This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

Run-to-run control with Bayesian optimization for soft landing of short-stroke reluctance actuators

Eduardo Moya-Lasheras and Carlos Sagues

***Abstract*—There is great interest in minimizing the impact forces of reluctance actuators during commutations, in order to reduce contact bouncing, acoustic noise and mechanical wear. In this regard, a run-to-run control algorithm is proposed to decrease the contact velocity, by exploiting the repetitive operations of these devices. The complete control is presented, with special focus on the optimization method and the input definition. The search method is based on Bayesian optimization, and several additions are introduced for its application in run-to-run control, e.g. the removal of stored points and the definition of a new acquisition function. Additionally, methods for the input parametrization and dimension reduction are presented. For analysis, Monte Carlo simulations are performed using a dynamic model of a commercial solenoid valve, comparing the proposed search method with two alternatives. Furthermore, the control strategy is validated through experimental testing, using several devices from the same ensemble of solenoid valves.**

I. Introduction

Reluctance actuators are electromechanical devices that rely on reluctance forces to change the position of their movable parts. Simple single-coil short-stroke reluctance actuators, e.g. electromagnetic relays or solenoid valves, are extensively used in on-off switching operations of electrical, hydraulic or pneumatic circuits. However, the range of applications is restricted because of one major drawback: the strong impact at the end of each commutation, which provokes contact bouncing, mechanical wear and acoustic noise. Mitigating those impacts is of great interest, as it potentially extends their service life, makes them operate more quietly, and opens them to applications with more stringent requirements. To decrease contact velocities of the moving parts in reluctance actuators, soft-landing control strategies must be designed.

One of the most straightforward approaches is to design feedback control strategies to track predefined position trajectories [1], [2]. However, in most low-cost short-stroke reluctance actuators, there is no affordable and feasible method to measure the position in real time. To circumvent this limitation, estimators can be designed for the armature position [3], or other position-dependent variables [4]. However, these solutions require precise models for each device that accounts for—among other aspects—its nonlinearities, discrete behavior, time-varying parameters, or measurement errors. Moreover, in many cases, there is a significant unit-to-unit variability among devices from the same ensemble, and the identification of each unit may impose a prohibitive cost.

Run-to-run (R2R) control, in contrast with other learning-type strategies, only requires one evaluation value for each cycle [5]. It is ideal for controlling low-cost reluctance actuators because it does not require the position during operations. Instead, other variables are derived to evaluate each cycle, e.g. the bouncing duration [6] or the sound intensity of the impact.

Measurement-based batch optimization implementations can be explicit or implicit, depending on whether it requires a process model or not [7]. Designing an explicit R2R strategy for reluctance actuators is very challenging. Even if the system dynamics is modeled, the cost function—which maps the decision vector (input) with the evaluation variable (output)—is unknown. Without an explicit definition of the cost function, each evaluation requires querying at a certain decision point and actually commutating the device. As stated by [7], the gradient can be approximated by disturbing the d decision variables. However, the estimated derivatives are very sensitive to noise and need at least $d+1$ function evaluations in each iteration. A better solution for this type of problem, proposed by [6], is based on a derivative-free pattern search method [8], but it still requires many function evaluations in each iteration.

Bayesian optimization is a well-known method of black-box global optimization. In each iteration, it approximates the black-box function with a random process regressor—typically Gaussian [9]—depending on data from previous iterations and, through the maximization of a utility function, selects the following points to be evaluated. It has been proven to be effective in real-time control applications, e.g. maximum power point tracking [10], or altitude optimization of airborne wind energy systems [11].

Besides the optimization algorithm, the input signal selection is critical for the control performance. Furthermore, the signal must be parametrized so it can be modified in each iteration from a limited set of decision variables. While bang-off-bang voltage signals are adequate for soft landing and very easily parametrizable, the dynamics of the current is relatively slow and must be taken into account. Although the current parametrization is more challenging, this paper argues that it is a better choice for input than the voltage, because it is more robust to changes in the temperature.

This paper presents a R2R strategy with Bayesian optimization for soft-landing control of reluctance actuators. Several elements are introduced to the optimization algorithm: the limitation of stored previous data by means of the combination or removal of observations, the adaptability of the search bounds, and the definition of a new acquisition function. Another contribution is the input parametrization, which relates the decision variables to parameters of the dynamic model.

This work was supported in part by the Ministerio de Ciencia Innovación y Universidades, Gobierno de España - European Union under project RTC-2017-5965-6 of subprogram Retos-Colaboración, in part by the Gobierno de Aragón-FSE 2014-20, and in part by project DGA-T45_17R/FSE. (Corresponding author: Eduardo Moya-Lasheras.)

E. Moya-Lasheras and C. Sagues are with the Departamento de Informatica e Ingenieria de Sistemas (DIIS) and the Instituto de Investigacion en Ingenieria de Aragon (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain (email: emoya@unizar.es, csagues@unizar.es).

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

To demonstrate the effectiveness of the proposal, multiple simulations are performed with a dynamic model that fits a commercial solenoid valve. The proposed R2R control is compared with two alternatives. The first one uses a pattern search method, and was previously proposed for bounce reduction of relays and contactors. The second one uses a Nelder-Mead search method, which requires fewer function evaluations in each iteration than the pattern search method. Then, to validate the control applicability in real devices, the R2R strategies are applied to solenoid valves using an experimental setup.

The preliminary ideas of the proposal were presented at the European Control Conference [12]. The most substantial changes are the addition of adaptive search bounds (Section II-D), the definition of a new acquisition function (Section II-E), the model-based input parametrization (Section III), and the new simulations and experiments to support the contributions (Section IV).

II. Run-to-run control with Bayesian optimization

A. Main algorithm

The generalized R2R control algorithm is presented. It must be iterative to account for and exploit the cyclic operations of the actuators. These devices are characterized by having two distinct operation types depending on the motion direction: making and breaking. These two types of operation act alternatively, so a complete commutation cycle consists of one operation of each. The R2R solution (see Algorithm 1) is a loop in which every iteration k comprises three steps.

The first step is the generation of the input signals for the making and breaking operations from a set of d decision variables (Generate Input). Formally, it is expressed as

$$\boldsymbol{u}^k_{\mathbf{m/b}}(t) = \boldsymbol{\mathcal{U}}_{\mathbf{m/b}}\big(\boldsymbol{x}^k_{\mathbf{m/b}}, t\big), \tag{1}$$

where t denotes the time dependence, $\mathrm{m/b}$ serves as a distinction between making (m) and breaking (b) operations, $\boldsymbol{u}^k_{\mathbf{m/b}}(t)$ is the input signal, $\boldsymbol{x}^k_{\mathbf{m/b}} \in \mathbb{R}^d$ is the vector of decision variables, and $\boldsymbol{\mathcal{U}}_{\mathbf{m/b}}$ is the input function. In order to define $\boldsymbol{\mathcal{U}}_{\mathbf{m/b}}$, the input must be parametrized.

The second step is the application of these signals and the observation of the costs $y^k_{\mathrm{m/b}}$. In general, they depend on their corresponding input, which in turn depends on $\boldsymbol{x}^k_{\mathbf{m/b}}$,

$$y^k_{\mathrm{m/b}} = \psi\left(\boldsymbol{x}^k_{\mathbf{m/b}}\right) + \delta^k, \tag{2}$$

where ψ is the black-box cost function, and δ^k denotes the additive effect of system disturbances and measurement errors.

Then, the last step is the optimization process (Search), where the objective is to minimize the costs. The next decision vectors ($\boldsymbol{x}^{k+1}_{\mathbf{m}}$ and $\boldsymbol{x}^{k+1}_{\mathbf{b}}$) are obtained from previous data,

$$\boldsymbol{\mathcal{X}}^k_{\mathbf{m/b}} = \left\{\boldsymbol{x}^i_{\mathbf{m/b}} \mid i = 1, 2, \ldots, k\right\}, \tag{3}$$

$$\mathcal{Y}^k_{\mathrm{m/b}} = \left\{y^i_{\mathrm{m/b}} \mid i = 1, 2, \ldots, k\right\}. \tag{4}$$

Notice that the frequency of the cycles is limited by the computation time of the functions Generate Input and Search. If that time is not small enough, it is necessary to adapt the algorithm to work around this issue, e.g. by commutating the device several times in each iteration without updating the decision vectors, or by computing in parallel the function algorithms for the making and breaking operations.

Algorithm 1 Run-to-run control

1: **Initialize:** $\boldsymbol{x}^1_{\mathbf{m}}$, $\boldsymbol{x}^1_{\mathbf{b}}$
2: **for** $k \leftarrow 1$ **to** num. commutations **do**
3: $\quad \boldsymbol{u}^k_{\mathbf{m}}(t) \leftarrow$ Generate Input($\boldsymbol{x}^k_{\mathbf{m}}$)
4: $\quad \boldsymbol{u}^k_{\mathbf{b}}(t) \leftarrow$ Generate Input($\boldsymbol{x}^k_{\mathbf{b}}$)
5: $\quad$ Apply $\boldsymbol{u}^k_{\mathbf{m}}(t)$ and measure y^k_{m}
6: $\quad$ Apply $\boldsymbol{u}^k_{\mathbf{b}}(t)$ and measure y^k_{b}
7: $\quad \boldsymbol{x}^{k+1}_{\mathbf{m}} \leftarrow$Search($\boldsymbol{\mathcal{X}}^k_{\mathbf{m}}$, $\mathcal{Y}^k_{\mathrm{m}}$)
8: $\quad \boldsymbol{x}^{k+1}_{\mathbf{b}} \leftarrow$Search($\boldsymbol{\mathcal{X}}^k_{\mathbf{b}}$, $\mathcal{Y}^k_{\mathrm{b}}$)
9: **end for**

While the function Generate Input depends on the choice of input signal and the type of actuator, the following description of optimization function SEARCH is generalized.

