A Novel Algorithm Based on Bayesian Optimization for Run-To-Run Control of Short-Stroke Reluctance Actuators

Eduardo Moya-Lasheras, Edgar Ramirez-Laboreo and Carlos Sagues

***Abstract*— There is great interest in minimizing the impact forces and bounces of reluctance actuators during commutations, in order to reduce acoustic noise and mechanical wear. In that regard, a model-free run-to-run control algorithm is presented to decrease the contact velocity, by exploiting the repetitive operations of these devices. The problem is mathematically formulated and the algorithm is expressed in pseudocode. As the main contribution of this paper, a search method is proposed for the run-to-run strategy based on Bayesian optimization. Adjustments are carried out for its application in run-to-run control, e.g. the removal of stored points and the definition of a new acquisition function. For validation, simulations are performed using a dynamic model of a commercial solenoid valve, and defining the input parametrization. The results show the improvement of the proposed method with respect to a state-of-the-art search.**

I. Introduction

Reluctance-based electromagnetic actuators are electromechanical devices that rely on reluctance forces to change the position of their movables parts. In particular, simple single-coil short-stroke reluctance actuators, e.g. electromagnetic relays or solenoid valves, are extensively used in on-off switching operations of electrical 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 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]. However, in most low-cost short-stroke reluctance actuators, there is no affordable and feasible method to measure the position in real time. For getting around this, estimators can be designed for the armature position [2], or other position-dependent variables [3]. 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.

Cycle-to-cycle learning-type strategies aim at improving the robustness of the soft-landing control by taking advantage of the repetitive functioning of these switching devices. Run-to-run control, in contrast with other learning-type strategies, only requires one evaluation value for each cycle [4]. It is ideal to control low-cost reluctance actuators because, although their position cannot be accurately obtained during operations, other variables can be derived to evaluate each cycle, e.g. the duration of the bouncing [5] 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 [6]. Designing a model-based run-to-run strategy is very challenging, as it needs a robust and accurate model, and may not even be possible for certain low-cost actuators that are not designed to be controlled. Therefore, the proposed optimization is model-free. As such, the cost function to be optimized, which maps the decision (input) with the evaluation (output) variables, is a black box. Without an explicit definition of the cost function, each evaluation requires querying at a certain decision point and actually commuting the device. As stated by [6], 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 [5], is based in a derivative-free pattern search method [7]. However, the optimization still requires several 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—which is typically Gaussian [8]—depending on data from previous iterations and, through the maximization of an utility function, selects the following points to be evaluated. It has proven to be effective in real-time control applications, e.g. maximum power point tracking [9], or altitude optimization of airborne wind energy systems [10].

As the main contribution of this paper, a new approach is proposed for the run-to-run control of reluctance-based electromagnetic actuators, based on Bayesian optimization. Several adjustments are introduced to the algorithm: the limitation of the number of stored previous data by means of the combination or removal of observations, and the definition of a new acquisition function. To demonstrate

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 Ministerio de Educación, Cultura y Deporte, Gobierno de España under grant FPU14/04171, in part by DGA Scholarship Orden IIU/1/2017 confined within Programa Operativo FSE Aragón 2014-2020, and in part by project DGA-T45_17R/FSE.

E. Moya-Lasheras, E. Ramirez-Laboreo 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, ramirlab@unizar.es, csagues@unizar.es).

the effectiveness of the proposal, multiple simulations are performed, with a dynamic model that fits a commercial solenoid valve. Results obtained from using the Bayesian optimization and the state-of-the-art pattern search method are analyzed and compared.

II. Problem statement

Firstly, the dynamics of the system are described. Most commonly in run-to-run (R2R) optimization problems, the system dynamics is formulated with a conventional state-space representation [11], but it is not accurate enough in this scenario. As mentioned in previous works [12], [3], reluctance actuators have three different dynamic modes, corresponding to the upper and lower position boundaries and the motion between those limits. Furthermore, other dynamical or output variables may also have discrete behaviors. Therefore, the dynamical system cannot be considered a continuous dynamical system, but a hybrid automaton. Similarly to the generalized form presented in [13], it can be formulated as

$$\dot{\boldsymbol{\chi}}^k(t) = \boldsymbol{F}_q\left(\boldsymbol{\chi}^k(t),\, \boldsymbol{u}^k(t),\, \boldsymbol{d}^k(t)\right), \quad \boldsymbol{\chi}^k(t) \in C_q, \tag{1}$$

$$\left(\boldsymbol{\chi}^k(t^+),\, q^k(t^+)\right) = \boldsymbol{G}_q\left(\boldsymbol{\chi}^k(t)\right), \quad \boldsymbol{\chi}^k(t) \in D_q, \tag{2}$$

$$\left(\boldsymbol{\chi}^k(0),\, q^k(0)\right) = \left(\boldsymbol{\chi}_0,\, q_0\right), \tag{3}$$

where k is the iteration number, $\boldsymbol{\chi}^k(t)$ the continuous state vector, $q^k(t) \in \{1,\, 2,\, \ldots,\, Q\}$ the discrete state, $\boldsymbol{u}^k(t)$ the continuous input vector, $\boldsymbol{d}^k(t)$ the disturbance vector, $\boldsymbol{F}_q$ the flow map, $\boldsymbol{G}_q$ the jump map, C_q the flow set, D_q the jump set, and $(\boldsymbol{\chi}_0,\, q_0)$ the initial states. To simplify, the usual continuous output vector is not directly defined, but it would be included in—or derived from—$\boldsymbol{\chi}^k(t)$.

