Pairwise Preferences-Based Optimization of a Path-Based Velocity Planner in Robotic Sealing Tasks

Loris Roveda[1], Beatrice Maggioni[2], Elia Marescotti[2], Asad Ali Shahid[1], Andrea Maria Zanchettin[2], Alberto Bemporad[3], Dario Piga[1]

***Abstract*—Production plants are being re-designed to implement human-centered solutions. Especially considering high added-value operations, robots are required to optimize their behavior to achieve a task quality at least comparable to the one obtained by the skilled operators. A manual programming and tuning of the manipulator is not an efficient solution, requiring to adopt towards automated strategies. Adding external sensors (*e.g.*, cameras) increases the robotic cell complexity and it doesn't solve the issue since it is usually difficult to build explicit reward functions measuring the robot performance, while it is easier for the user to define a qualitative comparison between two experiments. According to these needs, in this paper, the recently-developed preferences-based optimization approach GLISp is employed and adapted to tune the novel developed path-based velocity planner. The implemented solution defines an intuitive human-centered procedure, capable of transferring (through pairwise preferences between experiments) the task knowledge from the operator to the manipulator. A Franka EMIKA panda robot has been employed as a test platform to perform a robotic sealing task (*i.e.*, material deposition task), validating the proposed methodology. The proposed approach has been compared with a programming by demonstration approach, and with the manual tuning of the path-based velocity planner. Achieved results demonstrate the improved deposition quality obtained with the proposed optimized path-based velocity planner methodology in a limited number of experimental trials (20).**

I. Introduction

Within Industry 4.0 paradigm, industrial workplaces are being re-designed to enhance the automation of the production lines by exploiting human-centered robotic solutions. Robots are substituting the operators in tedious and difficult applications [1], while assisting them in higher added-value tasks [2]. However, the simple replication of a target task/behavior is not anymore sufficient. In fact, customized production and variability in the operations [3] require the robot to continuously learn new applications, being able to adapt to (partially) new scenarios while optimizing the target task quality [4]. Such a capability is particularly required for high-precision tasks, in which the robot motion affects the final result, as in sealing and welding applications [5]. In these cases, the robot velocity has to be adapted on the basis of the target geometrical path features to achieve the expected quality. A manual setting of the reference velocity might be beneficial to transfer the task knowledge from an expert operator to the manipulator. Nevertheless, such *ad hoc* programming or physical demonstration of each new trajectory is not always feasible, mainly because of the required effort. In addition, programming by demonstration might require external sensors (such as cameras [6]), increasing the complexity of the robotic cell. Such solutions are not commonly applied in a real industrial environment, in which light conditions, dirty workplaces, and dynamic reconfigurability of the cell might prevent the use of external sensors. Indeed, an intuitive control tuning approach, exploiting the expertise of the operator, is expected to have a high impact in the considered context.

In this paper, a novel path-based velocity planner is developed to execute a robotic sealing task, adapting the execution velocity on the basis of the confronted geometrical features. A semi-automated tuning approach is then proposed for its parameters optimization, where the user is only requested to iteratively provide qualitative feedback by expressing pairwise preferences between experiments.

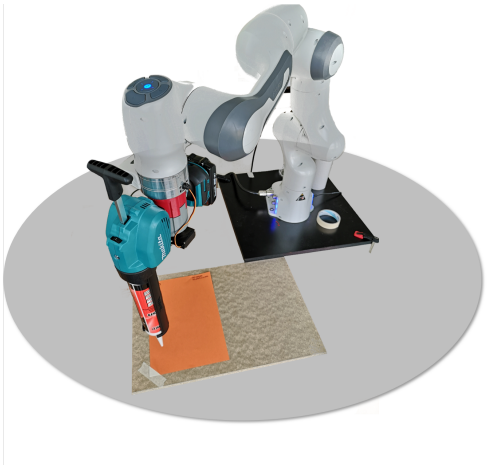


Fig. 1: Experimental setup for the validation of the proposed user's pairwise preferences-based optimization methodology applied to the tuning of a velocity planner for the execution of a sealing task. A Franka EMIKA panda robot is equipped with a MAKITA sealing gun. A servomotor (controlled with Arduino) has been embedded in the *ad hoc* designed flange to activate/deactivate the deposition.

Manuscript received: May 21, 2021; Accepted: June 28, 2021.

This paper was recommended for publication by Editor Hanna Kurniawati upon evaluation of the Associate Editor and Reviewers' comments.

The work has been developed within the project ASSASSINN, funded from H2020 CleanSky 2 under grant agreement n. 886977.

[1] Loris Roveda, Asad Ali Shahid, and Dario Piga are with Istituto Dalle Molle di studi sull'Intelligenza Artificiale (IDSIA), USI-SUPSI, Lugano, Switzerland loris.roveda@idsia.ch

[2] Beatrice Maggioni, Elia Marescotti, and Andrea Maria Zanchettin are with Politecnico di Milano, Department of Mechanical Engineering, Milano, Italy

[3] Alberto Bemporad is with the IMT School for Advanced Studies Lucca, Lucca, Italy

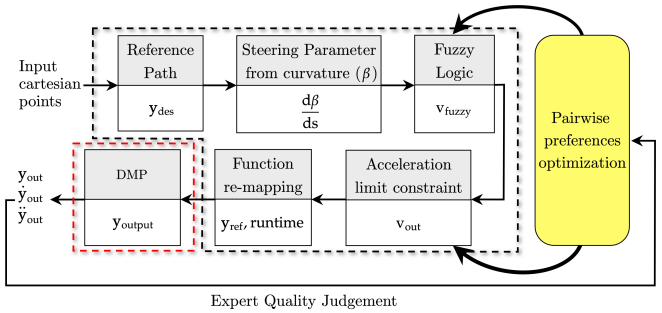


Fig. 2: Velocity planner enhanced by the user's pairwise preferences-based optimization methodology for parameter tuning. The velocity planner is composed by two main components: a *pre-processing* part (black-dashed box) and an *execution* part (red-dashed box). The user's pairwise preference based optimization (yellow block) tunes the *Fuzzy Logic* and the *acceleration limit constraint* blocks parameters.