B. Optimization method

The proposed Search function is based on Bayesian optimization, which is a method for finding the global optimum of an unknown function. From previous data, it builds a stochastic regressor of the black-box function and predicts the output for any point $\boldsymbol{x}$. The selection of the next point to be evaluated $\boldsymbol{x}^{k+1}$ is carried out by maximizing an acquisition function f_{acqn} of the predicted output. The evaluation y^{k+1} is obtained in the next iteration of the R2R Algorithm 1.

The proposed function is described in Algorithm 2. Its inputs are the current point (decision vector $\boldsymbol{x}^k$), which has been obtained in the previous iteration, and its evaluation y^k. Its output is the next point $\boldsymbol{x}^{k+1}$. Some parameters, e.g. the observation noise variance σ^2_{n}, are set as constant. Also, there are some persistent variables, e.g. the number of stored points j, which are changed inside the function but are not required outside of it. Note that these variables are different for each operation type but, for the sake of simplicity, that distinction is omitted. The algorithm can be divided into three steps:

1) **Learning**. Updating the stored points $\boldsymbol{X} \in \mathbb{R}^{d\times j}$ and their evaluations $\boldsymbol{Y} \in \mathbb{R}^{1\times j}$ by the addition of the kth decision vector $\boldsymbol{x}^k$ and its cost y^k. The variance of the last observation σ^2_{n} is added to the covariance $\boldsymbol{\Sigma}$.
2) **Data size constraining** ($j \leq j_{\max}$). Observations are merged or removed if necessary. Furthermore, the search space is modified in each iteration. These processes are further discussed in Subsection II-D.
3) **Acquisition**. Selection of next decision vector $\boldsymbol{x}^{k+1}$ by maximizing an acquisition function, given the previous data ($\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{\Sigma}$). The search is restricted between the lower bound $\boldsymbol{x}_{\mathbf{lb}}$ and the upper bound $\boldsymbol{x}_{\mathbf{ub}}$. The proposed acquisition function is defined in Subsection II-E.

C. Prior and posterior distributions

The selected model for regression is the Gaussian process. It is the most popular one in the context of Bayesian optimization because it only requires simple algebraic operations to

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

Algorithm 2 Optimization

function SEARCH($\boldsymbol{x}^k$, y^k)
  Constant: σ_n^2, d, j_max
  Persistent: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{\Sigma}$, j
  ▷ Learning
  $j \leftarrow j+1$
  $(\boldsymbol{X}_j,\, Y_j,\, \boldsymbol{\Sigma}_{j,j}) \leftarrow \left(\boldsymbol{x}^k,\, y^k,\, \sigma_\mathrm{n}^2\right)$
  ▷ Data size constraining
  $(\boldsymbol{X},\, \boldsymbol{Y},\, \boldsymbol{\Sigma},\, j) \leftarrow$ MERGE($\boldsymbol{X},\, \boldsymbol{Y},\, \boldsymbol{\Sigma},\, j$)
  $(\boldsymbol{x}_\mathbf{lb},\, \boldsymbol{x}_\mathbf{ub}) \leftarrow$ BOUNDS($\boldsymbol{X},\, \boldsymbol{Y},\, \boldsymbol{\Sigma}$)
  if $j > j_\mathrm{max}$ **then**
    $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\Sigma}, j) \leftarrow$ REMOVE($\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\Sigma}, j, \boldsymbol{x}_\mathbf{lb}, \boldsymbol{x}_\mathbf{ub}$)
  end if
  ▷ Acquisition
  $\boldsymbol{x}^{k+1} \leftarrow \arg\max_{\boldsymbol{x}_\mathbf{lb} \le \boldsymbol{x} \le \boldsymbol{x}_\mathbf{ub}} f_\mathrm{acqn}\left(\boldsymbol{x} \mid \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\Sigma}\right)$
  return $\boldsymbol{x}^{k+1}$
end function

determine the corresponding posterior distribution. In general, it is completely specified by a mean $m(\boldsymbol{x})$ and covariance function or kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ [9],

$$f(\boldsymbol{x}) \sim \mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right). \tag{5}$$

For convenience, m is assumed to be constant. The chosen covariance function is squared exponential,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{x}'\right)^\mathsf{T} \mathrm{diag}(\boldsymbol{l})^{-2}\left(\boldsymbol{x}-\boldsymbol{x}'\right)\right), \tag{6}$$

where σ_f^2 is the characteristic variance and $\mathrm{diag}(\boldsymbol{l}) \in \mathbb{R}^{d\times d}$ is a diagonal matrix with the lengthscales for each dimension.

In a given iteration, we have the training outputs $\boldsymbol{Y} = f(\boldsymbol{X}) + \boldsymbol{\varepsilon}$. The output noise $\boldsymbol{\varepsilon}$ is an independently distributed Gaussian random vector whose covariance is the diagonal matrix $\boldsymbol{\Sigma}$. Given the properties of Gaussian processes, the joint distribution of $\boldsymbol{Y}$ and an output f for an arbitrary $\boldsymbol{x}$ is multivariate normal,

$$\begin{bmatrix}\boldsymbol{Y}^\mathsf{T} \\ f\end{bmatrix} \sim \mathcal{N}\left(m\,\boldsymbol{J}_{j+1,1},\, \begin{bmatrix}\boldsymbol{K}+\boldsymbol{\Sigma} & \boldsymbol{k} \\ \boldsymbol{k}^\mathsf{T} & k(\boldsymbol{x},\boldsymbol{x})\end{bmatrix}\right), \tag{7}$$

where $\boldsymbol{J}$ denotes an all-ones matrix, and the covariance matrices $\boldsymbol{K} \in \mathbb{R}^{j\times j}$ and $\boldsymbol{k} \in \mathbb{R}^{j\times 1}$ are

$$\boldsymbol{K}_{i,i'} = k(\boldsymbol{X}_i, \boldsymbol{X}_{i'}), \quad \boldsymbol{k}_i = k(\boldsymbol{X}_i, \boldsymbol{x}), \quad \forall i, i' \le j. \tag{8}$$

The posterior predictive distribution for f is also Gaussian,

$$f \mid \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{x} \sim \mathcal{N}(\mu,\, \sigma^2), \tag{9}$$

where the mean μ and variance σ^2 depend on previous data,

$$\mu = \mu(\boldsymbol{x}) = \left(\boldsymbol{Y} - m\,\boldsymbol{J}_{1,j}\right)\left(\boldsymbol{K}+\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{k} + m, \tag{10}$$

$$\sigma^2 = \sigma^2(\boldsymbol{x}) = k(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}^\mathsf{T}\left(\boldsymbol{K}+\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{k}. \tag{11}$$

D. Data size constraining

For the application of the optimization method for cycle-to-cycle learning type control, it is imperative to constrain the size of stored data in order to prevent the ceaseless increase of computational requirements. For that purpose, three adjustments (functions in Algorithm 2) are introduced:

Algorithm 3 Adaptive search bounds

function BOUNDS($\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{\Sigma}$)
  Constant: $\boldsymbol{x}_\mathbf{lb,abs}$, $\boldsymbol{x}_\mathbf{ub,abs}$, α, c_shrink
  Persistent: $\boldsymbol{L}$, $\boldsymbol{D}_\boldsymbol{x}$, $\boldsymbol{x}_\mathbf{best}^k$
  $\boldsymbol{x}_\mathbf{best}^{k-1} \leftarrow \boldsymbol{x}_\mathbf{best}^k$
  $\boldsymbol{x}_\mathbf{best}^k \leftarrow \boldsymbol{x}_i$ s.t. $i = \arg\min_i \boldsymbol{\mu}(\boldsymbol{X}_i)$
  $\boldsymbol{D}_\boldsymbol{x} = \alpha\,\boldsymbol{D}_\boldsymbol{x} + (1-\alpha)\left(\boldsymbol{x}_\mathbf{best}^k - \boldsymbol{x}_\mathbf{best}^{k-1}\right)$
  $\boldsymbol{L} \leftarrow \mathrm{sat}_{\boldsymbol{L}_\mathbf{min}}^{\boldsymbol{L}_\mathbf{max}}\left(c_\mathrm{shrink}\left(\boldsymbol{L} + |\boldsymbol{D}_\boldsymbol{x}|\right)\right)$
  $\boldsymbol{x}_\mathbf{lb} \leftarrow \max(\boldsymbol{x}_\mathbf{lb,abs},\ \boldsymbol{x}_\mathbf{best}^k - \boldsymbol{L})$
  $\boldsymbol{x}_\mathbf{ub} \leftarrow \min(\boldsymbol{x}_\mathbf{ub,abs},\ \boldsymbol{x}_\mathbf{best}^k + \boldsymbol{L})$
  return $\boldsymbol{x}_\mathbf{lb}$, $\boldsymbol{x}_\mathbf{ub}$
end function