Secondly, the input signal is parameterized into a discrete set of decision variables, as is required for R2R control. For any instant t, the input is obtained as

$$\boldsymbol{u}^k(t) = \boldsymbol{\mathcal{U}}\left(\boldsymbol{x}^k,\, t\right), \tag{4}$$

where $\boldsymbol{x}^k$ is the decision vector and $\boldsymbol{\mathcal{U}}$ is the function that generates the input signal.

Thirdly, the variable to be optimized in each repeated cycle is defined. It is obtained directly or indirectly from measurements, which means the optimization problem is measurement-based. This variable may be related to the impact noise level, the bouncing duration, the transient time, or a combination of them. It can be defined as a terminal cost y_r^k to be optimized,

$$y_\mathrm{r}^k = \Psi\left(\boldsymbol{\chi}^k(t_\mathrm{f})\right), \tag{5}$$

being Ψ the cost function and t_f the operation final time.

The optimization process exploits the stored data from previous iterations, which are the decision vectors $\boldsymbol{x}^i$ and their corresponding observed costs y^i, for all $i = 1,\, 2,\, \ldots,\, k-1$. Note that, due to possible measurement errors, the *observed* costs y^i are different from the *real* ones y_r^i.

Finally, the general measurement-based terminal-cost optimization under uncertainty can be formulated as follows,

$$\min_{\boldsymbol{x}^k} \quad y_\mathrm{r}^k = \Psi\left(\boldsymbol{\chi}^k(t_\mathrm{f})\right), \tag{6}$$

$$\text{s.t.} \quad (1) - (4),$$

$$\boldsymbol{S}\left(\boldsymbol{\chi}^k(t),\, \boldsymbol{u}^k(t),\, q^k(t)\right) \leq \boldsymbol{0}, \tag{7}$$

$$\boldsymbol{T}\left(\boldsymbol{\chi}^k(t_\mathrm{f}),\, q^k(t_\mathrm{f})\right) \leq \boldsymbol{0}, \tag{8}$$

$$\text{given} \quad \boldsymbol{\mathcal{X}}^{k-1},\, \mathcal{Y}^{k-1}, \tag{9}$$

$$\text{where} \quad \boldsymbol{\mathcal{X}}^{k-1} = \left\{\boldsymbol{x}^i \,|\, i = 1,\, 2,\, \ldots,\, k-1\right\}, \tag{10}$$

$$\mathcal{Y}^{k-1} = \left\{y^i \,|\, i = 1,\, 2,\, \ldots,\, k-1\right\}, \tag{11}$$

$$y^i = y_\mathrm{r}^i + v^i, \tag{12}$$

being $\boldsymbol{S}$ and $\boldsymbol{T}$ the path and final state constraint functions, $\boldsymbol{\mathcal{X}}^{k-1}$ and $\mathcal{Y}^{k-1}$ the sets of previous decision vectors and observed costs, and v^i the ith observation noise.

Being an implicit R2R optimization, the system dynamics is treated as a black box. To simplify the optimization formulation, the input generation may be included in the black-box function, resulting in the following reformulation,

$$\min_{\boldsymbol{x}^k} \quad y_\mathrm{r}^k = \psi\left(\boldsymbol{x}^k\right) + \delta^k, \tag{13}$$

$$\text{s.t.} \quad \boldsymbol{\mathcal{S}}\left(\boldsymbol{x}^k\right) \leq \boldsymbol{0}, \tag{14}$$

$$\text{given} \quad \boldsymbol{\mathcal{X}}^{k-1},\, \mathcal{Y}^{k-1}, \tag{15}$$

$$\text{where} \quad (10) - (12),$$

where the new black-box cost function ψ maps the decision vector to the cost, δ^k is the additive effect of the disturbance $\boldsymbol{d}^k(t)$ from (1), and the new constraint function $\boldsymbol{\mathcal{S}}$ acts as a replacement of $\boldsymbol{S}$ and $\boldsymbol{T}$.

III. Run-to-run control

This section presents the R2R control strategy with the proposed search method, which is based on Bayesian optimization (R2R-BO).

A. Main algorithm

Firstly, the generalized run-to-run control algorithm is presented. It must be iterative to account for and exploit the cyclic operations of reluctance-based electromagnetic 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, which means that a complete commutation cycle consists of one operation of each.

The R2R solution (see Algorithm 1) consists in a loop in which every iteration k comprises the generation of the input signals for the making and breaking operations ($\boldsymbol{u}_\mathbf{m}^k$ and $\boldsymbol{u}_\mathbf{b}^k$ respectively) from their corresponding decision vectors ($\boldsymbol{x}_\mathbf{m}^k$ and $\boldsymbol{x}_\mathbf{b}^k$), the application of these signals and the observation of the costs (y_m^k and y_b^k), and lastly, the optimization process in which the next decision vectors ($\boldsymbol{x}_\mathbf{m}^{k+1}$ and $\boldsymbol{x}_\mathbf{b}^{k+1}$) are obtained from previous data ($\boldsymbol{\mathcal{X}}_\mathbf{m}^k$, $\mathcal{Y}_\mathrm{m}^k$, $\boldsymbol{\mathcal{X}}_\mathbf{b}^k$ and $\mathcal{Y}_\mathrm{b}^k$).

