validation tests and will be shortly distributed to a larger user group.

ACKNOWLEDGMENT

The authors wish to thank Humanware s.r.l. for the mechanical design and Robotech s.r.l. for the electronic implementation. This work has been funded by EU project ECHORD++. The work on the *software architecture* section has been contributed by the EU FP7 Project ReMeDi contract 610902 and by internal funding of Scuola Superiore Sant'Anna.

REFERENCES

[1] L. Armesto and J. Tornero. Slam based on kalman filter for multi-rate fusion of laser and encoder measurements. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1860–1865. IEEE, 2005.

[2] C. Avizzano, M. Satler, and E. Ruffaldi. Portable haptic interface with omni-directional movement and force capability. *IEEE Transactions on Haptics*, 7(2):110–120, April 2014.

[3] C. A. Avizzano, M. Satler, G. Cappiello, A. Scoglio, E. Ruffaldi, and M. Bergamasco. Motore: A mobile haptic interface for neuro-rehabilitation. In *Robot and Human Interactive Communication, RO-MAN. The 20th IEEE International Symposium on*, pages 383–388. IEEE, 2011. Cite this for citing the MOTORE haptic device.

[4] M. Bergamasco, A. Frisoli, and C. Avizzano. Exoskeletons as man-machine interface systems for teleoperation and interaction in virtual environments. *Advances in Telerobotics*, pages 61–76, 2007.

[5] B. R. Brewer, S. K. McDowell, and L. C. Worthen-Chaudhari. Post-stroke upper extremity rehabilitation: a review of robotic systems and clinical results. *Topics in stroke rehabilitation*, 14(6):22–44, 2007.

[6] B. Carter, M. Good, M. Dorohoff, J. Lew, and R. L. W. Ii. Mechanical design and modeling of an omni-directional robocup player. In *in Proceedings RoboCup 2001 International Symposium*, 2001.

[7] A. Formaglio, A. Giannitrapani, M. Franzini, D. Prattichizzo, and F. Barbagli. Performance of mobile haptic interfaces. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 8343–8348. IEEE, 2006.

[8] N. Hogan, H. Krebs, J. Charnnarong, P. Srikrishna, and A. Sharon. Mit-manus: a workstation for manual therapy and training. i. In *Robot and Human Communication, 1992. Proceedings., IEEE International Workshop on*, pages 161–165. IEEE, 2002.

[9] M. K. Holden. Virtual environments for motor rehabilitation: review. *Cyberpsychology & behavior*, 8(3):187–211, 2005.

[10] http://www.anoto.com/.

[11] S. Julier and J. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2005.

[12] S. M. Killough and F. G. Pin. Design of an omnidirectional and holonomic wheeled platform prototype. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 84–90. IEEE, 1992.

[13] H. Krebs, M. Ferraro, S. Buerger, M. Newbery, A. Makiyama, M. Sandmann, D. Lynch, B. Volpe, and N. Hogan. Rehabilitation robotics: pilot trial of a spatial extension for mit-manus. *Journal of NeuroEngineering and Rehabilitation*, 1(1):5, 2004.

[14] Y. Liu, R. Williams, and J. Zhu. Integrated control and navigation for omni-directional mobile robot based on trajectory linearization. In *American Control Conference, 2007. ACC'07*, pages 2153–2158. IEEE, 2007.

[15] L. Lugo-Villeda, A. Frisoli, O. Sandoval-Gonzalez, M. Padilla, V. Parra-Vega, C. Avizzano, E. Ruffaldi, and M. Bergamasco. Haptic guidance of light-exoskeleton for arm-rehabilitation tasks. In *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pages 903–908. IEEE, 2009.

[16] L. Marchal-Crespo and D. J. Reinkensmeyer. Review of control strategies for robotic movement training after neurologic injury. *Journal of neuroengineering and rehabilitation*, 6(1):20, 2009.

[17] N. Nitzsche, U. Hanebeck, and G. Schmidt. Design issues of mobile haptic interfaces. *Journal of Robotic Systems*, 20(9), 2003.

[18] F. Pin and S. Killough. A new family of omnidirectional and holonomic wheeled platforms for mobile robots. *Robotics and Automation, IEEE Transactions on*, 10(4):480–489, 2002.

[19] M. Sarac, M. A. Ergin, A. Erdogan, and V. Patoglu. Assiston-mobile: A series elastic holonomic mobile platform for upper extremity rehabilitation. *Robotica*, 32(08):1433–1459, 2014.

[20] M. Scopelliti, M. V. Giuliani, and F. Fornara. Robots in a domestic setting: a psychological approach. *Universal Access in the Information Society*, 4(2):146–155, 2005.

[21] R. Williams, B. Carter, P. Gallina, and G. Rosati. Dynamic model with slip for wheeled omnidirectional robots. *IEEE transactions on Robotics and Automation*, 18(3):285–293, 2002.