A. Related work

1) Path-based velocity planner: Trajectory planning for robotic systems is still a hot topic [7]. In fact, the robot motion has to be designed on the basis of the specific operating environment and the task to maximize its performance. Prior works have focused on the optimization of the velocity profile for various applications, such as spray painting applications [8] and underwater welding [9]. In [10], an algorithm that has been developed for welding applications is described. The planner finds the optimized motion for both the robot end-effector and joints, but it doesn't set the velocity along the path. In [11], a sealing task is performed using global planning interpolation and trapezoidal speed profile, although without considering any variable velocity along the path. Some advanced methods have been proposed in order to generate the robot trajectory planning, such as Dynamic Movement Primitives [12]. However, the main issue with the available trajectory planning approaches is that they do not address the problem of the punctual velocity characterization along the path's natural coordinate (critical issue for many applications, such as sealing tasks).

2) Preferences-based optimization algorithms: User's preferences-based learning techniques have been used in many domains. The contribution in [13] exploits active learning by querying the human expert to rank the execution performance of a grasping task. In [14], a preferences-based method has been proposed to learn a reward function in the context of reinforcement learning. A deep neural network is trained that predicts a reward using human preferences over pairs of trajectories for learning Atari games and robotic tasks in simulation. User's preferences have also been combined with demonstrations to reduce the required number of queries [15], where a robotic manipulator is trained to reach a goal configuration while avoiding an obstacle. Calibration of model predictive control parameters based on user's preferences is discussed in [16]. In [17], a Bayesian deep learning method is proposed to optimize the parameters for a navigation task using humans' preference evaluations.

B. Paper contribution

This paper develops a novel path-based velocity planner for robotic tasks. By exploiting a Fuzzy Logic approach, the developed velocity planner allows to classify the encountered geometrical path features (*i.e.*, *straight lines*, *large curves*, *tight curves*, *very tight curves*), accordingly modulating their execution velocity. The Dynamic Movement Primitives framework is then employed for the generation of the robot's reference velocity. The execution velocities of the geometrical features, together with the acceleration limit, must be optimized to maximize the task quality (*i.e.*, homogeneity of the deposition, absence of vibrations and instabilities). For such a purpose, a user's pairwise preferences-based optimization methodology is employed. The pairwise preferences-based optimization algorithm GLISp [18] is used, which iteratively suggests a new set of tuning parameters to the user for testing and comparison against the previous best task execution. In addition, a set of acceptability criteria are also defined: general acceptability, deposition vibrations, straight lines homogeneity, large curves homogeneity, tight curves homogeneity, very tight curves homogeneity. Indeed, it is possible to provide useful information to the optimization, penalizing the configurations that do not satisfy the user.

A Franka EMIKA panda robot, equipped with a MAKITA sealing gun, has been employed as a test platform for a sealing task (Figure 1). The robot has to deposit the sealant material on a complex path. The velocity planner must be tuned to provide the proper reference velocity for each above defined geometrical path feature, optimizing the deposition. Indeed, the robot's execution velocity affects the deposition (*e.g.*, an excessively slow motion results in material accumulation, while a too fast motion results in a lack of material). The pairwise preferences-based optimization approach has been compared with the programming by demonstration approach in [4], and with the manual tuning of the path-based velocity planner parameters. An improved deposition quality is shown in a limited number of experimental trials (20), transferring the expert operator's knowledge to the robot in an intuitive and effective way. Quantitative measurements on the thickness of the deposited material are given to support the achieved results.

The main contributions of the paper are therefore:

- the development of a novel path-based velocity planner, modulating the robot velocity on the basis of the encountered geometrical features;
- the adoption of an advanced preferences-based optimization algorithm for the tuning of the velocity planner parameters in a real industrial challenging robotic task;
- the validation of the proposed control and optimization framework w.r.t. the available state of the art methods (comparing its performance with a programming by demonstration approach).

II. Velocity Planner

In order to execute a velocity-dependent task (such as welding or sealing), an advanced path-based velocity planner has to be designed. In fact, on the basis of the confronted geometrical path features, the robot's velocity has to be

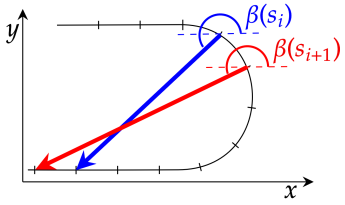


Fig. 3: Steering parameter *steer* computation on the basis of the local path curvature, with $n = 7$.

modulated in order to achieve the target task quality. This paper proposes a novel path-based velocity planner to face the described issue. In addition, due to the fact that the tuning of the robot velocity for each considered geometrical feature is not trivial, an automated optimization strategy is required to solve this issue. This paper adopts an intuitive optimization methodology capable of exploiting the qualitative task evaluations (in terms of pairwise preferences between the quality achieved in different experiments) from an expert operator, avoiding the use of external sensors and the definition of complex reward functions. The developed path-based velocity planner scheme is shown in Figure 2, highlighting the user's pairwise preference-based optimization for parameter tuning. The velocity planner is composed of two main components: a *pre-processing* part (black-dashed box in the figure) and an *execution* part (red-dashed box). The two components are described below.

