Online reconfiguration of a variable-stiffness actuator

Rene Jimenez-Fabian, Maarten Weckx, David Rodriguez-Cianca, Dirk Lefeber, Member, IEEE, and Bram Vanderborght, Senior Member, IEEE

Abstract—This manuscript presents an algorithm to adjust the configuration of a variable-stiffness actuator to reduce its electrical energy requirements during the execution of repetitive tasks. The algorithm is based on the gradient descent optimization method with a modification of the adaptation step size, a forgetting term, and a projection rule to cope with the variation of the actual objective function and signal noise. The performance of the algorithm is validated with experimental results that confirm its capability to reduce the energy requirements of the actuator’s driving mechanism. Simulation results illustrate the use of the algorithm in a two-degree-of-freedom system.

Index Terms—Adaptive systems, actuators, energy storage, optimization methods.

I. INTRODUCTION

ACTUATOR technology is under constant development to improve the way in which machines operate [1]. The general tendency towards portable devices with a higher level of autonomy and better power-to-mass ratio have stimulated the creation of new actuation concepts to improve the use of energy sources.

Some of the recent actuation principles are based on the combination of traditional actuators (e.g., electric and pneumatic) with elastic elements to introduce new capabilities as improved shock absorption [2] and lower motor power requirements [3]. An attractive feature of this actuation paradigm is the ability of reusing part of the interaction energy to produce positive work. In particular, variable-stiffness actuators (VSAs) [4] offer the possibility of adjusting the way in which this energy is stored and released by changing their output stiffness.

Concerning the utilization of VSAs as means of reducing the actuation energy requirements, the majority of the proposed control schemes are based on mathematical models of the actuated systems. Among these model-based schemes there is the work of Vanderborght et al. [5] in which the effect of the stiffness modification of an antagonistic pneumatic actuator is evaluated. The proposed scheme is able to reduce the energy consumption of the actuator by matching its compliance to that of the desired trajectory. An extension of this work [6] suggests the use of the average slope of the torque-angle characteristics of the desired trajectory as an approximation of the optimal stiffness. Both approaches, however, do not take into account other intrinsic energy-storage mechanisms that may play a relevant role in the system dynamics (e.g., inertia).

In [7] another model-based strategy is proposed to optimize the time-varying torque and stiffness profiles in explosive motion by making a better use of the energy-storage capabilities of the actuator. Özparpucu et al. [8] developed optimal control strategies to change the elastic properties of VSAs and used them to maximize the final link velocity. Roozing et al. [9] suggested a control strategy for an asymmetric antagonistic actuator in which the low- and high-frequency components of the desired torque are generated by a low- and a high-stiffness power branch, respectively. The work in [10] presents a general classification of VSAs stiffness-adjustment mechanisms and provides some theoretical criteria to evaluate their performance from the perspective of the energy manipulation.

The actual performance of model-based control strategies, as those previously mentioned, may be affected by the variation of the system parameters and unmodeled dynamics [11]. Inertia, motor and transmission efficiency, nonlinear phenomena, motor controller dynamics, and other factors that play an important role in the energy manipulation of the actuator are often neglected to facilitate theoretical analyses. Although some techniques have been proposed to improve the estimation of unknown mechanical signals and parameters in VSAs (e.g., [12], [13], [14]), their use in the optimization of the actual electrical energy consumption of the actuator is in many cases impeded by the lack of a proper mapping from such mechanical quantities to the actual electrical energy requirements of the device.

Recursive optimization [15] and learning control strategies [16] offer a different perspective in which the adaptation is performed with a less accurate knowledge of the system dynamics. These techniques have also been applied to estimate the optimal stiffness configuration in some robotic applications aimed at improving the energy manipulation of the actuated systems. For example, Uemura et al. [17] proposed an online parameter-adaptation law to adjust the actuator stiffness for a torque-requirement minimization. Koller et al. [18] applied an online optimization scheme, based on the work of Felt et al. [19], to a powered ankle device to modulate the provided assistance using the metabolic effort as the objective function. Nasiri et al. [20] developed an online stiffness adaptation rule to reduce the mechanical energy consumption by minimizing the required actuator force using parallel-elastic elements.

In this paper we present a recursive algorithm to adjust the configuration of a VSA based on the actual profile of its input electrical energy requirements during the execution
of repetitive tasks. In contrast to some of the previously referred algorithms that focus only on the optimization of the mechanical output, the proposed algorithm can directly tackle the reduction of the input energy requirements without the need of sophisticated models of mechanical transmissions and electrical motors. The term “reconfiguration” in the title of this manuscript is used instead of other terms like “stiffness control” or “stiffness optimization” to emphasize the fact that the algorithm is not explicitly designed to track a stiffness reference or to minimize an already-known function of this parameter.

The development of the algorithm was motivated by a need for an optimal configuration for a lower-limb prosthetic joint powered by a VSA [21], [22]. Nevertheless, the algorithm in its current form could also be used in other applications where the tasks are repetitive with gradual changes in the optimal settings. Examples of these applications are exoskeletons, manipulators with repetitive motion, and legged robots.

The main advantages of the proposed algorithm are its simplicity, its low computational requirements, and its robustness against fluctuations of the input signals due to noise, and changes in the energy consumption profile of the executed task. Also, the algorithm does not require a precise knowledge of the controlled system, besides the physical limits of the configuration parameter and the task periodicity. The proposed algorithm can also be extended to a multiple-degree-of-freedom case to deal with complex systems with several VSAs.

The structure of the algorithm follows that of the recursive gradient optimization. To allow its use in a real application, subject to variations of the optimal configuration and noise in the measurements, we introduced the following modifications: (1) a modification of the step size to avoid the direct use of derivatives. This modification also simplifies the implementation of the algorithm and reduces the computational load. (2) A forgetting term which, in conjunction with the modified step size and a projection rule, allows to detect changes on the objective function minimizer. The forgetting term also provides a mechanism to avoid stagnation due to errors in the measurements. In the ideal case, convergence to a small region that contains the minimizer can be guaranteed if the objective function is convex and the algorithm parameters are sufficiently small.