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

Algorithm 1 Run-to-run

Initialize: $\boldsymbol{x}_{\mathbf{m}}^1$, $\boldsymbol{x}_{\mathbf{b}}^1$
for $k \leftarrow 1$ **to** num. commutations **do**
  $\boldsymbol{u}_{\mathbf{m}}^k(t) \leftarrow$ GENERATE INPUT($\boldsymbol{x}_{\mathbf{m}}^k$)
  $\boldsymbol{u}_{\mathbf{b}}^k(t) \leftarrow$ GENERATE INPUT($\boldsymbol{x}_{\mathbf{b}}^k$)
  Apply $\boldsymbol{u}_{\mathbf{m}}^k(t)$ and measure y_{m}^k
  Apply $\boldsymbol{u}_{\mathbf{b}}^k(t)$ and measure y_{b}^k
  $\boldsymbol{x}_{\mathbf{m}}^{k+1} \leftarrow$SEARCH($\boldsymbol{\mathcal{X}}_{\mathbf{m}}^k$, $\mathcal{Y}_{\mathrm{m}}^k$)
  $\boldsymbol{x}_{\mathbf{b}}^{k+1} \leftarrow$SEARCH($\boldsymbol{\mathcal{X}}_{\mathbf{b}}^k$, $\mathcal{Y}_{\mathrm{b}}^k$)
end for

to adapt the algorithm to work around this issue, e.g. by commuting 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.

While the function GENERATE INPUT must be specifically defined for each situation, the following description of optimization function SEARCH is generalized for any actuator.

B. Bayesian optimization

Bayesian optimization is an iterative method for finding the global optimum of an unknown function. It relies on previous data to build a stochastic model of the black-box function. The selection of the next point $\boldsymbol{x}^{k+1}$ is carried out by maximizing an utility function (acquisition function f_{acqn}). The evaluation y^{k+1} is obtained in the next iteration of the R2R Algorithm 1.

The function is described in Algorithm 2. Its inputs are the current point (decision vector $\boldsymbol{x}^k$), which was obtained in the previous iteration, and its evaluation y^k. Its output is the next point $\boldsymbol{x}^{k+1}$. Some parameters, e.g. the observation noise variance ${\sigma_{\mathrm{n}}}^2$, 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 ${\sigma_{\mathrm{n}}}^2$ is added to the covariance $\boldsymbol{\Sigma}$, which is a diagonal matrix (i.e. noises are uncorrelated).
2) **Data size constraining** ($j \leq j_{\max}$). Observations are merged or removed if necessary. These processes are further discussed in Subsection III-D.
3) **Acquisition**. Selection of next decision vector $\boldsymbol{x}^{k+1}$ by maximizing an acquisition function, given the previous decision vectors $\boldsymbol{X}$, costs $\boldsymbol{Y}$ and the observation covariance $\boldsymbol{\Sigma}$. 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 III-E.

C. Prior and posterior distributions

The selected model for regression is the Gaussian process, which is the most popular one in the context of Bayesian optimization because it only requires simple algebraic operations to determine the corresponding posterior distribution. In general, it is completely specified by a mean $m(\boldsymbol{x})$ and covariance function or kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ [8],

$$f(\boldsymbol{x}) \sim \mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right). \tag{16}$$

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

$$k(\boldsymbol{x}, \boldsymbol{x}') = {\sigma_f}^2 \exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{x}'\right)^{\mathsf{T}} \boldsymbol{M}^{-2}\left(\boldsymbol{x}-\boldsymbol{x}'\right)\right), \tag{17}$$

where ${\sigma_f}^2$ is the characteristic variance and $\boldsymbol{M} \in \mathbb{R}^{d\times d}$ 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 $\boldsymbol{\Sigma}$. By the properties of Gaussian processes, the joint distribution of $\boldsymbol{Y}$ and an output f for an arbitrary $\boldsymbol{x}$ is multivariate normal,

$$\begin{bmatrix}\boldsymbol{Y}^{\mathsf{T}} \\ f\end{bmatrix} \sim \mathcal{N}\left(m\,\boldsymbol{J}_{j+1,1},\, \begin{bmatrix}\boldsymbol{K}+\boldsymbol{\Sigma} & \boldsymbol{k} \\ \boldsymbol{k}^{\mathsf{T}} & k(\boldsymbol{x},\boldsymbol{x})\end{bmatrix}\right), \tag{18}$$

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

$$\boldsymbol{K}_{i,i'} = k(\boldsymbol{X}_i, \boldsymbol{X}_{i'}), \quad \boldsymbol{k}_i = k(\boldsymbol{X}_i, \boldsymbol{x}), \quad \forall i, i' \leq j. \tag{19}$$

The posterior predictive distribution for f is also Gaussian,

$$f \,|\, \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{x} \sim \mathcal{N}(\mu,\, \sigma^2), \tag{20}$$

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

$$\mu = \left(\boldsymbol{Y} - m\,\boldsymbol{J}_{1,j}\right)\left(\boldsymbol{K}+\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{k} + m, \tag{21}$$

$$\sigma^2 = k(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}^{\mathsf{T}}\left(\boldsymbol{K}+\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{k}. \tag{22}$$

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, two adjustments are introduced.