1) MERGE. If several observations are performed for the same input, there is no need to store them separately. Those cost evaluations can be merged by using Bayesian inference and the equivalent cost mean and variance are obtained [12].
2) BOUNDS. The search space is made adaptive, shrinking or expanding (see Algorithm 3). The space expansion in an iteration k is related to the variation of $\boldsymbol{x}_\mathbf{best}$ from the previous iteration $k-1$. First, the current one $\boldsymbol{x}_\mathbf{best}^k$ is selected based on its mean value (see (10)). Secondly, the variations are filtered with an exponentially moving average (EMA), where $\alpha \in (0,1)$ is the EMA coefficient. Thirdly, the bound length vector $\boldsymbol{L} \in \mathbb{R}^d$ is updated in each iteration, shrinking if ${\boldsymbol{D}_\boldsymbol{x}}^k$ is small and expanding otherwise (where $c_\mathrm{shrink} \in (0,1)$ is the shrinkage coefficient, $|{\boldsymbol{D}_\boldsymbol{x}}^k|$ is the element-wise absolute value, and $\mathrm{sat}_{\boldsymbol{L}_\mathbf{max}}^{\boldsymbol{L}_\mathbf{min}}$ denotes an element-wise saturation function between the chosen $\boldsymbol{L}_\mathbf{min}$ and $\boldsymbol{L}_\mathbf{max}$). Finally, the bounds $\boldsymbol{x}_\mathbf{lb}$ and $\boldsymbol{x}_\mathbf{ub}$ are calculated around the $\boldsymbol{x}_\mathbf{best}^k$, ensuring that they do not surpass the absolute ones $\boldsymbol{x}_\mathbf{lb,abs}$ and $\boldsymbol{x}_\mathbf{ub,abs}$.
3) REMOVE. Merging observations does not guarantee that the size of data history is bounded. Thus, the third measure consists in a two-step removal of points, in the case that the number of stored points j surpasses a chosen limit j_max. For the first step, note that the effect of points that are far away from the bounded space—depending on the characteristic length scales $\boldsymbol{l}$ (see (6))—can be considered negligible. Thus, any point $\boldsymbol{X}_i$ that does not meet

$$\boldsymbol{x}_\mathbf{lb} - 3\,\boldsymbol{l} \le \boldsymbol{X}_i \le \boldsymbol{x}_\mathbf{ub} + 3\,\boldsymbol{l} \tag{12}$$

is removed. The shrinkage and removal of distant points are especially useful for high-dimensional inputs, in which the limited number of stored data is not enough to generate a regressor of the entire search space. However, in any case, the data size constraining is still not guaranteed. Then, as the second step, a removal criterion is defined to ensure that $j \le j_\mathrm{max}$. In the previous work [12], a removal criterion is proposed. The objective is to find the index i which minimizes the increment of differential entropy in $\boldsymbol{X}_i$ due to its removal. After some algebraic simplifications, the

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

index to be removed is obtained as

$$\arg\min_i \frac{\sigma^2(\boldsymbol{X}_i)}{\Sigma_{i,i}} \tag{13}$$

where, for each point $\boldsymbol{X}_i$, the posterior variance σ^2 is calculated from (11).

E. Acquisition

The final part of the algorithm is the selection of the next point $\boldsymbol{x}^{k+1}$ to be evaluated, which must trade off between obtaining the most information of the function (exploration) and attempting to minimize it (exploitation). As the predicted cost f is a random variable, the selection of $\boldsymbol{x}^{k+1}$ must be carried out by the maximization of an acquisition function dependent on μ and σ^2, defined in equations (10) and (11). The previous work [12] proposes a generalization of the expected improvement: the expected net improvement. First, the net improvement over the remaining Δk commutations is defined as a piecewise function of the random variable $f \sim \mathcal{N}(\mu, \sigma^2)$,

$$I_{\text{net}}(f) = \begin{cases} \Delta k\,(\mu_{\min} - f), & \text{if } f \le \mu_{\min}, \\ \mu_{\min} - f, & \text{if } f > \mu_{\min}, \end{cases} \tag{14}$$

where $\mu_{\min} = \min_i \mu(\boldsymbol{X}_i)$ is the best observation so far. On the one hand, an improvement over $\mu_{\min}$ means a potential improvement for the remaining Δk commutations. On the other hand, a worsening over $\mu_{\min}$ only affects the next commutation, because in the following one it would be possible to commute with an expected cost of $\mu_{\min}$.

Then, the expected net improvement is derived, which can be expressed in relation to the expected improvement $\mathrm{E}[I(f)]$ —which is one of the most common acquisition functions—,

$$\mathrm{E}[I_{\text{net}}(f)] = \mu_{\min} - \mu + (\Delta k - 1)\,\mathrm{E}[I(f)]. \tag{15}$$

Note that, if $\Delta k = 1$, the maximization of $\mathrm{E}[I_{\text{net}}(f)]$ is equivalent to the minimization of μ. As Δk decreases, the acquisition favors exploitation over exploration, which is the intended behavior. Also, notice that as Δk increases, $\mathrm{E}[I_{\text{net}}(f)]/\Delta k$ tends to $\mathrm{E}[I(f)]$, which would correspond to a regret-free optimization. Moreover, if the number of commutations is not known, Δk can be set as an expectation.

In this control problem, the objective is to minimize a cost related to absolute or non-negative values, e.g. contact velocities, impact sound intensities, or bouncing durations. Taking that into account, the acquisition function can be further improved. For convenience, f is kept as a Gaussian random variable given by (9) and an auxiliary variable f_{s} is defined by saturating f and ensuring its non-negativity,

$$f_{\text{s}} = \max(f, 0). \tag{16}$$

Substituting f with f_{s} in (14), the net improvement of the saturated f_{s} is

$$I_{\text{net}}(f_{\text{s}}) = \begin{cases} \Delta k\,\mu_{\min}, & \text{if } f \le 0, \\ \Delta k\,(\mu_{\min} - f), & \text{if } 0 < f \le \mu_{\min}, \\ \mu_{\min} - f, & \text{if } f > \mu_{\min}, \end{cases} \tag{17}$$

and its expected value, which is the proposed acquisition function f_{acqn}, can be expressed in terms of the already defined $\mathrm{E}[I_{\text{net}}(f)]$,

$$\begin{aligned} f_{\text{acqn}} &= \mathrm{E}[I_{\text{net}}(f_{\text{s}})] \\ &= \mathrm{E}[I_{\text{net}}(f)] - \Delta k\,\sigma\left(\varphi\left(-\frac{\mu}{\sigma}\right) - \frac{\mu}{\sigma}\,\Phi\left(-\frac{\mu}{\sigma}\right)\right), \end{aligned} \tag{18}$$

where φ and Φ are the standard normal probability and cumulative distribution functions, respectively. Note that the subtracting term is always positive, i.e. $\mathrm{E}[I_{\text{net}}(f_{\text{s}})] < \mathrm{E}[I_{\text{net}}(f)]$, and increases with respect to the variance, prioritizing exploitation over exploration.

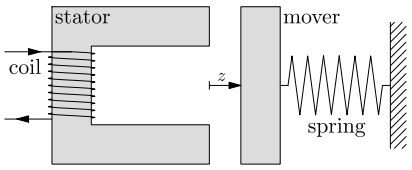


Fig. 1. Schematic diagram of a reluctance actuator.

III. Input Parametrization

One of the critical points of a R2R strategy is the selection of the input and its parametrization.

A. System model

To justify the input selection, the dynamical system must be characterized and discussed. Thus, a generalized dynamic model for a reluctance actuator is presented. In Fig. 1, the device is represented in its most basic form. The magnetic core is divided into a fixed part (stator) and a movable part (mover). The air gap between them is dependent on the mover position. Low-cost actuators typically have a single coil and no permanent magnets, which implies that the magnetic force has only one direction, attracting the mover toward the stator. In order to separate the mover from the stator, these devices rely on passive elastic forces. The armature position is restricted between a lower limit ($z_{\min}$) and an upper limit ($z_{\max}$).

Both the mechanical and electromagnetic dynamics must be modeled. To shorten the expressions, the time t dependency and the iteration k distinction are omitted. With respect to the mechanical subsystem, the motion dynamics is given by Newton's second law, with three forces,

$$m_{\text{mov}}\,a = F_{\text{e}}(z) + F_{\text{f}}(z, v) + F_{\text{mag}}(z, \phi), \tag{19}$$

where z, v and a are the position, velocity and acceleration of the armature; F_{e}, F_{f}, F_{mag} are the elastic, friction and magnetic forces; ϕ is the magnetic flux; and m_{mov} is the moving mass. The only force that can be controlled—albeit indirectly—is F_{mag}, which depends on the derivative of the gap reluctance $\mathcal{R}_{\text{g}}$ and the magnetic flux ϕ [13],

$$F_{\text{mag}} = -\mathcal{R}'_{\text{g}}(z)\,\phi^2/2, \qquad \mathcal{R}'_{\text{g}}(z) = \partial\mathcal{R}_{\text{g}}(z)/\partial z. \tag{20}$$

Then, the dynamics of ϕ can be modeled using as input the coil current i_{coil} or voltage v_{coil}. The relation between the magnetic flux and the current can be expressed in terms of the reluctance, given Ampère's circuital law,

$$N\,i_{\text{coil}} + i_{\text{eddy}} = \big(\mathcal{R}_{\text{c}}(\phi) + \mathcal{R}_{\text{g}}(z)\big)\,\phi, \tag{21}$$

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

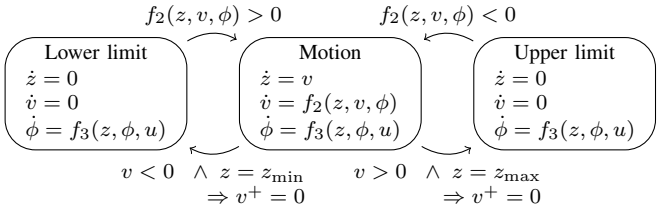


Fig. 2. Diagram of the hybrid automaton.