A. Pre-processing block

1) Reference path pre-processing block: The algorithm considers a generic input matrix defining the robot end effector's reference path in the Cartesian space (*i.e.*, considering the required degrees of freedom for the task execution, possibly involving both Cartesian translations and rotations). The input path is generated by the operator (*e.g.*, based on the CAD of the target part to be processed). The input path can be generated making use of mathematical functions (if possible), or by the definition of via points (properly positioned to describe the path shape) to be interpolated. Then, then input path is re-sampled in order to be equally spaced along the path natural coordinate s. This procedure permits to easily characterize the geometrical path features, homogeneously considering the punctual complexity of the path. To properly approximate the path, the discretization step d_s has to be defined on the basis of the characteristic size of the tool used for the task execution (*e.g.*, the diameter of the nozzle of the tool for the material deposition). In fact, it is possible to better characterize the path geometrical features correlating the characteristic size of the tool with the discretization step d_s. Once the pre-processing is performed, the output is sent to the next block for the characterization of the geometrical path features.

2) Steering parameter computation block: The geometrical path features are evaluated on the basis of the local path curvature. For this purpose, the steering parameter *steer* is introduced (Figure 3), and it is computed as follows:

- vector $v_{i,i+n}$ is defined, connecting the actual path point with the one being n steps forward;
- vector $v_{i+1,i+1+n}$ is defined, connecting the next point with the one being $n+1$ steps forward;
- angles $\beta(s_i)$ and $\beta(s_{i+1})$ are computed (between the horizontal axis and the vector $v_{i,i+n}$/$v_{i+1,i+1+n}$, respectively);
- the steering parameter is defined as:

$$steer = |\beta(s_{i+1}) - \beta(s_i)|. \tag{1}$$

Therefore, for a straight line, the steering parameter *steer* will be null, and it progressively rises as the curve becomes tighter. The absolute value of the difference between $\beta(s_{i+1})$ and $\beta(s_i)$ is considered for the calculation of the steering parameter *steer*. In fact, with the aim to characterize the geometrical path features, left-wise **or** right-wise curves are treated the same.

The steering parameter *steer* is then sent to the *Fuzzy Logic* block in order to perform the classification of the geometrical path features.

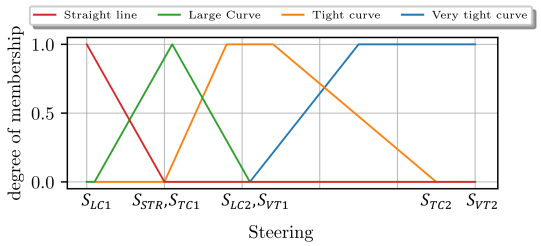
3) Fuzzy Logic block: A mathematical correlation between the geometrical path features and a proper execution velocities is too difficult to be defined in a closed form. Such a correlation, in fact, is a function of the process characteristics, of the robot dynamics, as well as of the final task quality. Therefore, a heuristic methodology to classify the geometrical path features has to be defined. To this aim, a Fuzzy Logic approach [19] is proposed in this paper.

The Fuzzy Logic block is build on the definition of its input and output membership functions. The input membership function is based on the following classification of the geometrical path features: *straight lines*, *large curves*, *tight curves*, *very tight curves*. This classification is performed on the basis of the steering parameter *steer* in (1), which is then used as an input to the Fuzzy Logic block (Figure 4a). The output membership function is based on the definition of following task execution velocities: *very slow*, *slow*, *medium*, *fast*. The Fuzzy Logic block gives the execution velocity v_{fuzzy} for the actual classified geometrical path feature as an output (Figure 4b). The velocity v_{fuzzy} is defined on the basis of the parameters v_{vs}, v_{s1}, v_{s2}, v_{m1}, v_{m2}, v_{f1}, v_{f2} shaping the output membership function. The following rules are applied to correlate the classification of the geometrical path features and the Fuzzy Logic output velocities:

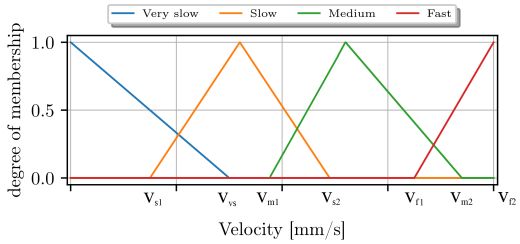
$$\begin{cases} \#1\ If\ steer:\ straigth\ line,\ then\ v_{fuzzy}: fast, \\ \#2\ If\ steer:\ large\ curve,\ then\ v_{fuzzy}: medium, \\ \#3\ If\ steer:\ tight\ curve,\ then\ v_{fuzzy}: slow, \\ \#4\ If\ steer:\ very\ tight\ curve,\ then\ v_{fuzzy}: very\ slow. \end{cases} \tag{2}$$

In order to properly execute the target task, the output membership function in Figure 4b has to be properly defined for the calculation of the velocity v_{fuzzy}. Therefore, the parameters v_{vs}, v_{s1}, v_{s2}, v_{m1}, v_{m2}, v_{f1}, v_{f2} have to be optimized to maximize the task quality.

The Fuzzy Logic output velocity v_{fuzzy} is then sent to the *acceleration limit modulation* block in order to satisfy the acceleration constraints (*e.g.*, related to the used hardware).



(a) Fuzzy Logic input membership function.



(b) Fuzzy Logic output membership function.

Fig. 4: Fuzzy Logic membership functions.

4) Acceleration limit modulation block: The velocity v_{fuzzy}, which is expressed as a function of the path natural coordinate, is modulated to satisfy target acceleration/deceleration limit a_{max}. The following criteria are defined to modulate v_{fuzzy}:

- if $a(s_{i+1}) > a_{max}$, then set $a(s_{i+1}) = a_{max}$, such that $v_{out}(s_{i+1}) < v_{fuzzy}(s_{i+1})$ and the new time instant is set to be longer;
- if $a(s_{i+1}) < -a_{max}$, then fix $v_{out,back}(s_{i+1}) = v_{fuzzy}(s_{i+1})$, and lower the previous computed velocity values, $v_{out,old}(s_i, s_{i-1}, \dots)$, up to the convergence of related deceleration values (backward).