Algorithm 2 Bayesian optimization

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

The first measure considers that, if two or more observations are performed for the same input, there is no need to store them separately. By using Bayesian inference, those cost evaluations can be merged and the equivalent cost mean and variance are obtained.

Suppose that the ith and jth columns of $\boldsymbol{X}$ are equal ($\boldsymbol{X}_i = \boldsymbol{X}_j$). Their corresponding costs are Y_i and Y_j, with variances $\Sigma_{i,i}$ and $\Sigma_{j,j}$. From the first observation, the prior probability density for an arbitrary y is proportional to

$$p(y) \propto \exp\left(-\frac{(y-Y_i)^2}{2\,\Sigma_{i,i}}\right), \tag{23}$$

and the likelihood of the second observation given y is proportional to

$$p(Y_j|y) \propto \exp\left(-\frac{(Y_j-y)^2}{2\,\Sigma_{j,j}}\right). \tag{24}$$

Therefore, the posterior probability density is proportional to

$$\begin{aligned} p(y|Y_j) &\propto p(y)\,p(Y_j|y) \\ &\propto \exp\left(-\frac{(y-Y_i)^2}{2\,\Sigma_{i,i}} - \frac{(Y_j-y)^2}{2\,\Sigma_{j,j}}\right) \\ &\propto \exp\left(-\frac{(y-Y_{\mathrm{eq}})^2}{2\,{\sigma_{\mathrm{eq}}}^2}\right), \end{aligned} \tag{25}$$

where

$$y_{\mathrm{eq}} = \frac{\Sigma_{j,j}\,Y_i + \Sigma_{i,i}\,Y_j}{\Sigma_{i,i}+\Sigma_{j,j}}, \qquad {\sigma_{\mathrm{eq}}}^2 = \frac{\Sigma_{i,i}\,\Sigma_{j,j}}{\Sigma_{i,i}+\Sigma_{j,j}}, \tag{26}$$

which are the resulting cost mean and variance from the merging. As Bayesian inference was used, the substitution of $(Y_i,\, \Sigma_{i,i})$ with $(y_{\mathrm{eq}},\, {\sigma_{\mathrm{eq}}}^2)$ and the removal of $(Y_j,\, \Sigma_{j,j})$ will result in the same posterior Gaussian probability distribution (see equation (20)) for the optimization phase. Note that the condition $\boldsymbol{X}_j = \boldsymbol{X}_i$ can be relaxed to allow some tolerance. A straightforward and effective way is to round all decision vectors, so two points that are very close together are treated as equal.

Merging observations contributes to reducing the size of the data history, but does not guarantee that it is bounded. Therefore, as a second measure, points are removed if the number surpasses the chosen limit $j_{\max}$ (function REMOVE from Algorithm 2). One way of approximating the Gaussian process for large data sets is by selecting a subset. The selection criterion introduced by [14] aims at keeping the most information of the function by maximizing the differential entropy. However, in the presented problem, only one point at most is needed to be removed from the set in each iteration. Consequently, instead of a selection criterion, it is more straightforward and computationally efficient to define a removal criterion. Considering that, the objective is to find the index i which minimizes the increment of entropy due to the removal of $\boldsymbol{X}_i$ from the set $\boldsymbol{X}$,

$$\arg\min_i \left(H(\sigma^2_{i-}) - H(\sigma^2_i)\right), \tag{27}$$

being $H(\sigma^2_i)$ and $H(\sigma^2_{i-})$ the entropy values before and after the removal of $\boldsymbol{X}_i$ respectively. Note that the differential entropy H of a normal distribution depends solely on the variance,

$$H(\sigma^2) = \frac{1}{2}\,\left(1 + \log\left(2\,\pi\,\sigma^2\right)\right). \tag{28}$$

For a point $\boldsymbol{X}_i$, the posterior variance σ^2_i is calculated from (22). If $\boldsymbol{X}_i$ were removed, the resulting posterior variance σ^2_{i-} would increase, depending on the observation noise $\Sigma_{i,i}$,

$$\sigma^2_{i-} = \frac{\Sigma_{i,i}\,\sigma^2_i}{\Sigma_{i,i} - \sigma^2_i}. \tag{29}$$

By disregarding the constants, the derived entropy increment is proportional to

$$H(\sigma^2_{i-}) - H(\sigma^2_i) \propto \log\left(\frac{\sigma^2_{i-}}{\sigma^2_i}\right) \propto -\log\left(1 - \frac{\sigma^2_i}{\Sigma_{i,i}}\right), \tag{30}$$

which monotonically increases with respect to $\sigma^2_i/\Sigma_{i,i}$. Therefore, the index to be removed is simply obtained as

$$\arg\min_i \left(\frac{\sigma^2_i}{\Sigma_{i,i}}\right). \tag{31}$$

E. Acquisition

The last step is the selection of the next point $\boldsymbol{x}^{k+1}$ to evaluate, which must trade off between obtaining the most information of the function (exploration) and attempting to minimize it (exploitation). As 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 (21) and (22). One of the most common acquisition functions is the expected improvement, which is appropriate in regular optimization problems, as it manages to balance exploration and exploitation