The proposed algorithm was successfully tested on a prototype of a VSA based on the Mechanically Adjustable Compliance and Controllable Equilibrium Position Actuator (MACCEPA) [23], [24]. The algorithm was also tested in a simulation of a two-degree-of-freedom system with a detailed model of the actuator. For the presented examples, the total electrical input energy of the main motor of the actuators was selected as the objective function.

The behavior of the algorithm was evaluated in the presence of noise in the input signals, variation of the actuator control action due to its nonlinear behavior, the actual dynamics of the controlled system, and variation of the operation conditions due to load changes. In the performed experiments and simulations the proposed algorithm was able to successfully drive the systems to a zone of minimal electrical energy consumption.

II. ADAPTATION OF THE ACTUATOR CONFIGURATION

A. Adaptation of the configuration parameter

The proposed adaptation mechanism is aimed at iteratively finding the value of a configuration parameter $p$ that minimizes an objective function $E$ in a repetitive task performed by the actuator. For a single-degree-of-freedom case $p \in P = \{x \in \mathbb{R} \mid p_{\text{min}} \leq x \leq p_{\text{max}}\}$, where $p_{\text{min}}$ and $p_{\text{max}}$ represent the physical limits of $p$. In principle, $p$ can be any physical parameter that influences $E$. The only requirement for $E$ is that $E(p)$ is convex in $P$, and that a proper measurement or estimation of it is available in real time.

In our application, the actuator is assumed to be tracking a reference signal in a stable fashion to produce a required cyclic output with a certain amount of energy. There is a certain amount of energy, considered unknown, that is either entering the system due to its interaction with the environment or that has already been stored in the system as a result of the natural energy exchange between its components (e.g., mass of the actuator load and its elastic element). The total amount of energy required to perform the desired task is assumed to depend on the selected configuration parameter.

The adaptation of the configuration by the proposed algorithm is intended to be performed along several task cycles to spread its energy cost and reduce its impact on the overall energy consumption. For this reason, both $p$ and $E$ are assumed to be sampled at a rate $h$ every $\nu$ task cycles, where $\nu$ is a positive integer (i.e., the execution rate cannot be less than one task cycle). The sampling rate $h$ is not necessarily related to a time scale but to an event-driven sequence adapted to the actual fluctuation of the cycle length in a practical implementation of the algorithm.

The adaptation mechanism has the following structure:

$$p'(h + 1) = p(h) - \delta_p G(E, p) + f_f \quad (1)$$

where $p'(h + 1)$ is a preliminary estimate of the next-iteration value of $p$, $\delta_p$ is a positive constant that represents the amount by which the configuration parameter will be changed at each iteration, $G$ represents the direction of the adaptation, as in the gradient-optimization algorithm [25], and $f_f$ is a forgetting term. $G$ is calculated as

$$G(E, p) = \text{sign}(E(h) - E(h - 1))[p(h) - (p(h) - 1)] \quad (2)$$

where

$$\text{sign}(z) = \begin{cases} +1 & \text{if } z > 1 \\ 0 & \text{if } z = 0 \\ -1 & \text{if } z < 1 \end{cases}$$

In an ideal case, if $E$ is convex, $p'$ will naturally converge to the minimizer of $E$. However, in practice, the measurements of $E$ may be affected by noise, changes in the dynamical behavior of the system, and changes in the interaction forces. The objective function may also exhibit small variations around the minima (flatness). Thus, there is the possibility that $E(h) = E(h - 1)$ depending also on the resolution of the digital implementation of the algorithm. This condition eventually leads to $p(h) = p(h - 1)$ in subsequent iterations, producing stagnation of the algorithm. A similar situation can be observed when $p'$ reaches the boundary of $P$. 
The algorithm will subsequently produce the sequence \( p(h) = p(h-1) \) and is calculated as

\[
f_f = \frac{(\delta_p)^h}{1 + \gamma |p(h) - p(h-1)|^2}
\]

where \( \delta_p \) is a positive amount by which the estimation of \( p^* \) will be changed in the current iteration. \( \gamma > 0 \) should be sufficiently large to have \( f_f \rightarrow 0 \) when \( p(h) \neq p(h-1) \).

The forgetting term, which is a finite and fixed amount directly added to the estimate of the configuration parameter in our scheme, should not be confused with the forgetting factor sometimes used in integral adaptive laws that modifies the adaptation rate [25], [26].

The following projection rule is used to keep \( p \) within its practical limits and to avoid stagnation at the extrema

\[
p(h+1) = \begin{cases} 
    p_{\text{min}} + \delta_p, & \text{if } p'(h+1) < p_{\text{min}} \\
    p_{\text{max}} - \delta_p, & \text{if } p'(h+1) > p_{\text{max}} \\
    p'(h+1), & \text{otherwise}
\end{cases}
\]

(4)

where \( \delta_p \) is a small positive constant.

The algorithm can be implemented following the procedure listed in Table I. This table summarizes the essential steps that need to be taken to use the algorithm once the values for \( \delta_p, \delta_p', \) and \( \delta_p \), have been chosen (see Section II-C bellow). Starting at \( h = 1 \), the algorithm requires two different consecutive initial values of \( p; p_{0.1} \) and \( p_{0.2} \). (where \( p_{0.2} = p_{0.1} \pm \delta_p \) could be the simplest choice) for which their corresponding values of \( E \) have to be measured at the end of the corresponding task cycles. With these preliminary values of \( p \) and \( E \) the algorithm can be started and maintained in a continuous loop until some stop criteria are reached.