The acceleration/deceleration limit a_{max} can be optimized in order to satisfy the hardware limits, while maximizing the target task quality. In fact, high acceleration/deceleration might result in vibrations, thus compromising the task quality.

The re-modulated velocity v_{out} is then sent to the *function re-mapping* block in order to project v_{out} from the path natural coordinate domain to the time domain.

5) Function re-mapping block: The last step for the generation of the reference task velocity concerns a function re-mapping v_{out} from the path natural coordinate domain to the time domain. For such a purpose, the time interval $dt_{i,i+1}$ between the i^{th} and i^{th+1} path points can be computed as follows:

$$dt_{i,i+1} = \frac{d_s}{v_{mean,i}} \tag{3}$$

where the spatial step d_s is fixed as defined in Section II-A1, and the mean velocity $v_{mean,i}$ can be computed as follows:

$$v_{mean,i} = \frac{v_{out,i+1} + v_{out,i}}{2} \tag{4}$$

where $v_{out,i}$ and $v_{out,i+1}$ correspond to the i^{th} and i^{th+1} path points, respectively. Since the discretization step d_s is fixed along all the path, the time step $dt_{i,i+1}$ will be variable depending on the instantaneous velocities $v_{out,i}$ and $v_{out,i+1}$. The execution time is finally imposed to the each point of the path, making it possible to exploit the computed reference velocity.

The path with the derived time-law is then given to the *execution* block for the task realization.

B. Execution block

In order to execute the planned motion (as derived in Section II-A), the *Dynamic Movement Primitives* (DMP) framework [20] has been employed. This method well suits a path reproduction task, either in Cartesian or joint space, approximating the trajectory using the given via-points, rather than interpolating them. DMP are modelled as a mass-spring-damper critical system under the action of a (time independent) non-linear forcing term [21]. In such a way, DMP have intrinsic positional smoothing of noises and discontinuities and velocity reduction at sharp edges. The non-linear forcing term has to be tuned for the specific path learning and execution (*i.e.*, correctly reproducing the target motion) [22]. The DMP framework gives as an output the Cartesian trajectory to be executed by the robot velocity controller, providing to it the position and velocity reference, $\mathbf{y}_{out}$ and $\dot{\mathbf{y}}_{out}$, respectively. The Cartesian trajectory defined by the DMP approach is finally given to the trajectory tracking robot controller. The Cartesian reference is projected into joint space, computing the joint reference signals. A trajectory tracking PID controller has been implemented and tuned on the basis of [23], making the robot execute the target task.

It is important to underline that some approaches exploiting the DMP framework to modulate the robot velocity during the task execution are available. However, such approaches mainly deal with the velocity discontinuities [12], [24]. In this paper, the DMP framework is fed by the *pre-processing* block with a path-based reference velocity allowing to modulate the robot velocity accordingly with the encountered geometrical features along the target path.

III. Pairwise Preferences-based Optimization

As described in Section II, the employed velocity planner has to be tuned in order to optimize the task execution (*i.e.*, maximizing the task quality). In this paper, the Fuzzy Logic output membership function parameters v_{vs}, v_{s1}, v_{s2}, v_{m1}, v_{m2}, v_{f1}, v_{f2} and the acceleration/deceleration limit parameter a_{max} are considered as tuning parameters. In fact, such parameters are the ones affecting the task quality the most in the considered sealant deposition task.

The experiment-driven active preference learning algorithm GLISp (recently developed by some of the authors in [18]) is adopted to iteratively suggest a sequence of parameters for the velocity planner to be tested and compared. For the sake of completeness, we review the GLISp algorithm in this section.

To compact the notation, all the tuning parameters are collected in the vector $\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$, where Θ is a given bounded set in which the optimal parameters are sought.

A. Building a surrogate function from preferences

The first step of the GLISp algorithm is to build a surrogate function describing the observed preferences.

Formally, given two possible parameters θ_1 and θ_2, we define the *preference function* $\pi : \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_\theta} \to \{-1, 0, 1\}$ as

$$\pi(\theta_1, \theta_2) = \begin{cases} -1 & \text{if } \theta_1 \text{ "better" than } \theta_2 \\ 0 & \text{if } \theta_1 \text{ "as good as" } \theta_2 \\ 1 & \text{if } \theta_2 \text{ "better" than } \theta_1. \end{cases} \tag{5}$$

Assume that $N \geq 2$ samples $\{\theta_1 \ \dots \ \theta_N\}$ of the decision vector are generated, with $\theta_i, \theta_j \in \mathbb{R}^{n_\theta}$ such that $\theta_i \neq \theta_j$, $\forall i \neq j$, $i, j = 1, \dots, N$. For each of these parameters, an experiment is performed and the user has provided a *preference vector* $B = [b_1 \ \dots \ b_M]^T \in \{-1, 0, 1\}^M$ with

$$b_h = \pi(\theta_{i(h)}, \theta_{j(h)}), \tag{6}$$

where M is the number of expressed preferences, $h \in \{1, \dots, M\}$, $i(h), j(h) \in \{1, \dots, N\}$, $i(h) \neq j(h)$. Note that the element b_h of vector B represents the preference expressed by the user between the experimental performance achieved with parameters $\theta_{i(h)}$ and $\theta_{j(h)}$.