Firstly, the improvement of the Gaussian random variable f can be expressed as a function,

$$I(f) = \max(\mu_{\min} - f,\, 0), \tag{32}$$

where $\mu_{\min}$ is the best observation so far, according to their predictive posterior mean values,

$$\mu_{\min} = \min_i \boldsymbol{\mu}(\boldsymbol{X}_i), \tag{33}$$

$$\boldsymbol{\mu}(\boldsymbol{X}) = (\boldsymbol{Y} - m\,\boldsymbol{J}_{1,j})\,(\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}\,\boldsymbol{K} + m\,\boldsymbol{J}_{1,j}. \tag{34}$$

The expected improvement is the expectation of $I(f)$, which can be expressed as

$$\mathrm{E}[I(f)] = \sigma\left(\varphi(f_{\mathrm{n}}) + f_{\mathrm{n}}\,\Phi(f_{\mathrm{n}})\right), \quad f_{\mathrm{n}} = \frac{\mu_{\min} - \mu}{\sigma}, \tag{35}$$

where f_{n} is the normalization of f, while φ and Φ are the standard normal probability and cumulative distribution functions, respectively.

The reasoning behind (32) is that, if the obtained cost f is worse than the best $\mu_{\min}$, there is no improvement, but also no loss, so $I(f)$ is zero. This is suitable for optimization problems in which there is no regret. However, for R2R control, there must be a penalty for obtaining worse outputs than $\mu_{\min}$, as this is conducted in real time.

Assuming that there are n remaining commutations, an improvement over $\mu_{\min}$ would mean a potential improvement

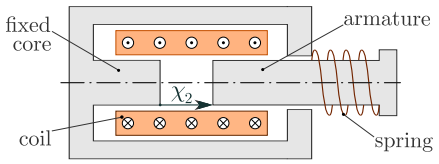


Fig. 1. Simplified diagram of linear-travel solenoid actuator.

for the remaining n commutations. On the other hand, a worsening over $\mu_{\min}$ would only mean a worsening for the next commutation, because in the following one it would be possible to commute with an expected cost of $\mu_{\min}$. Taking that into consideration, the *net* improvement is defined,

$$I_{\text{net}}(f) = \begin{cases} n\,(\mu_{\min} - f), & f \le \mu_{\min}, \\ \mu_{\min} - f, & f > \mu_{\min}, \end{cases} \tag{36}$$

and the proposed acquisition function f_{acqn} is its expectation, which can be expressed in relation to $\mathrm{E}[I(f)]$,

$$f_{\text{acqn}} = \mathrm{E}[I_{\text{net}}(f)] = \sigma\, f_{\text{n}} + (n-1)\,\mathrm{E}[I(f)]. \tag{37}$$

Note that, if $n = 1$, the expected I_{net} is simply $\mu_{\min} - \mu$, and its maximization is equivalent to the minimization of μ. As n decreases, the acquisition favors exploitation over exploration, which is the intended behavior. Also, notice that as n increases, $\mathrm{E}[I_{\text{net}}(f)]/n$ tends to $\mathrm{E}[I(f)]$, which would correspond to a regret-free optimization.

IV. Analysis

In this section, the proposed run-to-run control based on Bayesian optimization (R2R-BO) is compared with the state-of-the-art run-to-run strategy. For that purpose, simulations are performed with both alternatives.

A. System model, input and output

Firstly, a dynamic model for a reluctance actuator is defined. The chosen one is based on a commercial low-cost linear travel solenoid valve, which is schematically depicted in Fig. 1. The core is cylindrically asymmetrical, 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 at the opposite direction. The air gap length—and the armature position—is restricted to $[z_{\min},\, z_{\max}]$.

The system is modeled with a hybrid automaton. There are three state variables: the magnetic flux χ_1, the position χ_2 and the velocity χ_3. As the first operation in each cycle is the making, or closing of the gap, the initial state $\boldsymbol{\chi}_0$ is set to $[0 \quad z_{\max} \quad 0]^{\mathsf{T}}$. The input u is the voltage between the coil terminals. As the motion is constrained, χ_2 and χ_3 are static if the armature reaches one of the two limits. Additionally, as the generated magnetic force cannot change direction, it is convenient to restrict also the current and magnetic flux to non-negative values. In practice, this would be implemented with diodes, which only allow one current flow direction. In the model, this is achieved by the addition of dynamic modes in which the magnetic flux is static. Ultimately, the automaton presents six discrete states,

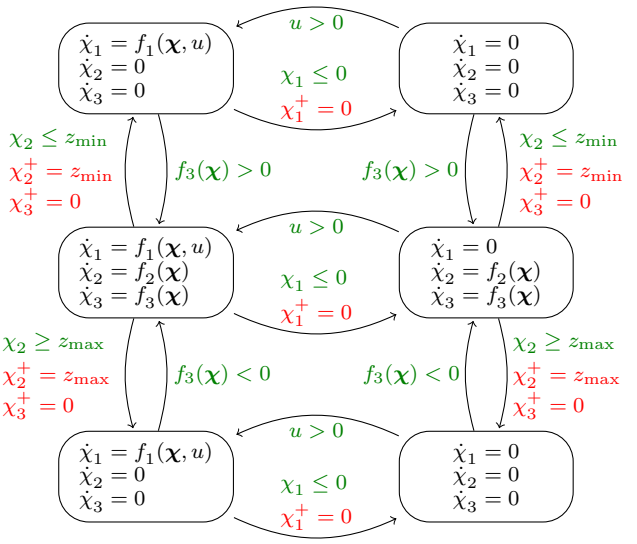


Fig. 2. Diagram of the hybrid automaton. Guard conditions in green and reset states in red.