**B. Behavior of the adaptation law**

Let us consider the case in which

\[
p(h+1) = p(h) - \delta_p G
\]

(5)

Under ideal conditions, if \( E \) is strictly convex in \( P \) with a strict minimizer \( p^* \) such that \( p_{\text{min}} < p^* < p_{\text{max}} \), the algorithm in (5) presents convergence properties similar to those of the gradient algorithm with a normalized step size. This can be better appreciated in a graphical form if we write \( G \) as

\[
G = \frac{E(h) - E(h-1)}{|g(h)|}, \quad p(h) \neq p(h-1)
\]

\[
\frac{g(h)}{|g(h)|}, \quad g(h) \neq 0
\]

where

\[
g(h) = \frac{E(h) - E(h-1)}{p(h) - p(h-1)}
\]

(6)

As can be seen in Fig. 1, \( -g(h)/|g(h)| \) represents the direction towards \( p^* \) at \( p(h) \) for a sufficiently small \( \delta_p \). As \( p \) is updated by adding or subtracting a fixed amount \( \delta_p \) at each iteration, \( p \) will converge to \( \hat{p} \), such that \( |\hat{p} - p^*| \) is minimum

\[
|\hat{p} - p^*| \leq \delta_p
\]

(7)

The algorithm will subsequently produce the sequence

\[
p = \hat{p} + \delta_p, \quad \hat{p} \rightarrow \hat{p} - \delta_p, \quad \hat{p} \quad \hat{p} + \delta_p, \ldots
\]

(8)

In the example in Fig. 1 both \( p(h+1) \) and \( p(h) \) satisfy (7) but the alternating sequence will be generated around \( p(h) \) which shows the minimal distance towards \( p^* \). This alternating sequence is at the core of the proposed algorithm that allows it to detect variations of \( p^* \). In conventional optimization schemes [27] the adaptation mechanism is usually stopped under some criteria. In contrast, the proposed algorithm continues sampling \( E \) around the initial value of \( \hat{p} \) in a systematic fashion. If a sufficiently large change occurs on \( p^* \), the switching behavior of (5) will detect the corresponding change in the slope of \( E \) and eventually will find a new value for \( \hat{p} \) and a new alternating sequence around it.

If \( E \) is convex but not strictly convex, for instance if \( E \) has a flat region of minima, the algorithm in (5) would stall at some point \( \hat{p} \) in this region. This situation is due to two identical consecutive measurements of \( E \) or due to a perturbation that produces a sudden change on \( E \) from one cycle to the other. If at a given sampling time \( h_s \), \( E(h_s) = E(h_s-1) \), then \( G = 0, p(h_s+1) = p(h_s) \), and subsequently \( p(h_s+d) = p(h_s+1), d = 2, 3, \ldots \). To recover the capability to detect changes on \( p^* \), the term \( f_f \) in (3) is used to automatically add \( (-1)^h \delta_p \) to (5), whenever \( p(h) = p(h-1) \), provided a sufficiently large \( \gamma \). This term will produce a new alternating sequence \( p = \hat{p} + \frac{1}{2} \delta_p, \quad \hat{p} - \frac{1}{2} \delta_p, \quad \hat{p} + \frac{1}{2} \delta_p, \ldots \) that will allow to continue sampling \( E \) around \( p \) and detect eventual changes in its profile. When \( |p^* - p_{\text{min}}| < \delta_p \) or \( |p^* - p_{\text{max}}| < \delta_p \) such that \( p(h+1) \) lies outside the boundary of \( P \), its value is adjusted using the projection rule (4) which will move it to the interior of \( P \) by a distance \( \delta_p \) from the boundary. In this case the alternating sequence would have the form \( p = p_{\text{min}}, p_{\text{min}} + \delta_p, p_{\text{max}}, p_{\text{max}} + \delta_p, \ldots \), if \( p(h+1) < p_{\text{min}} \), or \( p = p_{\text{max}}, p_{\text{max}} - \delta_p, p_{\text{min}}, p_{\text{min}} - \delta_p, \ldots \), if \( p(h+1) > p_{\text{max}} \). If the objective function is not convex we can only expect convergence to a local minimum.

The ideal behavior of the algorithm is illustrated in Fig. 2. In this numerical example, \( E \) is a quadratic function that emulates
hand, a constant step size implies a constant adaptation rate. In a more realistic situation, the measurements of $E$ will be affected by noise. If we assume that these measurements are represented by $E(h) + \epsilon(h)$, where $\epsilon$ represents random errors with the property $|\epsilon(h)| < \epsilon_0$, the algorithm in (1) will continue to work as described in the previous paragraphs as long as the direction towards the minimizer is still distinguishable. As it is shown in the example in Fig. 1, even in the worst case the algorithm will approach to a region $P_\epsilon$ that contains $p^*$ until the measurements of $E + \epsilon$ overlap for contiguous values of $p$. In this example, for $E(h - 2)$ and $E(h - 1)$ the calculation of the next value of $p$ will still go in the right direction even if the level of noise reaches its expected maximum. However, for $E(h)$ and $E(h - 1)$ there are some values of $\epsilon$ that could lead to a wrong estimation of the direction towards $p^*$, as it is illustrated with the points $M$ and $N$. In the shown example, $P_\epsilon$ would contain $(p(h + 1), p(h), p(h - 1))$. The direction towards $p^*$ is corrected, however, every time its estimated value lies outside $P_\epsilon$. The only inconvenience in this case is that the estimation of $p^*$ can freely move inside $P_\epsilon$ and that $P_\epsilon$ can be large if $E$ is relatively flat in the vicinity of $p^*$. External mechanisms can be implemented to detect if the current estimation of $p^*$ is in such a region to slow down or stop the optimization process, as it would be the case in any other optimization algorithm used in a practical situation. An example of such a mechanism is given in Section III-B.