The observed preferences are then used to learn a surrogate function $\hat{J} : \mathbb{R}^{n_\theta} \to \mathbb{R}$ of an (unknown) underlying performance index J. The surrogate $\hat{J}$ is parametrized as the following linear combination of Radial Basis Functions (RBFs):

$$\hat{J}(\theta) = \sum_{k=1}^{N} \beta_k \phi(\gamma d(\theta, \theta_i)), \tag{7}$$

where $d : \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_\theta} \to \mathbb{R}$ is the squared Euclidean distance

$$d(\theta, \theta_i) = \|\theta - \theta_i\|_2^2, \tag{8}$$

$\gamma > 0$ is a scalar parameter, $\phi : \mathbb{R} \to \mathbb{R}$ is an RBF, and $\beta = [\beta_1 \ \dots \ \beta_N]^T$ are the unknown coefficients to be computed based on the available user's preferences. Examples of RBFs are $\phi(\gamma d) = \frac{1}{1+(\gamma d)^2}$ (*inverse quadratic*) and $\phi(\gamma d) = e^{-(\gamma d)^2}$ (*Gaussian*) (see [25] for more examples).

According to the preference relation (5), the surrogate $\hat{J}$ has to satisfy the constraints:

$$\begin{array}{ll} \hat{J}(\theta_{i(h)}) \leq \hat{J}(\theta_{j(h)}) - \sigma + \varepsilon_h & \text{if } \pi(\theta_{i(h)}, \theta_{j(h)}) = -1 \\ \hat{J}(\theta_{i(h)}) \geq \hat{J}(\theta_{j(h)}) + \sigma - \varepsilon_h & \text{if } \pi(\theta_{i(h)}, \theta_{j(h)}) = 1 \\ |\hat{J}(\theta_{i(h)}) - \hat{J}(\theta_{j(h)})| \leq \sigma + \varepsilon_h & \text{if } \pi(\theta_{i(h)}, \theta_{j(h)}) = 0 \end{array} \tag{9}$$

for all $h = 1, \dots, M$, where $\sigma > 0$ is a given tolerance and ε_h are positive slack variables which are used to relax the preference constraints. Constraint infeasibility might be due to an inappropriate selection of the RBF (namely, poor flexibility in the parametric description of the surrogate $\hat{J}$) and/or inconsistent assessments done by the user.

Based on the above preference constraints, the coefficient vector β describing the surrogate $\hat{J}$ are obtained by solving the Quadratic Programming (QP) problem

$$\begin{array}{ll} \min_{\beta, \varepsilon} & \sum_{h=1}^{M} \varepsilon_h + \frac{\lambda}{2} \sum_{k=1}^{N} \beta_k^2 \\ \text{s.t.} & \sum_{k=1}^{N} (\phi(\gamma d(\theta_{i(h)}, \theta_k) - \phi(\gamma d(\theta_{j(h)}, \theta_k)) \beta_k \\ & \quad \leq -\sigma + \varepsilon_h, \qquad \forall h : \ b_h = -1 \\ & \sum_{k=1}^{N} (\phi(\gamma d(\theta_{i(h)}, \theta_k) - \phi(\gamma d(\theta_{j(h)}, \theta_k)) \beta_k \\ & \quad \geq \sigma - \varepsilon_h, \qquad \forall h : \ b_h = 1 \\ & \left| \sum_{k=1}^{N} (\phi(\gamma d(\theta_{i(h)}, \theta_k) - \phi(\gamma d(\theta_{j(h)}, \theta_k)) \beta_k \right| \\ & \quad \leq \sigma + \varepsilon_h, \qquad \forall h : \ b_h = 0 \\ & h = 1, \dots, M \end{array} \tag{10}$$

The scalar $\lambda > 0$ in the cost function (10) is a regularization parameter which guarantees uniqueness in the solution of the QP problem.

B. Acquisition function

Once a surrogate $\hat{J}$ is estimated, this function can be in principle minimized in order to find the optimal parameter vector θ of the velocity planner.

More specifically, the following steps can be followed: (*i*) generate a new sample by pure minimization of the estimated surrogate function $\hat{J}$, *i.e.*,

$$\theta_{N+1} = \arg\min \hat{J}(\theta) \ \text{ s.t. } \ \theta \in \Theta;$$

(*ii*) ask the user to express a preference $\pi(\theta_{N+1}, \theta_N^\star)$, where $\theta_N^\star \in \mathbb{R}^{n_\theta}$ is the best vector of the velocity planner parameters found so far, (*iii*) update the estimate of $\hat{J}$ through (10); and (*iv*) iterate over N. Such a procedure, which only *exploits* the current available observations in finding the optimal parameter vector θ, may easily miss more performing configurations of the velocity planner. A term promoting the *exploration* of the parameter space should thus be considered.

In the GLISp algorithm, an acquisition function is employed to balance exploitation *vs.* exploration when generating the new sample θ_{N+1}. The exploration function is constructed by using the inverse distance weighting (IDW) function $z : \mathbb{R}^{n_\theta} \to \mathbb{R}$ defined by

$$z(\theta) = \begin{cases} 0 & \text{if } \theta \in \{\theta_1, \dots, \theta_N\} \\ \tan^{-1}\left(\frac{1}{\sum_{i=1}^{N} w_i(\theta)}\right) & \text{otherwise} \end{cases} \tag{11}$$

where $w_i(\theta) = \frac{1}{d^2(\theta, \theta_i)}$. Clearly $z(\theta) = 0$ for all parameters already tested, and $z(\theta) > 0$ in $\mathbb{R}^{n_\theta} \setminus \{\theta_1, \dots, \theta_N\}$. The arc tangent function in (11) avoids that $z(\theta)$ gets excessively large far away from all sampled points.