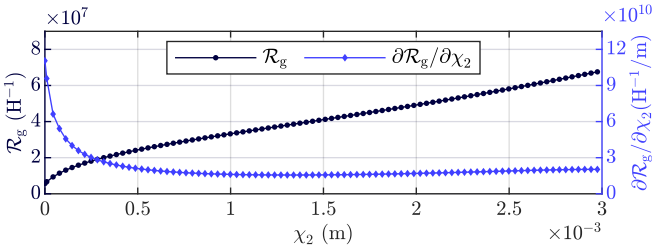


Fig. 3. Gap reluctance and its derivative with respect to the gap length.

with the corresponding guard conditions and reset states, as illustrated in Fig. 2. To shorten the expressions, the time t dependency and the iteration k distinction are omitted.

The dynamic functions $f_1(\boldsymbol{\chi}, u)$, $f_2(\boldsymbol{\chi})$ and $f_3(\boldsymbol{\chi})$, which were derived in a previous work [12], are defined as

$$f_1 = \frac{u}{N} - \frac{R\,(\mathcal{R}_{\text{c}}(\chi_1) + \mathcal{R}_{\text{g}}(\chi_2))\,\chi_1}{N^2}, \tag{38}$$

$$f_2 = \chi_3, \tag{39}$$

$$f_3 = \frac{1}{m_{\text{mov}}}\left(k_{\text{s}}\,(z_{\text{s}} - \chi_2) - c_{\text{f}}\,\chi_3 - \frac{\partial\mathcal{R}_{\text{g}}(\chi_2)}{\partial\chi_2}\frac{{\chi_1}^2}{2}\right), \tag{40}$$

where m_{mov} is the moving mass, k_{s} is the spring stiffness coefficient, z_{s} is the spring equilibrium position, c_{f} is the damping friction coefficient, R is the internal resistance, N is the number of coil turns, $\mathcal{R}_{\text{c}}$ is the core reluctance and $\mathcal{R}_{\text{g}}$ the gap reluctance. In order to consider the rarely negligible magnetic saturation phenomenon, $\mathcal{R}_{\text{c}}$ is modeled as a parametric expression derived from the Fröhlich-Kenelly equation,

$$\mathcal{R}_{\text{c}}(\chi_1) = \frac{k_1}{1 - k_2\,|\chi_1|}. \tag{41}$$

In contrast, $\mathcal{R}_{\text{g}}$ and its derivative are obtained from lookup tables, whose data were calculated from a finite element model, and are represented in Fig. 3.

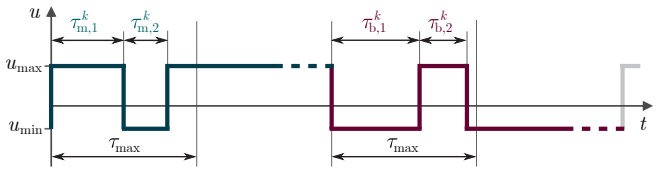


Fig. 4. Profile of the voltage input signals for breaking and making operations of the kth iteration.

TABLE I

SYSTEM DYNAMICS AND INPUT PARAMETERS

Parameter	Value	Parameter	Value
m_{mov}	1.6×10^{-3} kg	R	55 Ω
k_{s}	74.05 N/m	N	1200
z_{s}	1.5×10^{-2} m	k_1	2.32×10^{6} H^{-1}
c_{f}	0 Ns/m	k_2	5×10^{4} Wb^{-1}
$\bar{z}_{\min}$	3×10^{-4} m	$u_{\min/\max}$	∓50 V
$\bar{z}_{\max}$	1.3×10^{-3} m	$\tau_{\max}$	3×10^{-3} s

Secondly, an input parametrization method is defined, for the function GENERATE INPUT from Algorithm 1. The voltage is defined as a square signal, parameterized with two time intervals ($d = 2$) for each operation $\tau^k_{\mathrm{m/b},1}$ and $\tau^k_{\mathrm{m/b},2}$, as shown in Fig. 4. The constant $\tau_{\max}$ is the maximum time interval allowed for the commutation, and must be set accordingly. The two interval parameters must depend on the decision vectors $\boldsymbol{x}^k_{\mathbf{m/b}}$. They are normalized as follows,

$$x^k_{\mathrm{m/b},1} = \frac{\tau^k_{\mathrm{m/b},1}}{\tau_{\max}}, \qquad x^k_{\mathrm{m/b},2} = \frac{\tau^k_{\mathrm{m/b},2}}{\tau_{\max} - \tau^k_{\mathrm{m/b},1}}. \tag{42}$$

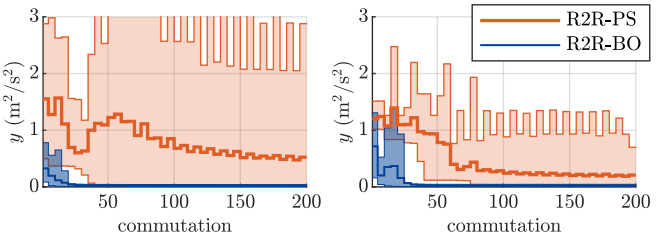
This way, every decision variable can conveniently be bounded to $[0, 1]$, and it is ensured that $\tau^k_{\mathrm{m/b},1} + \tau^k_{\mathrm{m/b},2} \leq \tau_{\max}$. The constant $\tau_{\max}$ must be set accordingly.