where N is the number of coil turns, $\mathcal{R}_\mathrm{c}$ is the core reluctance, and i_eddy is the net eddy current through the core. Assuming a uniform magnetic flux within the section, it can be shown [14], [15] that i_eddy must be proportional to the time derivative of the magnetic flux,

$$i_\mathrm{eddy} = -k_\mathrm{eddy}\,\dot{\phi}. \tag{22}$$

The system dynamics can be defined through a state-space representation, with three state variables: the position z, the velocity v and the magnetic flux ϕ. As the motion is constrained, z and v must be static if the armature reaches one of the two limits. Thus, the system is modeled with a hybrid automaton (Fig. 2), with three discrete modes, their corresponding guard conditions, and a reset function ($v^+ = 0$) in each transition from the motion mode. Most commonly, the voltage v_coil is treated as the system input u, because it is directly supplied to the device. In that case, the dynamics of the magnetic flux is given by the electrical circuit equation,

$$v_\mathrm{coil} = R\,i_\mathrm{coil} + N\,\dot{\phi}. \tag{23}$$

Then, substituting (21) into (23) and isolating $\dot{\phi}$, its expression is derived,

$$\dot{\phi} = -\frac{R\left(\mathcal{R}_\mathrm{g}(z) + \mathcal{R}_\mathrm{c}(\phi)\right)}{N^2 + R\,k_\mathrm{eddy}}\,\phi + \frac{N}{N^2 + R\,k_\mathrm{eddy}}\,v_\mathrm{coil}. \tag{24}$$

Notice the strong dependence on R, which can change significantly with the temperature. Thus, as a more robust alternative, this paper proposes to use the current signal as the input, $u = i_\mathrm{coil}$, which makes the dynamics of ϕ invariant with respect to R. Then, the dynamic functions f_2 and f_3 (see Fig. 2) are derived from (19) to (22),

$$f_2(z,v,\phi) = \frac{F_\mathrm{e}(z) + F_\mathrm{f}(z,v) - \mathcal{R}'_\mathrm{g}(z)\,\phi^2/2}{m_\mathrm{mov}}, \tag{25a}$$

$$f_3(z,\phi,i_\mathrm{coil}) = -\frac{\mathcal{R}_\mathrm{c}(\phi) + \mathcal{R}_\mathrm{g}(z)}{k_\mathrm{eddy}}\,\phi + \frac{N}{k_\mathrm{eddy}}\,i_\mathrm{coil}. \tag{25b}$$

B. Current parametrization

The input function (1) must be defined, relating the decision vector $\boldsymbol{x}$ to the continuous input signal $u(t)$. Thus, the input must be parametrized. On the one hand, using the current as input is advantageous, as discussed in Section III-A. On the other hand, its parametrization is more challenging. Regarding the voltage parametrization, bang-off-bang voltage signals can be constructed from the time intervals, which would act as the decision variables. In contrast, the current dynamics is slower, and must be taken into account. Therefore, this section presents a method of current parametrization based on the dynamic model. First, desired soft-landing position trajectories are predefined for both operations. These can be established in any way, as long as they are feasible, taking into account the physical limitations of the system. Our proposal is to use, for each operation type, the predefined position trajectory $z(t)$ as reference, modify it with computationally efficient linear transformations, and calculate the current signal $i_\mathrm{coil}(t)$ for a given set of parameters in each iteration.

The first parameters that are considered for decision variables are the initial and final position values, z_0 and z_f. Then, the nominal position trajectory z is linearly transformed into $\hat{z}$, so the boundary conditions $\hat{z}(0) = z_0$, $\hat{z}(t_\mathrm{f}) = z_\mathrm{f}$ are met,

$$\hat{z} = C_z\left(z - z(0)\right) + z_0, \qquad C_z = \frac{z_\mathrm{f} - z_0}{z(t_\mathrm{f}) - z(0)}, \tag{26}$$

The position derivatives up to the jerk $\dot{a}$ are necessary as well, so they must also be modified,

$$\hat{v} = C_z\,v, \qquad \hat{a} = C_z\,a, \qquad \hat{\dot{a}} = C_z\,\dot{a}. \tag{27}$$

Next, the current is obtained from (21) and (22), which depend on ϕ and $\dot{\phi}$, respectively. On the one hand, the variable ϕ can be isolated from (19),

$$\phi = \sqrt{\frac{2\left(F_\mathrm{e}(\hat{z}) + F_\mathrm{f}(\hat{z},\hat{v}) - m_\mathrm{mov}\,\hat{a}\right)}{\mathcal{R}'_\mathrm{g}(\hat{z})}}. \tag{28}$$

On the other hand, $\dot{\phi}$ can be obtained by calculating the time derivative of (19) and isolating it, assuming that the elastic and friction functions F_e and F_f are differentiable,

$$\dot{\phi} = \frac{1}{\mathcal{R}'_\mathrm{g}(\hat{z})\,\phi}\left(\frac{\partial F_\mathrm{e}}{\partial z}(\hat{z})\,\hat{v} + \frac{\partial F_\mathrm{f}}{\partial z}(\hat{z},\hat{v})\,\hat{v} + \frac{\partial F_\mathrm{f}}{\partial v}(\hat{z},\hat{v})\,\hat{a} - m_\mathrm{mov}\,\hat{\dot{a}} - \frac{1}{2}\,\mathcal{R}''_\mathrm{g}(\hat{z})\,\hat{v}\,\phi^2\right), \qquad \mathcal{R}''_\mathrm{g}(z) = \frac{\partial^2 \mathcal{R}_\mathrm{g}(x)}{\partial z^2}. \tag{29}$$

It is possible that, for a poorly chosen set of parameters and position trajectory, the radicand in (28) is negative in some interval. That would mean that the required magnetic force is positive (see 20), which is not physically possible. In that case, as the closest feasible approximation, ϕ and $\dot{\phi}$ are set to zero. Taken that into account, the input definition is finally expressed as

$$i_\mathrm{coil} = \begin{cases} \frac{\mathcal{R}_\mathrm{c}(\phi) + \mathcal{R}_\mathrm{g}(\hat{z})}{N}\,\phi + \frac{k_\mathrm{eddy}}{N}\,\dot{\phi}, & \text{if } \phi^2 > 0, \\ 0, & \text{if } \phi^2 \le 0. \end{cases} \tag{30}$$

Thus, the current signal is a function of the position trajectory z, its derivatives v, a, $\dot{a}$, and the model parameters, including z_0 and z_f. On the one hand, the position and its derivatives are established prior to the control. Moreover, depending on the case, some of the parameters are considered constants, because they have been identified accurately and their variability between units is negligible. On the other hand, the remaining parameters are assumed unknown and will be treated as the decision variables, which can be modified in each cycle. As their magnitudes may differ greatly, it is convenient to normalize each one. Consider $\boldsymbol{\theta}$ to be the vector of unknown parameters. Assuming that each parameter is bounded such

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

that $\theta_i \in [\theta_i^-,\ \theta_i^+]$, the normalization is achieved through a linear transformation,

$$x_i = \frac{2\,\theta_i - \theta_i^+ - \theta_i^-}{\theta_i^+ - \theta_i^-}. \tag{31}$$

Thus, the decision variables are bounded $x_i \in [-1,\ 1]$.

Note that this resembles an online identification process, as the parameter-related decision variables are optimized in an iterative fashion. However, the parameters are not optimized to reduce estimation errors, but solely to improve the performance, i.e. minimize the defined cost. Thus, the parameters may not converge to their real values if it is not necessary for the control. This is not a limitation but a design choice: the objective is to minimize a certain cost, not to identify the system. Note also that, in a R2R application, the data is very limited (only one cost value per iteration), so it would not be possible to guarantee that the model parameters converge to the real values. Furthermore, the parameter-related decision variables are optimized independently for each operation, so they may be able to correct certain phenomena which are not taken into account by the dynamic model and act differently in each operation type, e.g. magnetic hysteresis.

Note that the input has only been defined for the operations, i.e. during motion. Then, the signal for the complete cycle must be constructed. After each operation, steady current values must be applied to maintain the actuator in its position ($z_{\min}$ after the making operation and $z_{\max}$ after the breaking operation). Also, there needs to be feasible transitions between the steady and the operation currents, and vice versa.

C. Sensitivity analysis and dimension reduction

The convergence rate of the optimization process is strongly dependent on the dimension of the decision vector. Thus, it is highly recommendable to remove redundant decision variables whose effect on the input can be replicated through a combination of other variables. Some of these redundancies may be easy to spot through examination of the input function (see (28) to (30)). In that case, through some change of variables, an equivalent input function with fewer parameters can be derived. To ensure that there are no more redundant decision variables, the input function (1) can be interpreted as a discrete-time system, where the decision variables constitute the state vector,

$$\begin{cases} \boldsymbol{x}(t_{i+1}) &= \boldsymbol{x}(t_i), \\ u(t_i) &= \mathcal{U}(\boldsymbol{x}(t_i), t_i). \end{cases} \tag{32}$$

Then, its local observability matrix $\mathcal{O} \in \mathbb{R}^{d\times d}$ for consecutive time samples $t_1, t_2, \ldots, t_d$ is constructed, whose (i,j)th entry is $\mathcal{O}_{i,j} = \partial\mathcal{U}(\boldsymbol{x}, t_i)/\partial x_j$. If the matrix is full-rank—except for singularities—, the effect of the decision variables x_i are independent. Even so, it may be still possible to approximate the effect of one or several decision variables with a combination of the remaining ones. This analysis is very complex, due to the highly nonlinear input function (21). Thus, we propose to perform a local sensitivity analysis, at $\boldsymbol{x} = \boldsymbol{0}$, which is equivalent to assuming the errors of the first-order Taylor series negligible,

$$\mathcal{U}(\boldsymbol{x}, t) - \mathcal{U}(\boldsymbol{0}, t) \approx \sum_{i=1}^{d} \left.\frac{\partial\mathcal{U}(\boldsymbol{x}, t)}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{0}} x_i = \nabla\mathcal{U}(t)\,\boldsymbol{x}. \tag{33}$$