Then, given an exploration parameter $\delta \geq 0$, the *acquisition function* $a : \mathbb{R}^{n_\theta} \to \mathbb{R}$ is constructed as

$$a(\theta) = \frac{\hat{J}(\theta)}{\Delta \hat{J}} - \delta z(\theta), \tag{12}$$

where

$$\Delta \hat{J} = \max_i \{\hat{J}(\theta_i)\} - \min_i \{\hat{J}(\theta_i)\}$$

is the range of the surrogate function on the samples in $\{\theta_1,\ldots,\theta_N\}$ and is used in (12) as a normalization factor to simplify the choice of the exploration parameter δ.

As discussed below, given a set $\{\theta_1,\ldots,\theta_N\}$ of samples and a vector B of preferences defined by (6), the next parameter θ_{N+1} of the velocity planner to test is computed as the solution of the (non-convex) optimization problem

$$\theta_{N+1} = \arg\min_{\theta\in\Theta} a(\theta). \tag{13}$$

C. Including acceptability criteria

In the construction of the acquisition function $a(\theta)$ discussed in the previous section, the following user's satisfaction criteria for the executed task were not considered: general acceptability, deposition vibrations, straight lines homogeneity, large curves homogeneity, tight curves homogeneity, very tight curves homogeneity. However, the satisfaction of the user on the executed task can be a useful information to be exploited in the choice of the next parameter θ_{N+1}, thus penalizing configurations of the velocity planner which are expected to provide unsatisfactory performance.

In order to include this information, six independent binary-classification Gaussian Processes (one for each of the satisfaction criteria mentioned above) are trained based on the user's satisfaction of previous experiments. Each Gaussian Process maps the parameter vector θ into the probability of achieving satisfactory quality performance. If this probability is below a given threshold (50% in the considered application) a penalty function is then added to the acquisition function $a(\theta)$.

IV. Experimental results

A complete video showing the implementation of the proposed methodology is available at https://youtu.be/4-cjMvqfg5c. The proposed video shows experiments including several deposition paths, together with the resulting deposition velocity profiles obtained by the proposed path-based velocity planner (which are compared with the ones obtained using a constant velocity trajectory feeding the DMP block). The video also shows the evolution of the preference of the operator among the optimization iterations, and the acceptability judgments.

A. Setup description

Figure 1 shows the reference sealing task (*i.e.*, deposition task). The experimental setup consists of a Franka EMIKA panda robot, equipped with a MAKITA sealing gun. The robot is controlled exploiting its torque control (control frequency: 1000 Hz). A servomotor (controlled with Arduino) has been embedded in the *ad hoc* designed flange to activate/deactivate the deposition. The proposed velocity planner and the servomotor control have been implemented in the ROS environment.

B. Parameters setting

The steering parameter *steer* ranges defining the Fuzzy Logic input membership function (*i.e.*, allowing to classify the geometrical path features as in Figure 4a) has been defined as follows: *very tight curve steer* $= [0.22,\ 0.5]$, *tight curve steer* $= [0.1,\ 0.45]$ *large curve steer* $= [0.02,\ 0.22]$, *straight line steer* $= [0,\ 0.1]$. The ranges for the considered optimization variables has been defined as follows: $v_{vs} \in [20,\ 40]$ mm/s, $v_{s1} \in [10,\ 26]$ mm/s, $v_{s2} \in [26,\ 60]$ mm/s, $v_{m1} \in [20,\ 40]$ mm/s, $v_{m2} \in [60,\ 90]$ mm/s, $v_{f1} \in [50,\ 70]$ mm/s, $v_{f2} \in [70,\ 100]$ mm/s, $a_{max} \in [10,\ 80]$ mm/s^2.

C. Task optimization procedure

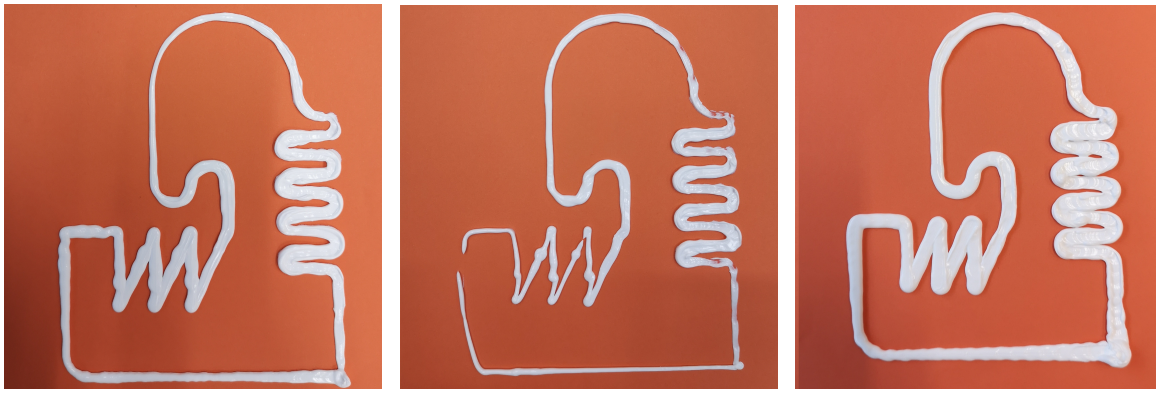
As explained in Section III, the pairwise preferences-based optimization takes as an input the user's preference on the best task execution. Therefore, at each iteration of the tuning algorithm, the expert operator performs a pairwise comparison on the global task quality between two experimental depositions (the last trial and the best one so far achieved in the optimization process). In addition, for the executed task, the user also provides a judgement (acceptable or not acceptable) on each of the following criteria: general acceptability, deposition vibrations, straight lines homogeneity, large curves homogeneity, tight curves homogeneity, very tight curves homogeneity. In such a way, the algorithm is able to discriminate between suitable and unsuitable depositions.