### C. Parameter selection

The behavior of the algorithm clearly depends on the selected values for its parameters. In particular, $\delta_p$ plays an important role in the correct estimation of the direction of adaptation. In the ideal case, the smaller $\delta_p$ is, the better the estimation of the direction of adaptation would be. On the other hand, in a practical case a very small $\delta_p$ would render the adaptation slow and/or very sensitive to noise in the measurements of $E$. Thus, a compromise needs to be found. One way to do this is to find a reasonable $\delta_p$ that is in agreement with the physical capabilities of the device. Sensor accuracy, mechanical play, output average error, and other information can be used to determine the accuracy at which the desired values of $p$ can actually be produced by the system, and thus to determine a meaningful lower bound for $\delta_p$ (and by extension, a lower bound for $\delta_{p_r}$ and $\delta_{p_r}$). The dynamics of the mechanism that physically adjusts the configuration parameter also plays an important role. During continuous operation of the VSA, the duration of the task cycle with respect to the time required to move the configuration

#### TABLE I

**ADAPTATION PROCEDURE ($\nu = 1$ task cycle).**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Initialization ($h = 1$), set the physical value of $p$ to $p_{0,1}$, execute one task cycle and measure $E(1)$ at the end of the task.</td>
</tr>
<tr>
<td>2.</td>
<td>Initialization ($h = 2$), set the physical value of $p$ to $p_{0,2}$, execute one task cycle and measure $E(2)$ at the end of the task.</td>
</tr>
<tr>
<td>3.</td>
<td>Estimation of the next value of $p$, calculate $p'(h + 1)$ using (1), (2), and (3).</td>
</tr>
<tr>
<td>4.</td>
<td>Correction of the estimated $p$, adjust $p'(h + 1)$ using (4).</td>
</tr>
<tr>
<td>5.</td>
<td>Task execution, set the physical value of $p$ to $p(h + 1)$, increment $h$, execute one task cycle, and measure $E$ at the end of the task.</td>
</tr>
<tr>
<td>6.</td>
<td>Restart/Termination, if termination criteria are satisfied, stop the algorithm; otherwise return to Step 3.</td>
</tr>
</tbody>
</table>

![Configuration parameter vs. Objective function](image1)

![Task cycle](image2)

Fig. 2. Recursive adaptation of the actuator’s configuration in an ideal case when the objective function suddenly changes in time: (a) The two objective functions with different optimal configurations. (b) Adjustment of the configuration parameter according to the change of the objective function occurring at 40 s.
parameter from one point to another may also serve as an indication of the maximum allowable value for \( \delta_p \). In principle, \( \delta_p, \delta_{pf}, \) and \( \delta_{pr} \) can all have different values. However, once a suitable value for \( \delta_p \) has been chosen, the algorithm can be simplified setting \( \delta_{pf} = \delta_{pr} = \delta_p \) without altering its fundamental behavior.

D. Multiple-degree-of-freedom case

The proposed algorithm can be extended to a multiple-degree-of-freedom case recalling that (6) can be taken as an approximation of \( \frac{dE}{dh} \) for a sufficiently small \( \delta_p \). Let us consider the case of \( n \) degrees of freedom in which there are \( n \) different actuators on the system and that \( E = E(p_1, p_2, \ldots, p_n) \to \mathbb{R} \) is convex with \( p_1, p_2, \ldots, p_n \) being the corresponding configuration parameters. To preserve the gradient structure of the algorithm, the variation of \( E \) with respect to the \( i \)-th configuration parameter needs to be determined holding constant the rest of the parameters. In other words, the objective function will be measured updating one configuration parameter at a time in a sequential manner. The ensemble of the adaptation laws that would achieve this can be generated with the following expression:

\[
p_i^t(h + q) = \begin{cases} 
p_i(h + q - 1) - \delta_i \left[ G_i + f_i \right] & \text{if } i = q \\
p_i(h + q - 1) & \text{otherwise (9)} \end{cases}
\]

where \( \delta_i \) is the increment of the \( i \)-th configuration parameter and

\[
G_i = \text{sign}([E(h + q - n) - E(h + q - n - 1)] \\times [p_i(h + q - n) - p_i(h + q - n - 1)])
\]

is the corresponding estimation of the direction of adaptation. The sampling variable is updated as \( h = 1, n + 1, 2n + 1, \ldots \). For each value of \( h \) there is a finite sequence \( q = 1, 2, \ldots, n \) and for every value of \( q \) there is another finite sequence \( i = 1, 2, \ldots, n \) that designates the \( i \)-th configuration parameter. The forgetting term and the projection rule can be extended in the following manner:

\[
f_i = \frac{(-1)^j}{1 + \gamma_i [p_i(h + q - n) - p_i(h + q - n - 1)]^2}
\]

and

\[
p_i(h + q) = \begin{cases} 
p_i^{\text{min}} + \delta_i & \text{if } p_i^t(h + q) < p_i^{\text{min}} \\
p_i^{\text{max}} - \delta_i & \text{if } p_i^t(h + q) > p_i^{\text{max}} \\
p_i^t(h + q) & \text{otherwise} \end{cases}
\]

where \( j = h \) if \( n \) is odd and \( j = 1, n + 2, 2(n + 1) + 1, 3(n + 1) + 1, \ldots \) if \( n \) is even. For simplicity, the adaptation step size was assumed identical for the adaptation law, the forgetting term, and the projection rule, thus \( \delta_{pf_i} = \delta_{pr_i} = \delta_p_r = \delta_p \).

III. EXPERIMENTAL RESULTS

A. Actuator description

The proposed algorithm was tested on a one-degree-of-freedom prototype of the VSA described in [24] (Fig. 3). This actuator consists of two mechanisms: the driving mechanism (DM) that controls the equilibrium position of the actuator, defined as the angular position of a lever arm (LA); and the stiffening mechanism (SM) that controls the stiffness settings through the manipulation of the position of one of the ends of a spring. The output of the actuator is determined by two main factors: the torque provided by the driving motor (DM motor) and the interaction forces acting on the output link (OL).