The sensitivities are the partial derivatives of $\mathcal{U}$, which can be calculated for the nominal case $\boldsymbol{x} = \boldsymbol{0}$ for every t. Just by comparing these partial derivatives, it is possible to determine which pair of parameters have very similar effects on the current, or which parameters have a negligible effect. Still, there may be other cases of near collinearity between sensitivities which are not apparent. One relatively simple way to analyze it is to sample the time $(t_1, t_2, \ldots)$ and create a Jacobian matrix such that each ith row is $\nabla\mathcal{U}(t_i)$. Then, the rank of the matrix with a chosen tolerance can be calculated and, if the rank is not full, the decision variables to be removed can be detected with a singular value decomposition. However, this method does not take into account the bounds of each parameter. Instead, we propose the following method:

A decision variable x_i is a candidate for removal if the variation of u due to x_i is nearly the same as a variation of u with a certain $\boldsymbol{x}$, where its ith component is zero. Formally, the set of all possible solutions is

$$\boldsymbol{\mathcal{X}}_{\setminus i} = \left\{\boldsymbol{x}\,|\,x_i = 0 \wedge \left(\nabla\mathcal{U}_i(t) - \nabla\mathcal{U}(t)\,\boldsymbol{x}\right)^2 \leq {\mathcal{U}_{\text{tol}}}^2\,\forall t\right\}. \tag{34}$$

where $\mathcal{U}_{\text{tol}}$ is an arbitrary tolerance. If $\boldsymbol{\mathcal{X}}_{\setminus i}$ is empty, the effect of the decision variable x_i is not redundant and thus it should not be removed. Otherwise, x_i is a candidate for removal. As the size $\boldsymbol{\mathcal{X}}_{\setminus i}$ may be larger than one—i.e. multiple solutions—we select the one with the smallest euclidean norm,

$$\boldsymbol{x}_{\setminus i} = \underset{\boldsymbol{x}\,\in\,\boldsymbol{\mathcal{X}}_{\setminus i}}{\arg\min} \|\boldsymbol{x}\|_2. \tag{35}$$

This is performed for every i. The decision variable x_i whose $\boldsymbol{x}_{\setminus i}$ has the minimum euclidean norm is removed. Note that, due to the removal of an input, the allowed variations of u are reduced. To be able to vary u as much as before the removal of x_j, the bounds $\boldsymbol{\theta}^+$ and $\boldsymbol{\theta}^-$ of the remaining parameters must be augmented depending on $\boldsymbol{x}_{\setminus j}$. Note that $\nabla\mathcal{U}$ is proportional to $(\boldsymbol{\theta}^+ - \boldsymbol{\theta}^-)$, so it must also be augmented. Note also that the inputs are still normalized ($x_i \in [-1, 1]$). The complete method is presented in Algorithm 4, which must be performed for the making and breaking inputs separately.

IV. Analysis

In this section, simulated and experimental results are presented and discussed.

A. Solenoid valve

The device used in the tests is a low-cost linear-travel solenoid valve (Fig. 3). The core is cylindrically symmetrical, with a fixed part and an armature. The coil current generates a magnetic flux through the core parts and the air gap between them. The spring force tends to open the gap, whereas the magnetic force points in the opposite direction.

Before specifying the input function, the model must be completely defined. Besides for the input parametrization,

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

Algorithm 4 Dimension reduction

loop
  for $i \leftarrow 1$ **to** d **do**
    Calculate $\boldsymbol{\mathcal{X}}_{\setminus i}$ ▷ Equation (34)
  end for
  if $\boldsymbol{\mathcal{X}}_{\setminus i} = \emptyset \ \forall i$ **then**
    break loop
  end if
  Calculate $\boldsymbol{x}_{\setminus i}$ ▷ Equation (35)
  $j \leftarrow \arg\min_i \|\boldsymbol{x}_{\setminus i}\|_2$
  for $i \leftarrow 1$ **to** d **do**
    $\Delta\theta \leftarrow (\theta_i^+ - \theta_i^-)\,|(x_{\setminus j})_i|$
    $\theta_i^+ \leftarrow \theta_i^+ + \Delta\theta/2$
    $\theta_i^- \leftarrow \theta_i^- - \Delta\theta/2$
    $\nabla\mathcal{U} \leftarrow \nabla\mathcal{U}\,\big(1 + |(x_{\setminus j})_i|\big)$
  end for
  $\big(\boldsymbol{x}, \boldsymbol{\theta}^+, \boldsymbol{\theta}^-\big) \leftarrow$ REMOVE$(x_j, \theta_j^+, \theta_j^-)$
  $d \leftarrow d - 1$
end loop

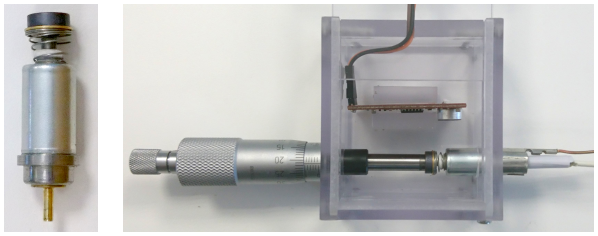


Fig. 3. Linear-travel short-stroke solenoid valve (left) and experimental setup with the valve, a micrometer to limit the maximum gap and an electret microphone to measure the impact noise (right).

the system model is useful for performing analyses through simulations, as presented in Section IV-C. The elastic force is modeled as an ideal spring, and the friction as a viscous force,

$$F_\mathrm{e} = k_\mathrm{sp}\,(z_\mathrm{sp} - z), \qquad F_\mathrm{f} = -c_\mathrm{f}\,v, \tag{36}$$

being k_sp the spring constant, z_sp the spring resting position, and c_f the friction coefficient.

In order to consider the rarely negligible magnetic saturation phenomenon, the core reluctance $\mathcal{R}_\mathrm{c}$ is modeled as a parametric expression derived from the Fröhlich-Kenelly equation,

$$\mathcal{R}_\mathrm{c}(\phi) = \frac{k_1}{1 - k_2\,\phi}. \tag{37}$$

In contrast, the gap reluctance $\mathcal{R}_\mathrm{g}$ and its derivatives are obtained from lookup tables, whose data were calculated from a finite element model [16]. In Fig. 4, $\mathcal{R}_\mathrm{g}$ and $\mathcal{R}'_\mathrm{g}$ are represented.

Having defined those functions, the dynamical system is fully characterized. The model parameters are summarized in Table I, along with their nominal values. Then, the input is parametrized following the process presented in Section III-B. First, the desired position trajectories have been optimally calculated following the procedure presented in [17], using the nominal parameter values and setting the final time to $t_\mathrm{f} = 3.5$ ms. Fig. 5 depicts the desired position and the nominal current signals for the making and breaking operations.

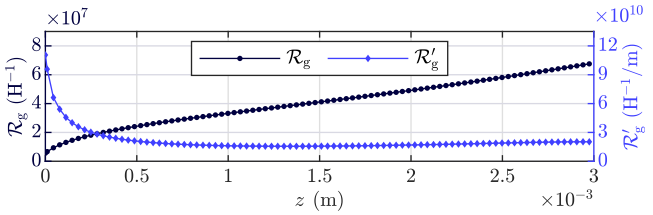


Fig. 4. Gap reluctance and its derivative with respect to the gap length.

TABLE I
PARAMETERS OF THE SOLENOID VALVE

Parameter	Nominal value		Parameter	Nominal value	
m_mov	1.6×10^{-3}	kg	k_eddy	1630	Ω^{-1}
k_sp	61.8	N/m	k_1	4.41×10^6	H^{-1}
z_sp	0.0192	m	k_2	3.8×10^4	Wb^{-1}
c_f	0.8	Ns/m	z_min	4×10^{-4}	m
N	1200		z_max	1.4×10^{-3}	m

Secondly, the input function is defined. To simplify the final expression, auxiliary variable $\hat{\phi} = \phi/\sqrt{m_\mathrm{mov}}$ and its derivative are used,

$$i_\mathrm{coil} = \begin{cases} \frac{k_1/(1-\hat{k}_2\,\hat{\phi}) + \mathcal{R}_\mathrm{g}(\hat{z})}{\hat{N}}\,\hat{\phi} + \frac{k_\mathrm{eddy}}{\hat{N}}\,\dot{\hat{\phi}}, & \hat{\phi}^2 > 0, \\ 0 & \hat{\phi}^2 \le 0, \end{cases} \tag{38}$$

being $\hat{N} = N/\sqrt{m_\mathrm{mov}}$ and $\hat{k}_2 = k_2\sqrt{m_\mathrm{mov}}$ auxiliary parameters. The auxiliary variable $\hat{\phi}$ can be derived from (28),

$$\hat{\phi} = \frac{\phi}{\sqrt{m_\mathrm{mov}}} = \sqrt{\frac{2\,(\hat{k}_\mathrm{sp}\,(z_\mathrm{sp} - \hat{z}) - \hat{c}_\mathrm{f}\,\hat{v} - \hat{a})}{\mathcal{R}'_\mathrm{g}(\hat{z})}}, \tag{39}$$

where $\hat{k}_\mathrm{sp} = k_\mathrm{sp}/m_\mathrm{mov}$ and $\hat{c}_\mathrm{f} = c_\mathrm{f}/m_\mathrm{mov}$. Equivalently, $\dot{\hat{\phi}}$ can be obtained from (29),

$$\dot{\hat{\phi}} = \frac{\dot{\phi}}{\sqrt{m_\mathrm{mov}}} = -\frac{\hat{k}_\mathrm{sp}\,\hat{v} + \hat{c}_\mathrm{f}\,\hat{a} + \hat{\hat{a}} + \mathcal{R}''_\mathrm{g}(\hat{z})\,\hat{v}\,\hat{\phi}^2/2}{\mathcal{R}'_\mathrm{g}(\hat{z})\,\hat{\phi}}. \tag{40}$$