The optimization algorithm is initialized with 5 random samplings. All the 5 experimental trials are pairwise compared. The pairwise comparisons, together with the evaluation of the acceptability criteria, are given to the algorithm to start the optimization process. As soon as a new deposition is available, it is pairwise compared with the best one so far, providing the judgement to the algorithm. In addition, acceptability criteria are also evaluated.

D. Achieved results

Figure 5 shows the performed depositions for experimental trials 7, 10, and 19 (being the optimized deposition). The proposed path (along X and Y coordinates) includes the defined geometry path features (*i.e.*, *straight lines*, *large curves*, *tight curves*, *very tight curves*). The robot velocity has, therefore, to be adapted in order to properly execute the task. It is possible to note the improvements achieved by the optimized iteration 19 in terms of reduced vibrations and homogeneity of the deposition, having iteration 7 showing unsatisfactory deposition in 4 acceptability criteria, and iteration 10 showing unsatisfactory deposition for all the defined acceptability criteria (see linked video for the complete analysis of the acceptability judgement).

To evaluate the performance, the proposed optimized path-based velocity planner (PBVP) has been compared with the programming by demonstration (PbD) approach in [4]. 5 task demonstrations have been performed by the operator. A Hidden Markov Model approach has then be used in order to select the most representative demonstration. This selected demonstration (pre- and post-processed as described in [4]) has been used to feed the DMP algorithm, providing to the robot controller the reference trajectory for the task execution. In addition, a manual tuning of the parameters of the PBVP has been performed. To avoid any influence on the tuning process,



Experimental trial 7 Experimental trial 10 Experimental trial 19

Fig. 5: Experimental trial 7, 10, and 19 are shown. Iteration 7 shows an unsatisfactory deposition for general acceptability, deposition vibrations, large curves, and tight curves criteria. Iteration 10 shows an unsatisfactory deposition for all the defined acceptability criteria. It is possible to note the improvements achieved at iteration 19 in terms of reduced vibrations and homogeneity of the deposition.

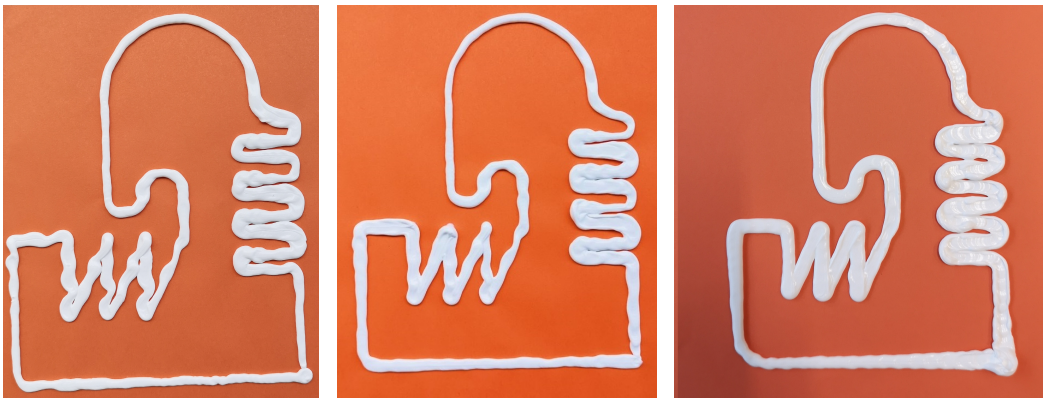
the manual tuning has been performed before the execution of the presented optimization procedure. 30 experimental trials have been required in order to get to an acceptable quality. Figure 6 shows the improved performance (in terms of homogeneity of the deposition and reduction of deposition vibrations) obtained exploiting the proposed optimization methodology, within a reduced number of experimental trials. To quantify the quality of the deposition that has been achieved with the PbD method, the manual tuning of the PBVP parameters, and the optimized tuning of the PBVP parameters, the thickness of the deposited material has been measured in 60 positions along the path. Having a nozzle diameter of 5 mm (*i.e.*, defining the reference thickness of the deposited material), the following results have been achieved in terms of mean and standard deviation for the PbD approach: mean thickness 3.3 mm, standard deviation 1.6 mm; for the manual tuned PBVP: mean thickness 3.5 mm, standard deviation 1.4 mm; and for the the optimized PBVP: mean thickness 4.7 mm, standard deviation 0.3 mm; demonstrating the improved deposition quality in terms of homogeneity. Time-based metrics have not been considered for the evaluation of the task. In fact, the total execution time is a function of the robot velocity along the path. However, the robot velocity affects the task quality (based on the encountered geometrical feature), and it has to be optimized in order to maximize the final achieved quality. Therefore, while it is possible to imposed soft time-based constraints, the main aim of such tasks is the maximization of their quality instead of the minimization of the total execution time.

It has to be noted that the PbD approach in [4] has been selected due to the fact that it doesn't involve the use of external sensors (*e.g.*, cameras). In fact, the main aim of the presented control and optimization framework is to provide the operator with an intuitive and easy to setup/use methodology for task knowledge transfer to the robot. While the proposed PBVP is capable to generalize its performance along different paths (as shown in the linked video, where several paths have been tested), a PbD approach is capable of providing a reference trajectory only for the demonstrated path. Generalizable performance can be achieved by PbD approaches, however requiring a lot of experiments and additional processing and implementation efforts. Nevertheless other approaches can be exploited to transfer the operator's task knowledge to the robot, their complexity in the setup and teaching procedures [6], or their huge number of parameters to be set [26], doesn't provide a feasible solution, especially for the considered deposition task.