Thirdly, as the objective is to minimize the impacts, the output or cost is defined as the sum of squared velocities during contact. Note that the bouncing phenomenon is not considered in the proposed model, but several impacts can still occur in each making or breaking operation. Furthermore, to be consistent with the formulation from Section II, the cumulative squared contact velocities should have been added to the model as an auxiliary variable.

B. Simulations

Finally, simulations are performed to analyze the proposed run-to-run control, and to compare the optimization algorithm with the pattern search method. To simulate the disturbance, white noise is added to the position constraints: $z^k_{\min} = \bar{z}_{\min} + \varepsilon^k_{z\min}$ and $z^k_{\max} = \bar{z}_{\max} + \varepsilon^k_{z\max}$, where $\varepsilon^k_{z\min}$ and $\varepsilon^k_{z\min}$ are independent normal random variables with zero mean and standard deviation σ_z. The parameters for the input generation and the model are specified in Table I.

The run-to-run strategy based on the pattern search method (R2R-PS), as it was previously proposed for soft landing of solenoid actuators, requires evaluating $2\,d + 1$ times in each iteration, one is the reevaluation of the previous best point and the other are evaluations around this point. As $d = 2$, it is necessary to perform 5 making and 5 breaking commutations in each iteration. The starting mesh for both operation types is composed of the points $[0.5 \quad 0.5]^{\mathsf{T}}$, $[0.75 \quad 0.5]^{\mathsf{T}}$, $[0.5 \quad 0.75]^{\mathsf{T}}$, $[0.25 \quad 0.5]^{\mathsf{T}}$ and $[0.5 \quad 0.25]^{\mathsf{T}}$. 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.



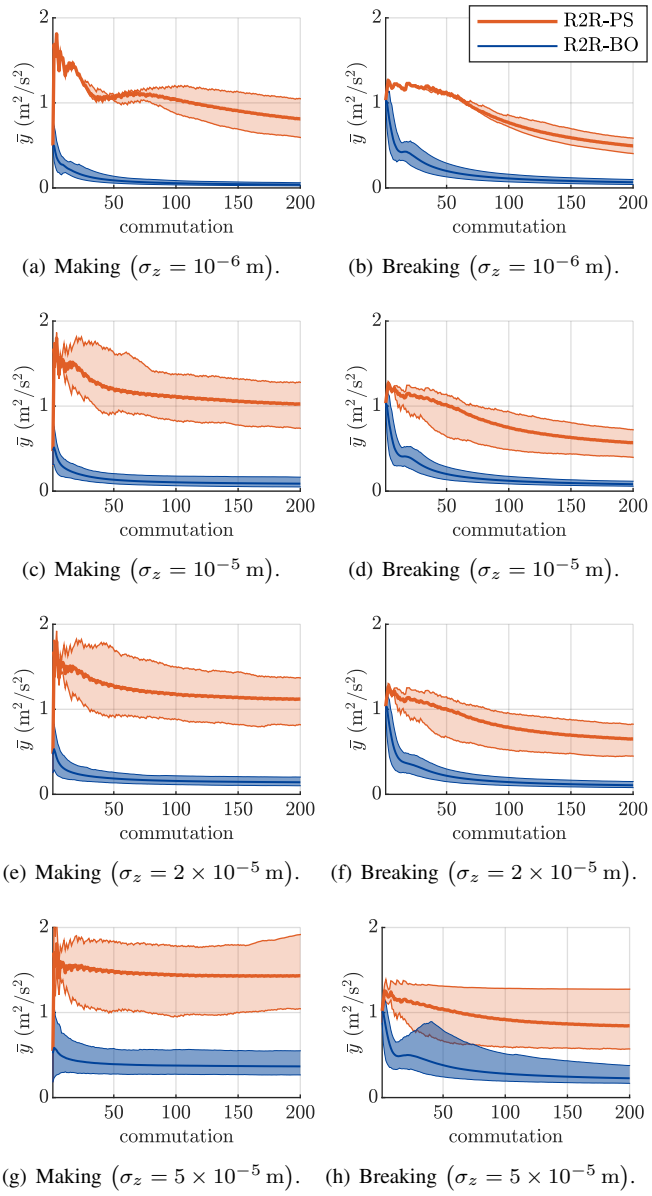
(a) Making operation. (b) Breaking operation.

Fig. 5. Comparison of results (mean values and 5th-95th percentile intervals) from R2R-PS and R2R-BO, for $\sigma_z = 10^{-6}$ m.

The proposed run-to-run algorithm based on Bayesian optimization (R2R-BO), on the other hand, requires one evaluation per iteration, independently of d. The first evaluated decision vector is set to $[0.5 \quad 0.5]^{\mathsf{T}}$ for both making and breaking operations. To limit the stored data, $j_{\max}$ is set to 50. Furthermore, the prior mean values and kernel hyperparameters from (17) are specified for each case so the optimization process works efficiently.