Thus, the input depends on these parameters:

$$\boldsymbol{\theta} = \begin{bmatrix} z_0 & z_\mathrm{f} & \hat{k}_\mathrm{sp} & z_\mathrm{sp} & \hat{c}_\mathrm{f} & \hat{N} & k_1 & \hat{k}_2 & k_\mathrm{eddy} \end{bmatrix}^\mathsf{T}, \tag{41}$$

which then are normalized (see (31)) to obtain the decision vector $\boldsymbol{x}$. The parameter bounds are set to $\pm 10\%$ of the nominal values θ_i^nom. Specifically,

$$\theta_i^\pm = \theta_i^\mathrm{nom} \pm 0.1\,\Delta z^\mathrm{nom}, \quad \Delta z^\mathrm{nom} = z_\mathrm{max}^\mathrm{nom} - z_\mathrm{min}^\mathrm{nom}, \tag{42}$$

if θ_i is z_0 or z_f. Otherwise,

$$\theta_i^\pm = \theta_i^\mathrm{nom} \pm 0.1\,\theta_i^\mathrm{nom}. \tag{43}$$

Note there was a redundant degree of freedom related to m_mov, and has been removed thanks to the change of variables (38) and the definition of auxiliary parameters (41). Theoretically, there are no other redundancies, because the observability matrix $\mathcal{O}$—as defined in Section III-C—is full-rank. However, there may be some decision variables whose effect can be approximated from a combination of the others. Thus, in order to detect other candidates for removal, we use Algorithm 4, which was presented in Section III-C. The tolerance is set to $\mathcal{U}_\mathrm{tol} = 10^{-3}$ A. Then, the decision

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

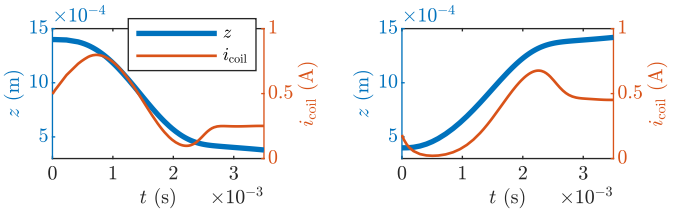


Fig. 5. Desired position trajectories and nominal current signals for the making (left) and breaking (right) operations.

variables to be removed are the ones related to $\hat{k}_{\text{sp}}$, k_1 and k_{eddy}, for both operation types. Therefore, those parameters are kept constants, and the bounds of the rest of parameters are augmented to compensate for the removal of degrees of freedom. The dimension of the new decision vectors is $d = 6$.

B. Compared strategies

The proposed R2R control based on Bayesian optimization (R2R-BO) is compared with two alternatives. They use the same input definition, but different optimization methods. The first R2R strategy is based on the pattern search method (R2R-PS), previously proposed for soft landing of actuators [6], and requires evaluating $2\,d + 1$ points in each iteration (one being the previous best point and the other being around it). The shrink and expand coefficients are set to $1/2$ and 2 respectively, i.e. the mesh size is halved or doubled if the new best point is the same or not, respectively, as the previous one. The starting meshes are set equal for both operations,

$$\boldsymbol{X}_{2\,d+1} = \begin{bmatrix}\mathbf{0} & \boldsymbol{I} & -\boldsymbol{I}\end{bmatrix}. \tag{44}$$

The second strategy is based on the Nelder-Mead method (R2R-NM), which is also a direct search method [18]. Compared to the pattern search, it requires in practice fewer evaluations per iteration. It has been previously applied for other types of learning controllers, e.g. a cyclic adaptive feedforward approach controller for solenoid valves in internal combustion engines [19]. For its application in R2R control, several modifications are introduced to the algorithm: the definition of a minimum simplex volume $V_{\min}$, the modification of the shrink step, and the rearrangement of the algorithm to allow only one evaluation per iteration. For more information about the algorithm, see Appendix A. The initial points are set such that they form a regular simplex, centered in $\mathbf{0}$, with every vertex at a unit distance from it, and randomly rotated. The method also depends on three constant coefficients for the reflection, expansion and contraction steps (c_{ref}, c_{exp} and c_{exp} respectively). The constants of the method are set as follows:

$$V_{\min} = 0.005^d, \quad c_{\text{ref}} = 1, \quad c_{\text{exp}} = 2, \quad c_{\text{con}} = 0.5. \tag{45}$$

With respect to our main proposal R2R-BO, a set of $2\,d + 1$ fixed initial points ($\boldsymbol{X}_{2\,d+1} = \begin{bmatrix}\mathbf{0} & \boldsymbol{I} & -\boldsymbol{I}\end{bmatrix}$) are evaluated, gaining some knowledge across the search space before starting the optimization. Regarding the Gaussian process regression, the prior mean values and kernel hyperparameters from (6) are specified for each case so the optimization process works efficiently. Then, the following variables are initialized:

$$\boldsymbol{L}^0 = \boldsymbol{J}_{d,1}, \qquad \boldsymbol{D_x}^0 = \mathbf{0}. \tag{46}$$

Several combinations of the constants c_{shrink} and α have been tested in a preliminary simulated analysis. The selected ones for the final comparison are:

$$c_{\text{shrink}} = 0.98, \qquad \alpha = 0.9. \tag{47}$$

The remaining constants are set:

$$j_{\max} = 50, \quad \boldsymbol{L_{\min}} = 0.001\,\boldsymbol{J}_{d,1}, \quad \boldsymbol{L_{\max}} = 2\,\boldsymbol{J}_{d,1}. \tag{48}$$

The number of stored data $j_{\max}$ depends on the implementation, but should be chosen as large as possible. The maximum bound length $\boldsymbol{L_{\max}}$ should be selected conservatively, ensuring that any point is reachable from any other. On the other hand, $\boldsymbol{L_{\min}}$ should be small enough to ensure that the smallest region can be properly approximated with $j_{\max}$ points.

C. Simulations

The strategies are compared through a Monte Carlo method, performing 500 simulations of 200 making and breaking commutations for each case. The simulations depend on the parameter vector $\boldsymbol{p}$, which is defined as

$$\boldsymbol{p} = \begin{bmatrix} z_{\min} & z_{\max} & k_{\text{sp}} & z_{\text{sp}} & c_{\text{f}} & N & k_1 & k_2 & k_{\text{eddy}} & m_{\text{mov}} \end{bmatrix}^{\mathsf{T}}. \tag{49}$$

To emulate variability, each model parameter p_i is perturbed in every commutation. Thus, for each kth commutation of the nth simulation, the parameter $p_i^{n,k}$ is defined as a random deviate. Specifically,

$$p_i^{n,k} \sim \mathcal{N}(\bar{p}_i^n,\, (\Delta z^{\text{nom}}\,\sigma_p)^2) \tag{50}$$

if p_i is $z_{\max}$ or $z_{\min}$. Otherwise,

$$p_i^{n,k} \sim \mathcal{N}(\bar{p}_i^n, (p_i^{\text{nom}}\,\sigma_p)^2), \tag{51}$$

where the p_i^{nom} is the corresponding nominal parameter (Table I). The relative standard deviation σ_p serves to emulate the cycle-to-cycle perturbation. Additionally, to emulate unit-to-unit variability, the mean $\bar{\boldsymbol{p}}^n$ is randomly modified in each simulation n, given a continuous uniform distribution,

$$\bar{p}_i^n \sim \text{unif}(p_i^{\text{nom}} - 0.07\,\Delta z^{\text{nom}},\, p_i^{\text{nom}} + 0.07\,\Delta z^{\text{nom}}), \tag{52}$$

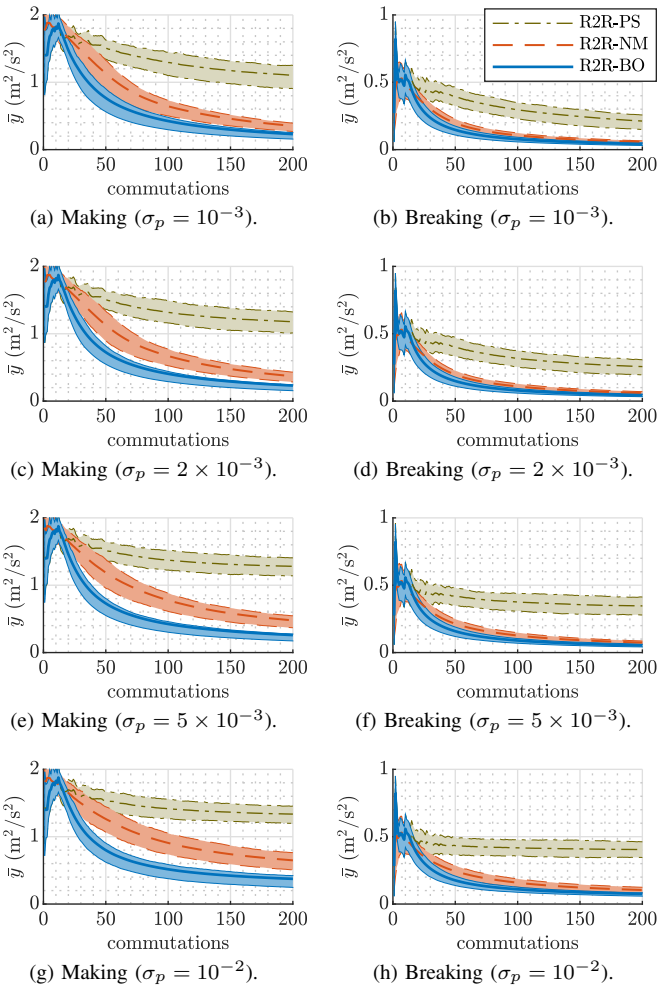
if p_i is $z_{\max}$ or $z_{\min}$. Otherwise,

$$\bar{p}_i^n \sim \text{unif}(0.93\,p_i^{\text{nom}},\, 1.07\,p_i^{\text{nom}}). \tag{53}$$

The objective is to minimize the impacts. Therefore, the output or cost y is defined as the sum of squared velocities during contact. Note that the elastic bouncing phenomenon is not considered in the proposed model, but bouncing will still occur if the impact acceleration has opposite sign to the impact velocity. Then, for every simulation, the average cost $\bar{y}$ is calculated for each number of commutations k, $\bar{y}(k) = \sum_{i=1}^{k} y^i/k$. This way, we determine how good each control is for any number of commutations, and how it improves as the number of commutations increases. In Fig. 6, the mean and 25th-75th percentile intervals of $\bar{y}$ are represented as functions of the number of commutations. Figs. 6a and 6b show the average costs for $\sigma_p = 10^{-3}$. Then, σ_p is increased to 2×10^{-3} (Figs. 6c and 6d), 5×10^{-3} (Figs. 6e and 6f), and 10^{-2} (Figs. 6g and 6h). R2R-BO and R2R-PS has the same 13 starting

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942



(a) Making ($\sigma_p = 10^{-3}$). (b) Breaking ($\sigma_p = 10^{-3}$).