The equilibrium position of the actuator can be defined as the position of LA, denoted by \( \varphi \), when no interaction forces are present on OL. In this particular case, OL and LA are aligned (Fig. 4, top diagram). When an interaction force is acting on OL its position, denoted by \( \theta \), will change with respect to that of LA, pulling on the cable and modifying the position of the shuttle with respect to that of the SM nut (Fig. 4, bottom diagram). The change in the position of the shuttle induces a deformation of the spring and creates a reaction torque at OL. By changing the position of the SM nut through an additional motor, the SM motor, we can manipulate the deformation of the actuator’s spring (i.e., the force that it exerts on the cable), changing the reaction torque acting on OL and, by extension, the apparent output stiffness of the actuator. In the actuator prototype an inertial load of mass \( m \) can be attached to the free end of OL.

The current prototype is equipped only with an optical encoder to measure the OL position and two load cells in parallel that measure the tension in the cable. Thus, the configuration of the actuator can only be changed when there are no interaction forces on OL as we do not have information relative to the position of LA. The measured force in the cable, directly related to the deformation of the actuator’s spring and its output stiffness, corresponds to \( p \) in our experimental system.
Both actuator motors are driven by commercial motor drives set in velocity control mode commanded by a proportional-integral-derivative (PID) force controller, in the case of the SM motor, and a proportional (P) position controller, in the case of the DM motor. The motor drives provide real-time measurements of the motor current and motor velocity that are used to calculate the motor voltage, and subsequently the motor electrical input power and energy. The system is controlled by a real-time data-acquisition system in which the adaptation algorithm is also implemented using Simulink Desktop Real Time®. The algorithm in the Simulink model is implemented for the current sampling time \( h \) instead of \( h + 1 \) [e.g., \( p'(h) = p(h - 1) - \delta_p G(h - 1) + f_f(h - 1) \)] to avoid algebraic loops.

\[ E = E_{DM} = \int_{t_0}^{t_0 + t_s} |P_{DM}| dt , \]

where \( P_{DM} = v_M i_M \) is the DM motor input power; \( v_M \) and \( i_M \) are the DM motor voltage and current, respectively; \( t_0 \) is the initial time of the cycle; and \( t_s = 10 \) s is the duration of the cycle. No energy regeneration (energy flow going from the mechanical subsystem to the electrical power source) is considered.

The performance of the actuator is subject to random perturbations due to the fact that the DM motor drive, set in velocity-control mode, calculates the actual motor velocity based on the information provided by the motor Hall sensors.

\[ p'(h) = p(h - 1) - \delta_p G(h - 1) + f_f(h - 1) \]

This information is less accurate when the motor velocity is in the vicinity of zero. When the actuator is tracking the zero-displacement segment of the reference, the actual position of the motor shaft tends to oscillate around zero, occasionally provoking a sudden corrective action of the OL position controller when this oscillation significantly affects the OL position (Fig. 6). This corrective action can represent a momentary increase of the DM motor energy consumption between 15% and 80% of its average value depending on the load and the actual tension in the cable (which directly affects the backlash at the output). The proposed algorithm is able to cope with this sort of perturbations, as it will be explained in the following paragraphs.

In a second series of experiments, the adaptation algorithm is run during 50 cycles with the FSM active to verify if it can actually find the desired configuration that minimizes \( E \).
starting from an arbitrary initial condition, in the presence of the random perturbations described above and load changes.

The following experimental results were gathered using \( \delta_p = \delta_p_1 = 10 \) N, \( \gamma = 1 \times 10^{12} \), \( \beta_{nom} = 1 \) cycle, \( h_l = 30 \) cycles, \( h_{sm} = 20 \) cycles, and \( \lambda = 0.5 \) J. The selection of \( \delta_p = 10 \) N is justified considering the fact that our experimental prototype can control the static force on the spring with an average error of 5.28 N. For each of the selected loads the algorithm is started either with the configuration parameter set to its minimum or its maximum allowable value, \( p(0) = \min_p = 0 \) N or \( p(0) = \max_p = 150 \) N, respectively.

Figs. 8 and 9 show that the proposed algorithm is able to converge to a region around the expected average optimal value of \( p \) in about 15 cycles. After 20 cycles, the algorithm is able to drive the system from an initial required DM motor input energy around 56.0 J/cycle (\( p(0) = \min_p \)) and 47.4 J/cycle (\( p(0) = \max_p \)) to an average of 44.8 J/cycle, for \( m = 3.0 \) kg. For \( m = 0.5 \) kg the system goes from an initial average required motor energy of 42.8 J/cycle (\( p(0) = \min_p \)) and 41.2 J/cycle (\( p(0) = \max_p \)) to an average minimum of 39.8 J/cycle after 20 cycles. The reduction of the energy requirements can be better appreciated in Fig. 10 which presents four data sets where no perturbations were registered at the beginning of the experiments.

### IV. Simulation Results

In this section we illustrate the use of the proposed algorithm in a double pendulum actuated by two VSAs. The simulated system is shown in Fig. 11 and consists of two rigid links, \( l_1 \) and \( l_2 \), with lumped masses, \( m_1 \) and \( m_2 \), as indicated.

The revolute joints \( a \) and \( b \) are actuated by two identical VSAs, \( \text{VSA}_1 \) and \( \text{VSA}_2 \), based on a previous prototype of the actuator described in [24]. In this version of the VSA \( \rho \) represents the displacement of the SM nut. The dynamical behavior of the actuators is given by the following model:

- **VSA:** \[
\begin{align*}
L_M \frac{dv_{M}}{dt} &+ R_M i_M + k_M \dot{\varphi}_i = v_{M} \\
J_D \ddot{\varphi}_i + B_D \dot{\varphi}_i + n_D T_i = k_N \dot{i}_M \\
L_m \frac{di_m}{dt} + R_m i_m + k_m \dot{\varphi}_i = i_m \\
J_S \dot{\varphi}_i + B_S \dot{\varphi}_i + f_S(t) + n_S F_S = k_S \dot{i}_m
\end{align*}
\]

where \( i_M \) is the DM motor current, \( \varphi_i \) is the LA angular position (in the diagram \( \varphi_i = \theta_i + \alpha_i \)), \( v_M \) is the DM motor voltage, \( i_m \) is the SM motor current, and \( v_{m} \) is the SM motor voltage. The actuator output torque is given by \( T_i = F_S \frac{BD}{A_i} \sin(\varphi_i - \theta_i) \) with \( F_S = k(A_i + B + pi - D) \) representing the spring force and \( A_i = \sqrt{B^2 + D^2 - 2BD \cos(\varphi_i - \theta_i)} \) representing a geometrical parameter that relates the spring force to the difference in
Fig. 10. Reduction of the DM motor energy requirements for two different loads and different initial conditions with no significant perturbations on the OL displacement.