V. Conclusions

In this paper, a path-based velocity planner is developed to modulate the robot velocity based on the encountered geometrical features. The path-based velocity planner parameters are optimized by adopting a pairwise preferences-based optimization, maximizing the task performance only exploiting the user's qualitative feedback. A Franka EMIKA panda robot is used as a test platform to perform a sealing task, validating the proposed framework in comparison with a programming by demonstration approach. Experimental results show the effectiveness of the proposed control and optimization approach, providing an intuitive and easy implementable methodology, not requiring for external sensors.

Future work is devoted to derive a modeling correlating the path curvature (*i.e.*, defining the geometrical path features) and the deposition velocity w.r.t. the user's preferences (*i.e.*, the quality of the deposition task). Such model will be derived optimizing the deposition velocity (exploiting the preferences-based optimization algorithm in this paper) for a set of curvature values, independently. Aggregating the data, the complete model will then be employed in order to optimize the deposition velocity along a generic complex path. The pairwise preferences-based optimization algorithm will be also improved in order to consider additional judgements criteria (*e.g.*, quantitative criteria based on a laser scanner system providing online measurements related to the thickness of the deposited material, enabling to implement a feedback controller to account for deposition errors). In addition, the deposition flow will be taken into account as an optimization



PbD approach [4] PBVP Manual tuning PBVP Optimized tuning

Fig. 6: Deposition performance achieved by the PbD approach, the PBVP manual tuning, and the PBVP preferences-based optimized tuning. The latter lead to a more homogeneous deposition and smaller vibrations.

variable in the advanced setup within the H2020 CS2 ASSASSINN project.

REFERENCES

[1] L. Roveda, N. Castaman, P. Franceschi, S. Ghidoni, and N. Pedrocchi, "A control framework definition to overcome position/interaction dynamics uncertainties in force-controlled tasks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6819–6825.

[2] L. Roveda, S. Haghshenas, M. Caimmi, N. Pedrocchi, and L. Molinari Tosatti, "Assisting operators in heavy industrial tasks: on the design of an optimized cooperative impedance fuzzy-controller with embedded safety rules," *Frontiers in Robotics and AI*, vol. 6, p. 75, 2019.

[3] P. Zawadzki and K. Żywicki, "Smart product design and production control for effective mass customization in the industry 4.0 concept," *Management and production engineering review*, vol. 7, 2016.

[4] L. Roveda, M. Magni, M. Cantoni, D. Piga, and G. Bucca, "Human–robot collaboration in sensorless assembly task learning enhanced by uncertainties adaptation via bayesian optimization," *Robotics and Autonomous Systems*, vol. 136, p. 103711, 2021.

[5] Z. Chen, J. Wang, S. Li, J. Ren, Q. Wang, Q. Cheng, and W. Li, "An optimized trajectory planning for welding robot," *Materials Science and Engineering*, vol. 324, no. 1, p. 012009, 2018.

[6] A. Vakanski, F. Janabi-Sharifi, and I. Mantegh, "An image-based trajectory planning approach for robust robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 98, pp. 241–257, 2017.

[7] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion and operation planning of robotic systems*, pp. 3–27, 2015.

[8] G. Trigatti, P. Boscariol, L. Scalera, D. Pillan, and A. Gasparetto, "A new path-constrained trajectory planning strategy for spray painting robots-rev. 1," *The International Journal of Advanced Manufacturing Technology*, vol. 98, no. 9, pp. 2287–2296, 2018.

[9] L. Xiang, X. Xie, and X. Lu, "An optimal trajectory control strategy for underwater welding robot," *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 12, no. 1, pp. JAMDSM0008–JAMDSM0008, 2018.

[10] J. De Maeyer, B. Moyaers, and E. Demeester, "Cartesian path planning for arc welding robots: Evaluation of the descartes algorithm," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2017, pp. 1–8.

[11] L. Anderlucci, "Smooth trajectory planning for anthropomorphic industrial robots employed in continuous processes," *Politecnico di Torino*, 2019.

[12] T. Kulvicius, K. Ning, M. Tamosiunaite, and F. Worgötter, "Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 145–157, 2011.

[13] C. Daniel, M. Viering, J. Metz, O. Kroemer, and J. Peters, "Active reward learning."

[14] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *arXiv preprint arXiv:1706.03741*, 2017.

[15] M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh, "Learning reward functions by integrating human demonstrations and preferences," *arXiv preprint arXiv:1906.08928*, 2019.

[16] M. Zhu, A. Bemporad, and D. Piga, "Preference-based MPC calibration," *arXiv: 2003.11294*, 2020.

[17] J. Choi, C. Dance, J.-e. Kim, K.-s. Park, J. Han, J. Seo, and M. Kim, "Fast adaptation of deep reinforcement learning-based navigation skills to human preference," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3363–3370.

[18] A. Bemporad and D. Piga, "Global optimization based on active preference learning with radial basis functions," *Machine Learning*, vol. 110, pp. 417–448, 2020.

[19] T. J. Ross *et al.*, *Fuzzy logic with engineering applications*. Wiley Online Library, 2004, vol. 2.

[20] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *arXiv preprint arXiv:2102.03861*, 2021.

[21] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," Tech. Rep., 2002.

[22] ——, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1398–1403.

[23] L. Roveda, M. Forgione, and D. Piga, "Robot control parameters autotuning in trajectory tracking applications," *Control Engineering Practice*, vol. 101, p. 104488, 2020.

[24] B. Nemec and A. Ude, "Action sequencing using dynamic movement primitives," *Robotica*, vol. 30, no. 5, p. 837, 2012.

[25] A. Bemporad, "Global optimization via inverse distance weighting and radial basis functions," *Computational Optimization and Applications*, vol. 77, pp. 571–595, 2020.

[26] J. Huang, P. Hu, K. Wu, and M. Zeng, "Optimal time-jerk trajectory planning for industrial robots," *Mechanism and Machine Theory*, vol. 121, pp. 530–544, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X17302914