(c) Making ($\sigma_p = 2 \times 10^{-3}$). (d) Breaking ($\sigma_p = 2 \times 10^{-3}$).

(e) Making ($\sigma_p = 5 \times 10^{-3}$). (f) Breaking ($\sigma_p = 5 \times 10^{-3}$).

(g) Making ($\sigma_p = 10^{-2}$). (h) Breaking ($\sigma_p = 10^{-2}$).

Fig. 6. Comparison of results (mean values and 25th-75th percentile intervals) from R2R-PS, R2R-NM and R2R-BO, for different σ_p.

points, so their costs are the same up until that point (the R2R-NM results at the start are also very similar). However, from that point forward the costs from the different strategies start to diverge: the R2R-BO costs are the smallest, following by the costs from R2R-NM, which is still much better than the R2R-PS. As expected, all costs get worse as σ_p increases, and the difference between R2R-BO and R2R-NM augments. In the breaking operations, the costs are generally smaller than in the making operations, and the improvement of R2R-BO over R2R-NM is less significant.

On the one hand, note that R2R-PS and R2R-NM do not require any knowledge of the black-box functions, which makes their implementation for different actuators more straightforward than R2R-BO. They are also more computationally efficient. Between those two, the best strategy is clearly R2R-NM. On the other hand, the best results are obtained with R2R-BO. Moreover, R2R-BO takes into account the stochasticity, so the improvement over R2R-NM is more appreciable for larger cycle-to-cycle variabilities.

D. Experimental results

To validate the R2R control, experimental testings are performed with five solenoid valves. They are supposedly equal to the one used for identification (see Section III-A), but unit-to-unit variability is expected. Each control strategy is tested ten times for each device. The impact velocities are unavailable, due to the lack of a position sensor. Instead, an electret microphone is used to measure the impact sound (Fig. 3). The cost is then calculated from the microphone voltage signals,

$$y = \int_{t_0}^{t_0+\Delta t} {u_{\text{audio}}}^2(t)\,\mathrm{d}t, \tag{54}$$

where t_0 is established as the first instant t where $u_{\text{audio}}(t) > \max(u_{\text{audio}})/5$ and $\Delta t = 0.01$ s.

For reference, a non-controlled case is also evaluated, with squared voltage signals of 0 and 60 V. Then, the controlled costs y are divided by the average non-controlled one, resulting in normalized costs. This analysis is focused on the making operations, which present the most notable impact noises. The normalized costs of the making operations are summarized in the histograms from Fig. 7.

If the control is applied for only 25 commutations, there is a slight improvement over the non-controlled case: in average 13% (R2R-PS), 15% (R2R-NM) and 25% (R2R-BO). As expected, the results improve as the number of commutations increases. For 200 commutations, the average improvements are 35% (R2R-PS), 44% (R2R-NM) and 64% (R2R-BO). Note that the results of R2R-NM are worse than expected from the simulated analysis. In the fist 100 commutations, the difference between R2R-PS and R2R-NM is not very significant. This indicates that the strategy has a slower convergence rate when applied to the real system. Nevertheless, equivalently to the simulated results, the worst strategy is R2R-PS. Note also that R2R-NM is able to obtain normalized costs below 0.10 more often than R2R-BO, but it also obtains normalized costs larger than 0.35 much more frequently. R2R-BO, on the other hand, is more conservative, keeping most normalized costs around 0.25. Thus, its results are the most robust. Moreover, in average, R2R-BO is better than the other two alternatives for any number of commutations.

V. Conclusions

This paper presents a new run-to-run strategy based on Bayesian optimization for soft-landing control of short-stroke reluctance actuators. The control does not use position feedback, which makes it useful for applications in which position sensors or observers are not feasible. The complete algorithm has been separated into different parts, most notably the input definition and the search algorithm.

The current has been selected as the input and an input function has been defined. The input function is model-based, so it requires prior knowledge of the system dynamics. Methods for current parametrization and parameter reduction have been proposed for simple reluctance actuators, and they have been put into effect with the dynamic model of a plunger-type solenoid valve. Using the current as input makes the position dynamics independent on the coil resistance. Thus, the controller is more robust to temperature changes. This is an improvement over previous works [6], [12], which propose to use the voltage as input.

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

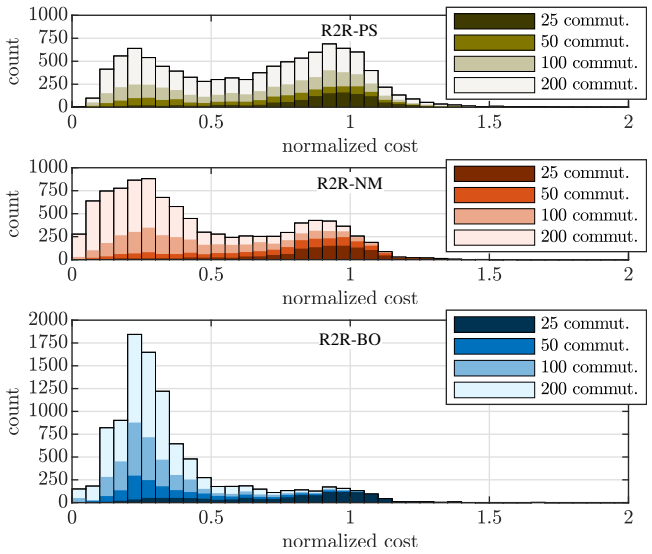


Fig. 7. Histograms of normalized costs for the first 25, 50, 100 and 200 commutations of each control.

Although the input definition is based on a model, the cost function is still a black box. Thus, the cycle-to-cycle search process is a black-box optimization. The proposed algorithm is completely generalized. It is based on Bayesian optimization, which has been adapted in several ways for its application in the run-to-run control. The proposed control (R2R-BO) has been compared with two alternatives, that uses direct search methods (R2R-PS and R2R-NM). One important advantage of R2R-BO is that, as it uses Gaussian process regressors, it directly accounts for uncertainty and hence it is more robust. Besides, it efficiently exploits previous data to select new points and converge rapidly to an optimal solution. The simulated and experimental results show the improvement of R2R-BO with respect to the other alternatives.

APPENDIX A
ALTERNATIVE SEARCH METHOD

For comparison purposes, an alternative search method has been developed. It is based on the Nelder-Mead simplex method, which is one of the most widely used direct search methods. The algorithm keeps a set of $d+1$ evaluated points (where d is the dimension of the search space) forming the vertices of a non-degenerate simplex. In each iteration, one or more tasks are performed, modifying the vertices and reevaluating the function in those new points. The tasks are:

1) The worst vertex (the point which corresponds to the largest function evaluation) is reflected with respect to the centroid of the remaining vertices.
2) If the reflected point is the best so far, the reflected point is expanded farther from the centroid. The best one of these two is kept.
3) If the reflected vertex is the worst point, a contraction is performed toward the centroid.
4) If the contracted point is worse than its original, all vertices of the simplex are modified and evaluated.

In order to implement it in the R2R control, the standard algorithm is modified in several ways. The first modification is related to the last task. The standard method shrinks the simplex, fixed on the best vertex. As stated by various authors [20], [21], the shrinkage is potentially problematic because it may rapidly converge to a non-optimum. It is also inadvisable for optimizing stochastic functions, as the best point is never reevaluated. [20] proposed a translation such that the previous best vertex becomes the center of the new one. In our proposal, the simplex is also centered at the best point. However, instead of maintaining its shape, the vertices are recalculated so they form a randomly rotated regular simplex with the same volume. This prevents both simplex degeneracy and retention of misleadingly good evaluations due to noisy measurements.

Secondly, instead of setting a termination condition, the algorithm is designed to continually operate. Specifically, instead of terminating the optimization when the simplex volume is lesser than a chosen minimum $V_{\min}$, the simplex is prevented to be contracted below that threshold. The simplex volume V is updated in each iteration, augmenting it when expanding and reducing it when contracting, without requiring to compute it from the vertices in every iteration. Also, the actual simplex volume is not important, it is possible to initialize it to 1 and then set $V_{\min}$ in accordance.