Fig. 11. Double pendulum actuated by two VSAs.

The angular position of LA and OL. The term \( f_{S_1}(t) = (b_1 + b_2 f_{S_2}) \text{sign}(p_1) \) represents nonlinear friction phenomena in the SM spindle where \( b_1 \) and \( b_2 \) are positive constants. Table II presents a complete list of the parameters and their respective values used in the numerical simulations.

The actuators’ output torque is controlled using a PID with a nested PID position controller on the actuators’ LA position and a current PI control loop on the DM motor, similar to that used in conventional motor drives. The double pendulum is controlled using a computed-torque control scheme [28] with an outer PD control loop on the absolute positions of the links, \( q_1 \) and \( q_2 \). This controller is tuned for the nominal masses of the pendulum, \( m_1 = m_2 = 2.0 \) kg. In Fig. 11, \( q_1 = \theta_1 \) and \( q_2 = \theta_1 + \theta_2 \). The computed torque is set as the reference input to the VSAs. The internal model signals used for feedback are affected by uniformly distributed noise. The simulated motor outputs and other modeled mechanical elements are affected by the backlash specified in the components’ specifications. The motor controllers are also saturated according to the maximum allowable motor current. The SM nut positions in the model are controlled with a PI and a nested motor-velocity PI controller emulating the SM motor drives.

The system is tracking two sinusoidal references given by \( q_{1r} = a_1 \sin(\omega_1 t) \) and \( q_{2r} = a_2 \sin(\omega_2 t) \) to conform a task cycle with a duration \( t_s = 25.13 \) s which correspond to two sinusoidal cycles for \( q_1 \) and one sinusoidal cycle for \( q_2 \) (\( a_1 = a_2 = 1.56 \) rad, \( \omega_1 = 0.50 \) rad/s, \( \omega_2 = 0.25 \) rad/s).

The objective function in this case is chosen as the total absolute input energy of the DM motors required per task cycle, thus

\[
E = E_{DM_1} + E_{DM_2} = \int_{t_0}^{t_0 + t_s} (|P_{DM_1}| + |P_{DM_2}|) dt
\]

where \( P_{DM} = v_{M_1 i_{M_1}} \) is the DM motor input power. The adaptation algorithm is implemented for \( n = 2 \), thus (9), (10), and (11) are expanded in the following manner:

\[
\begin{align*}
(q = 1, i = 1): & \quad p_1'(h + 1) = p_1(h) - \delta_1 [G_1 + f_{f_1}] \\
(q = 1, i = 2): & \quad p_2'(h + 1) = p_2(h) \\
(q = 2, i = 1): & \quad p_1'(h + 2) = p_1(h + 1) \\
(q = 2, i = 2): & \quad p_2'(h + 2) = p_2(h + 1) - \delta_2 [G_2 + f_{f_2}]
\end{align*}
\]

with the adaptation directions given by

\[
\begin{align*}
G_1 &= \text{sign}([E(h - 1) - E(h - 2)] [p_1(h - 1) - p_1(h - 2)]) \\
G_2 &= \text{sign}([E(h) - E(h - 1)] [p_2(h) - p_2(h - 1)])
\end{align*}
\]

and the forgetting terms given by

\[
\begin{align*}
\gamma_1 &= (1 - \lambda)^{j-1} \\
\gamma_2 &= (1 - \lambda)^{j-1}
\end{align*}
\]

where \( q = 1, 4, 7, 10, \ldots \) as \( n \) is even in this example (see Section II-D). The expansion of the projection rules is straightforward. Notice that the sampling variable is updated as \( h = 1, 3, 5, \ldots \) for the calculation of the algorithm but the configuration parameters are updated every task cycle in the physical system (\( \nu = 1 \) task cycle). In other words, for a given sampling time \( h \), (12) and (13) are calculated and \( p_1 \) and \( p_2 \) updated in the system before executing (14) and (15) in the following sampling time. The parameters of the algorithm are chosen as \( \delta_i = 1 \) mm, and \( \gamma_i = 1 \times 10^{12} \). No specific