The two optimization methods are compared through a Monte Carlo method: performing 500 simulations of 200 making and breaking commutations for each given σ_z. For the first comparison, σ_z is set to 10^{-6} m and the resulting costs y are obtained for each commutation and each operation type. R2R-PS requires 5 commutations per iteration so, for a better visualization, the results are grouped 5 by 5. In Fig. 5, from each set of 2500 costs, the mean is displayed, as well as the interval between the 5th and 95th percentiles. R2R-PS is able to reach costs close to zero, as the 5th percentile values indicate, but the observation randomness, albeit low, slows down the convergence rate—especially in the making operation—, as the mean and 95th percentile values indicate. In contrast, R2R-BO does not have that problem, as it takes into account the stochasticity.

For the following graphics, we display the average cost $\bar{y}$ for each number of commutations k, $\bar{y}(k) = \sum_{i=1}^{k} y^i/k$. It varies less abruptly, making it more suitable for increasing position deviations and eliminating the need of grouping in sets of 5. In Fig. 6, the mean and percentile intervals of $\bar{y}$ are represented as a function of the number of commutations. Figs. 6(a) and 6(b) show the average costs for $\sigma_z = 10^{-6}$ m, obtained from the previous costs y (Figs. 5(a) and 5(b)). The σ_z is increased to 10^{-5} m (Figs. 6(c) and 6(d)), 2×10^{-5} m (Figs. 6(e) and 6(f)), and 5×10^{-5} m (Figs. 6(g) and 6(h)). As expected R2R-BO is consistently better than R2R-PS. The R2R-BO results are very good for $\sigma_z \leq 2 \times 10^{-5}$ m. In particular, the R2R-BO making costs are slightly better than the breaking ones for $\sigma_z = 10^{-6}$ m because, in this



(a) Making $\left(\sigma_z = 10^{-6}\,\mathrm{m}\right)$. (b) Breaking $\left(\sigma_z = 10^{-6}\,\mathrm{m}\right)$.

(c) Making $\left(\sigma_z = 10^{-5}\,\mathrm{m}\right)$. (d) Breaking $\left(\sigma_z = 10^{-5}\,\mathrm{m}\right)$.

(e) Making $\left(\sigma_z = 2 \times 10^{-5}\,\mathrm{m}\right)$. (f) Breaking $\left(\sigma_z = 2 \times 10^{-5}\,\mathrm{m}\right)$.

(g) Making $\left(\sigma_z = 5 \times 10^{-5}\,\mathrm{m}\right)$. (h) Breaking $\left(\sigma_z = 5 \times 10^{-5}\,\mathrm{m}\right)$.

Fig. 6. Comparison of results (mean values and 5th-95th percentile intervals) from R2R-PS ans R2R-BO, for different position deviations.

specific case, the starting point is significantly better. Despite that, for greater deviations, the breaking average costs reach lower minimums. Notice also that setting $\sigma_z = 5 \times 10^{-5}$ m, although it is merely a 5 % of the nominal valve travel distance, is enough to make the impact velocities vary greatly between commutations, especially in the making operations.

In general, note that R2R-PS does not require any knowledge of the black-box functions, which makes its implementation for different actuators more straightforward, but in turn is less efficient, even with low uncertainties or disturbances. However, it could be improved by adjusting or adapting the shrink and expand coefficients. It could also be improved by storing previous data, like R2R-BO, and merging observations of same points.

V. Conclusions

In this paper, we have presented a new run-to-run strategy with a search algorithm based on Bayesian optimization, for soft-landing control of short-stroke reluctance actuators. Simulation results show the improvement over another state-of-the-art run-to-run strategy [5], which was based on a pattern search method and aimed at reducing contact bounce in relays and contactors.

One important advantage of the proposed strategy (R2R-BO) is that, as it uses Gaussian process regressors, it directly accounts for uncertainty and hence it is more robust. Another advantage is the efficient exploitation of previous data to select following points and converge rapidly to an optimum, by means of the mentioned regression and the definition of an appropriate acquisition function.

The proposed method has still room for improvement. Similarly to pattern search and other methods, like random search, the algorithm could benefit from adaptive search bounds, closing the space around the best point and increasing the convergence rate. This could be specially useful if the dimension of the decision vector increases, and the maximum number of stored points is insufficient for generating a regressor accurate enough for the entire space.

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] 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.

[3] 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, 2019.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[9] H. Abdelrahman, F. Berkenkamp, J. Poland, and A. Krause, "Bayesian optimization for maximum power point tracking in photovoltaic power plants," *2016 Eur. Control Conf.*, pp. 2078–2083, 2016.

[10] 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.

[11] B. Srinivasan, C. Primus, D. Bonvin, and N. Ricker, "Run-to-run optimization via control of generalized constraints," *Control Eng. Pract.*, vol. 9, no. 8, pp. 911–919, aug 2001.

[12] E. Moya-Lasheras, C. Sagues, E. Ramirez-Laboreo, and S. Llorente, "Nonlinear bounded state estimation for sensorless control of an electromagnetic device," in *2017 IEEE 56th Annu. Conf. Decis. Control*. IEEE, dec 2017, pp. 5050–5055.

[13] R. Goebel, R. G. Sanfelice, A. R. Teel, and A. R. Goebel, Rafal and Sanfelice, Ricardo G and Teel, "Hybrid dynamical systems," *IEEE Control Syst. Mag.*, vol. 29, no. 2, pp. 28–93, 2009.

[14] N. D. Lawrence, M. Seeger, and R. Herbrich, "Fast sparse Gaussian process methods: The informative vector machine," *Adv. Neural Inf. Process. Syst.*, vol. 15, pp. 609–616, 2002.