The function is described in Algorithm 5. Its inputs are the point $\boldsymbol{x}^k$, which was obtained in the previous iteration, and its evaluation y^k. Its output is the next point $\boldsymbol{x}^{k+1}$. In the process, several variables are updated inside the function: the simplex vertices $\boldsymbol{X} \in \mathbb{R}^{d\times(d+1)}$ and their respective evaluations $\boldsymbol{Y} \in \mathbb{R}^{1\times(d+1)}$, the centroid $\boldsymbol{x}_{\mathbf{c}} \in \mathbb{R}^d$, and the volume V. Normally, the Nelder-Mead method is presented in a sequential way with multiple evaluations per iteration. However, as stated in Algorithm 1, only one evaluation is needed in each function call. Thus, the current step (reflecting, expanding, contracting or rearranging) is stored in the persistent variable $s \in \{1,2,3,4\}$. For the same reason, the index of the vertex to be evaluated is stored in the persistent variable $j \in \{1,2,\ldots,(d+1)\}$. Note that all these variables are different for each operation type but, for the sake of simplicity, that distinction is omitted. The method also depends on three constant coefficients for the reflection, expansion and contraction steps (c_{ref}, c_{exp} and c_{exp} respectively).

REFERENCES

[1] P. Eyabi and G. Washington, "Modeling and sensorless control of an electromagnetic valve actuator," *Mechatronics*, vol. 16, no. 3-4, pp. 159–175, apr 2006.
[2] P. Mercorelli, "An Antisaturating Adaptive Preaction and a Slide Surface to Achieve Soft Landing Control for Electromagnetic Actuators," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 1, pp. 76–85, feb 2012.
[3] M. Rahman, N. Cheung, and Khiang Wee Lim, "Position estimation in solenoid actuators," *IEEE Trans. Ind. Appl.*, vol. 32, no. 3, pp. 552–559, 1996.
[4] E. Ramirez-Laboreo, E. Moya-Lasheras, and C. Sagues, "Real-Time Electromagnetic Estimation for Reluctance Actuators," *IEEE Trans. Ind. Electron.*, vol. 66, no. 3, pp. 1952–1961, mar 2019.
[5] Y. Wang, F. Gao, and F. J. Doyle, "Survey on iterative learning control, repetitive control, and run-to-run control," *J. Process Control*, vol. 19, no. 10, pp. 1589–1600, dec 2009.
[6] E. Ramirez-Laboreo, C. Sagues, and S. Llorente, "A New Run-to-Run Approach for Reducing Contact Bounce in Electromagnetic Switches," *IEEE Trans. Ind. Electron.*, vol. 64, no. 1, pp. 535–543, 2017.
[7] B. Srinivasan, D. Bonvin, E. Visser, and S. Palanki, "Dynamic optimization of batch processes: II. Role of measurements in handling uncertainty," *Comput. Chem. Eng.*, vol. 27, no. 1, pp. 27–44, jan 2003.

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TMECH.2020.2987942

Algorithm 5 Nelder-Mead optimization

function SEARCH($\boldsymbol{x}^k, y^k$)
  Constant: d, c_{ref}, c_{exp}, c_{con}, $V_{\min}$
  Persistent: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{x_c}$, V, s, j
  switch s ▷ Update simplex $\boldsymbol{X}, \boldsymbol{Y}$ and step s
    case 1 ▷ Reflect
      if $y^k \leq Y_{d+1}$ **then**
        $(\boldsymbol{X}_{d+1}, Y_{d+1}) \leftarrow (\boldsymbol{x}^k, y^k)$
      end if
      if $y^k < Y_1$ **then**
        $s \leftarrow 2$
      else if $y^k > Y_d$ **and** $c_{\text{con}} V \geq V_{\min}$ **then**
        $s \leftarrow 3$
      else if $y^k > Y_d$ **then**
        $\boldsymbol{X} \leftarrow$CREATE SIMPLEX($\boldsymbol{X}_1, V$)
        $j \leftarrow 1$ ▷ Initialize index of simplex vertex
        $s \leftarrow 4$
      end if
    case 2 ▷ Expand
      if $y^k < Y_{d+1}$ **then**
        $(\boldsymbol{X}_{d+1}, Y_{d+1}) \leftarrow (\boldsymbol{x}^k, y^k)$
      end if
      $s \leftarrow 1$
    case 3 ▷ Contract
      if $y^k < Y_{d+1}$ **then**
        $(\boldsymbol{X}_{d+1}, Y_{d+1}) \leftarrow (\boldsymbol{x}^k, y^k)$
        $s \leftarrow 1$
      else
        $\boldsymbol{X} \leftarrow$CREATE SIMPLEX($\boldsymbol{X}_1, V$)
        $j \leftarrow 1$ ▷ Initialize index of simplex vertex
        $s \leftarrow 4$
      end if
    case 4 ▷ Simplex
      $Y_j \leftarrow y^k$
      if $j = d + 1$ **then**
        $s \leftarrow 1$
      end if
  switch s ▷ Find next decision vector $\boldsymbol{x}^{k+1}$
    case 1 ▷ Reflect
      $(\boldsymbol{X}, \boldsymbol{Y}) \leftarrow$SORT($\boldsymbol{Y}$) ▷ $Y_1 < Y_2 < \ldots < Y_{d+1}$
      $\boldsymbol{x_c} \leftarrow \sum_{j=1}^{d} \boldsymbol{X}_j / d$ ▷ Centroid of $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_d$
      $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x_c} + c_{\text{ref}} (\boldsymbol{x_c} - \boldsymbol{X}_{d+1})$
    case 2 ▷ Expand
      $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x_c} + c_{\text{exp}} (\boldsymbol{X}_{d+1} - \boldsymbol{x_c})$
      $V \leftarrow c_{\text{exp}} V$ ▷ Update volume
    case 3 ▷ Contract
      $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x_c} + c_{\text{con}} (\boldsymbol{X}_{d+1} - \boldsymbol{x_c})$
      $V \leftarrow c_{\text{con}} V$ ▷ Update volume
    case 4 ▷ Simplex
      $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{X}_j$
      $j \leftarrow j + 1$ ▷ Next simplex vertex
  return $\boldsymbol{x}^{k+1}$
end function

[8] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: a review of algorithms and comparison of software implementations," *J. Glob. Optim.*, vol. 56, no. 3, pp. 1247–1293, jul 2013.
[9] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.
[10] H. Abdelrahman, F. Berkenkamp, J. Poland, and A. Krause, "Bayesian optimization for maximum power point tracking in photovoltaic power plants," in *2016 Eur. Control Conf.*, 2016, pp. 2078–2083.
[11] A. Baheri, S. Bin-Karim, A. Bafandeh, and C. Vermillion, "Real-time control using Bayesian optimization: A case study in airborne wind energy systems," *Control Eng. Pract.*, vol. 69, pp. 131–140, dec 2017.
[12] E. Moya-Lasheras, E. Ramirez-Laboreo, and C. Sagues, "A Novel Algorithm Based on Bayesian Optimization for Run-to-Run Control of Short-Stroke Reluctance Actuators," in *2019 Eur. Control Conf.*, jun 2019, pp. 1103–1109.
[13] E. Ramirez-Laboreo, M. G. L. Roes, and C. Sagues, "Hybrid Dynamical Model for Reluctance Actuators Including Saturation, Hysteresis and Eddy Currents," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 3, pp. 1396–1406, jun 2019.
[14] N. H. Vrijsen, "Magnetic Hysteresis Phenomena in Electromagnetic Actuation Systems," Ph.D. dissertation, Eindhoven University of Technology, 2014.
[15] E. Ramirez-Laboreo, "Modeling and Control of Reluctance Actuators," Ph.D. dissertation, Universidad de Zaragoza, 2019.
[16] E. Ramirez-Laboreo and C. Sagues, "Reluctance actuator characterization via FEM simulations and experimental tests," *Mechatronics*, vol. 56, pp. 58–66, dec 2018.
[17] E. Moya-Lasheras, E. Ramirez-Laboreo, and C. Sagues, "Probability-Based Optimal Control Design for Soft Landing of Short-Stroke Actuators," *IEEE Trans. Control Syst. Technol.*, 2019.
[18] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, jan 1965.
[19] J. Tsai, C. R. Koch, and M. Saif, "Cycle Adaptive Feedforward Approach Controllers for an Electromagnetic Valve Actuator," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 3, pp. 738–746, may 2012.
[20] R. R. Ernst, "Measurement and Control of Magnetic Field Homogeneity," *Rev. Sci. Instrum.*, vol. 39, no. 7, pp. 998–1012, jul 1968.
[21] B. L. Nelson, W. D. Kelton, G. M. Clark, R. R. Barton, and S. John, "Modifications of the Nelder-Mead Simplex Method for Stochastic Simulation Response Optimization," in *1991 Winter Simul. Conf.*, 1991, pp. 945–953.

Eduardo Moya-Lasheras (S'18) received the B.S. degree in industrial technologies engineering in 2014 and the M.S. degree in industrial engineering (mention in industrial automation and robotics) in 2016 from the University of Zaragoza, Zaragoza, Spain, where he is currently working toward the Ph.D. degree in systems engineering and computer science.

Since 2015, he has been with the Departamento de Informatica e Ingenieria de Sistemas (DIIS) and with the Instituto de Investigacion en Ingenieria de Aragon (I3A), University of Zaragoza. His current research interests include modeling, estimation, optimization and control of electromechanical devices.

Carlos Sagues (M'00–SM'11) received the M.S. degree in computer science and systems engineering and the Ph.D. degree in industrial engineering from the University of Zaragoza, Spain, in 1989 and 1992, respectively.

In 1994 he joined, as an Associate Professor, the Departamento de Informatica e Ingenieria de Sistemas, University of Zaragoza, where he became a Full Professor in 2009, and was also the Head Teacher. He was engaged in research on force and infrared sensors for robots. His current research interests include control systems and industry applications, computer vision, visual control, and multivehicle cooperative control.

https://webdiis.unizar.es/~csagues/

Copyright (c) 2020 IEEE. Personal use is permitted. For any other purposes, permission must be obtained from the IEEE by emailing pubs-permissions@ieee.org.