<table>
<thead>
<tr>
<th>Parameter (symbol)</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spring stiffness ( k )</td>
<td>68.68</td>
<td>N/m</td>
</tr>
<tr>
<td>Effective length of OL ( (D) )</td>
<td>0.056</td>
<td>m</td>
</tr>
<tr>
<td>Effective length of LA ( (E) )</td>
<td>0.047</td>
<td>m</td>
</tr>
<tr>
<td>DM motor inductance ( (L_{DM}) )</td>
<td>0.031</td>
<td>mH</td>
</tr>
<tr>
<td>SM motor inductance ( (L_{SM}) )</td>
<td>0.031</td>
<td>mH</td>
</tr>
<tr>
<td>DM motor resistance ( (R_{DM}) )</td>
<td>0.341</td>
<td>Ω</td>
</tr>
<tr>
<td>SM motor resistance ( (R_{SM}) )</td>
<td>0.341</td>
<td>Ω</td>
</tr>
<tr>
<td>DM angular velocity constant ( (k_{w_{DM}}) )</td>
<td>1.199</td>
<td>rad/s</td>
</tr>
<tr>
<td>SM linear velocity constant ( (k_{w_{SM}}) )</td>
<td>0.204</td>
<td>rad/s</td>
</tr>
<tr>
<td>DM motor torque constant ( (k_{T_{DM}}) )</td>
<td>0.014</td>
<td>N m/ rad</td>
</tr>
<tr>
<td>SM torque constant ((k_{T_{SM}}))</td>
<td>0.065</td>
<td>N m/ rad</td>
</tr>
<tr>
<td>DM mass moment of inertia ( (J_{DM}) )</td>
<td>0.080 × 10^{-4}</td>
<td>kg m^2</td>
</tr>
<tr>
<td>SM mass moment of inertia ( b ) ( (J_{SM}) )</td>
<td>0.324</td>
<td>kg m</td>
</tr>
<tr>
<td>DM transmission ratio ( (n_{DM}) )</td>
<td>0.011</td>
<td>−</td>
</tr>
<tr>
<td>SM transmission ratio ( (n_{SM}) )</td>
<td>0.570</td>
<td>rad</td>
</tr>
<tr>
<td>DM friction coefficient ( a ) ( (b_{DM}) )</td>
<td>0.035</td>
<td>rad</td>
</tr>
<tr>
<td>SM friction coefficient ( a ) ( (b_{SM}) )</td>
<td>2.262 × 10^3</td>
<td>rad</td>
</tr>
<tr>
<td>SM Coulomb friction constant ( a ) ( (b_{C}) )</td>
<td>0.200</td>
<td>N m</td>
</tr>
<tr>
<td>SM stiction friction constant ( a ) ( (b_{s}) )</td>
<td>0.520</td>
<td>mm</td>
</tr>
<tr>
<td>Length of the pendulum links</td>
<td>0.350</td>
<td>m</td>
</tr>
<tr>
<td>Pendulum fixed mass ( (m_{OL}) )</td>
<td>2.0</td>
<td>kg</td>
</tr>
</tbody>
</table>

\( ^a \) Experimental estimation.
\( ^b \) Normalized to length.
strategy is implemented to stop the adaptation algorithm in the numerical simulations.

To determine the profile of the selected objective function, in a preliminary simulation the system is set to track the above-mentioned position references while the configuration parameters are swept in all the allowable range \((p_1 \in [0.0, 20.0] \text{ mm})\) for a load \(m_2 = 2.5 \text{ kg}\) (the configuration algorithm is not run at this time). Fig. 12 shows the corresponding mesh and contour plots that represent the average \(E\). As it can be seen, the objective function appears to be strictly convex with a minimizer \(p^* = [p_1^* \ p_2^*]^T = [0 \ 0]^T \text{ mm}\) for which \(E = 903 \text{ J/cycle}\).

![Fig. 12. Profile of the motor energy consumption for \(m = 2.5 \text{ kg}\) and recursive parameter modification: (a) Contour plots, (b) 3D representation.](image)

In a second simulation the algorithm is run with the initial conditions \(p_1(0) = 20.0 \text{ mm}\) and \(p_2(0) = 15.0 \text{ mm}\). Figs. 12(a) and (b) show the trajectory described by the successive approximations (small circles) of the configuration parameters. After 40 cycles the estimation of the optimal configuration converges to a zone that contains the identified minimizer.

In a similar simulation using \(m_2 = 0.5 \text{ kg}\), for which the pendulum controller produces a larger overshoot in comparison to the nominal value, the average of the objective function exhibits two local minima: one at \(p = [p_1 \ p_2]^T = [0 \ 10]^T \text{ mm}\) for which \(E = 1984 \text{ J/cycle}\), and another at \(p = [8 \ 0]^T \text{ mm}\) for which \(E = 1228 \text{ J/cycle}\). As the convexity requirement is not fulfilled in this case, the algorithm cannot guarantee convergence to a global minimum. Nevertheless, as explained in Section II-B, the algorithm should be able to converge to a local minimum. Fig. 13 illustrates how the algorithm converges to a region that contains \([8 \ 0]^T\) in less than 40 iterations starting at \(p(0) = [14 \ 18]^T \text{ mm}\).

![Fig. 13. Profile of the motor energy consumption for \(m = 0.5 \text{ kg}\) and recursive parameter modification: (a) Contour plots, (b) 3D representation.](image)

To verify the ability of the algorithm to track the change in the energy consumption profile, in a third simulation the system is initially run with \(m_2 = 2.5 \text{ kg}\) and after 70 cycles the load is changed to \(m_2 = 0.5 \text{ kg}\). Fig. 14(a) depicts the change in the value of \(E\) as \(p\) approaches the minimizer. The sudden change of \(E\) at iteration 70 is due to the change in \(m_2\). From iteration 70 to iteration 80 the algorithm adjusts \(p\) and drives again the system to a region of low energy consumption. Fig. 14(b) shows the modification of \(p_1\) and \(p_2\).

The proposed algorithm is able to converge to a zone that contains the minimizer \([0 \ 0]^T \text{ mm}\) in about 40 iterations. The configuration parameters remain in this zone until the mass \(m_2\) is modified after 70 cycles. After this change, the first configuration parameter is properly adjusted until it reaches a region that contains one of the new expected minimizers at \([8 \ 0]^T\) mm.

![Fig. 14. Behavior of the algorithm during a system load change: (a) Energy consumption profile, (b) Iterative modification of the configuration parameters.](image)

V. DISCUSSION

Changing stiffness in a continuous manner in MACCEPA-based actuators, as well as some other VSA designs, may require an excessive amount of energy [10]. For this reason, the proposed algorithm is intended for repetitive tasks with a relatively low rate of change of the actual optimal configuration. In the context of human-like locomotion, for which the algorithm is originally intended, the actuator will most likely be subject to quasi-cyclic motion (e.g., ground-level walking) for which the changes in the optimal configuration are expected to happen gradually and after several cycles with similar characteristics (e.g., transition after several gait cycles from walking on a horizontal surface to an inclined plane or from one type of terrain to another). The actuator reconfiguration in our experimental and simulation examples is done over several cycles to spread the energy cost of the adaptation process among the entire sequence of motion.

As confirmed by the experiments and simulations, the proposed algorithm is able to cope with the variation of the objective function due to changes in the operation condition, perturbations, and signal noise. In the presented examples, the algorithm successfully drove the actuator’s stiffness settings to a region where the energy requirements of the DM motor were minimal in a relatively small amount of iterations. For the experimental case the energy required to adjust the actuator configuration from one iteration to the other is in average
3.5 J/cycle. This represents an average of 70 J that would be added to the total energy consumption of the actuator assuming that the algorithm reaches the desired minimum in 20 iterations.

It should be noted that in the type of VSA used in this work, the mechanism that regulates the output position of the actuator is independent from the one that regulates the stiffness configuration. Thus, during a static condition (with zero output), the modification of the stiffness configuration does not affect the output position of the actuator. The adjustment of the stiffness value is expected to be done between tasks and not during the execution of a task (e.g., during the swing phase in a gait cycle). Thus, for practical purposes, the stiffness configuration of the actuator can be considered fixed within the execution of each task during which the dynamical behavior of the actuated system is not affected by the change in the stiffness configuration.

VI. Conclusions

This manuscript presented an algorithm to adjust the configuration of a variable-stiffness actuator to minimize the energy requirements of its driving mechanism during the execution of a repetitive task. The algorithm is based on a gradient-optimization scheme and directly utilizes online measurements of the system’s energy consumption. The adaptation process does not require a precise knowledge of the system dynamics nor the performed task besides the physical limits of the configuration parameter and the periodicity of the task.

Clear advantages of the proposed algorithm are its simplicity, relatively low computational requirements, easy practical implementation, and the possibility of being extended to a multiple-degree-of-freedom case. In contrast to most of the related algorithms currently available in the literature, focused on the minimization of mechanical energy requirements at the output of the actuator, the proposed algorithm can directly deal with the electrical input energy requirements of the system without the need of a mathematical model of the mechanical transmissions or electrical motors.

The behavior of the algorithm was demonstrated using an experimental prototype of a variable-stiffness actuator under typical perturbations of the energy consumption profile due to system parameter variation, noise, and the inherent nonlinear dynamics of the system. The algorithm in its multiple-degree-of-freedom version was tested in numerical simulations using a double pendulum actuated by two variable-stiffness actuators. In both cases, the proposed algorithm was able to adjust the configuration of the actuators to drive its main electrical motors to a working regime with minimal energy requirements.

Minor disadvantages of the proposed algorithm are its constant adaptation rate and the fact that the reconfiguration of several actuators can only be done in a sequential fashion. As part of the future work other mechanisms to estimate the direction of adaptation will be explored to improve the convergence speed of the algorithm. In particular, this will be done for the case of multiple degrees of freedom as the number of iterations to find the optimal configuration increases proportionally to the number of configuration parameters in the current form of the algorithm. Also, the authors are currently looking for a suitable extension of the algorithm to include the energy requirements of the adaptation mechanism by treating it as a supplemental control input that modifies the elastic energy of the system.

The use of the proposed algorithm is not limited to VSAs. In principle, nothing prevents its implementation in other actuation systems where a configuration parameter can be clearly defined and where the executed tasks are repetitive.

Acknowledgment

This work was funded by the European Commission ERC Starting grant SPEAR (No. 337596). The authors would like to thank Marnix De Boom and Marc Luypaert for their work building the tested prototype.

References


Rene Jimenez Fabian received his Doctoral degree in engineering from Universidad Nacional Autónoma de México (UNAM) in 2006. He has worked in several research projects at different institutions including the Faculty of Engineering and the Engineering Institute of UNAM (1998 - 2007), the Earthquake Engineering Research Center at University of California, Berkeley (2007 - 2008), and the Department of Theoretical Mechanics, Dynamics and Vibrations at Université de Mons (2009 - 2011). He has been working on the development of new elastic actuators at the Robotics and Multibody Mechanics Research Group at Vrije Universiteit Brussel since 2012.

Maarten Weckx was born in Wilrijk, Belgium, in 1989. He received the Master of Science degree in Mechanical Engineering in February 2013. He has been working on the development of new actuator technologies at the Department of Mechanical Engineering of Vrije Universiteit Brussel within the Robotics and Multibody Mechanics Research Group since March 2013.

David Rodriguez-Cianca received the B.Sc and M.Sc degrees in Mechanical Engineering from the Technical University of Madrid, Spain, in 2014 and 2016 respectively. He combined his last years of college with his work as research assistant at the Spanish National Research Council (CSIC) in the development of bioinspired actuation systems for biped robots and lower limb exoskeletons within an European funded project. He is currently a PhD student at the Vrije Universiteit Brussel (VUB), Belgium, where he is the recipient of a doctoral grant from the Research Foundation Flanders (FWO). His work focuses on the design, control and implementation of novel actuation technologies in lower limb exoskeletons for rehabilitation.

Dirk Lefeber received the degree in Study of Civil Engineering at the Vrije Universiteit Brussel and a PhD in Applied Sciences, Vrije Universiteit Brussel, in 1986. Currently he is professor at the dept. of Mechanical Engineering and head of the Robotics and Multibody Mechanics Research Group, Vrije Universiteit Brussel. His research interests are new actuators with adaptable compliance, dynamically balanced robots, robot assistants, rehabilitation robotics and multibody dynamics.

Prof. dr. ir. Bram Vanderborght received his PhD in 2007. The focus of his research was the use of adaptable compliance of pneumatic artificial muscles in the dynamically balanced biped Lucy. In 2006 he performed research on the humanoid robot HRP-2 in AIST, Tsukuba (Japan). From October 2007-April 2010 he worked as post-doc researcher at IIT (Italy). Since October 2009, he is appointed as professor at the VUB. He is member of the Young Academy of the Royal Flemish Academy of Belgium for Science and the Arts. In 2013 he received an ERC Starting Grant. His research interests include cognitive and physical human robot interaction with core technology using variable impedance actuators.