Dynamical System-based Imitation Learning for Visual Servoing using the Large Projection Formulation

Antonio Paolillo[1], Paolo Robuffo Giordano[2], Matteo Saveriano[3]

***Abstract*— Nowadays ubiquitous robots must be adaptive and easy to use. To this end, dynamical system-based imitation learning plays an important role. In fact, it allows to realize stable and complex robotic tasks without explicitly coding them, thus facilitating the robot use. However, the adaptation capabilities of dynamical systems have not been fully exploited due to the lack of closed-loop implementations making use of visual feedback. In this regard, the integration of visual information allows higher flexibility to cope with environmental changes. This work presents a dynamical system-based imitation learning for visual servoing, based on the large projection task priority formulation. The proposed scheme enables complex and stable visual tasks, as demonstrated by a simulation analysis and experiments with a robotic manipulator.**

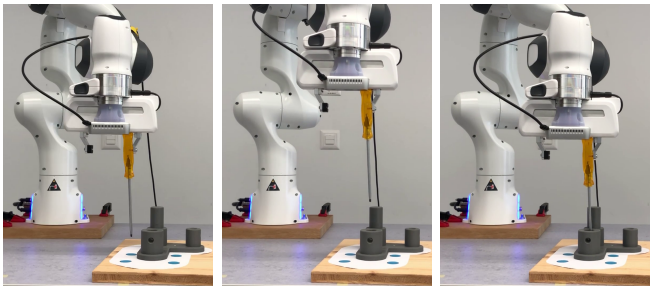

Fig. 1. In complex robotic tasks, such as tool insertion in narrow locations, dynamical systems allow to easily program the desired motion, and visual servoing allows to adapt the learned motion to changes of the environment.

I. Introduction

The level of ubiquity reached by robots imposes strict requirements of adaptability, interactivity, and practicality of use. Today robots have to be easily accessible, also to unskilled users. For this reason, the recent research effort is focused on providing robots with friendly control frameworks. In industries, for instance, the communication between technicians and machines can be eased with gesture-based interactions [1] or other sorts of intuitive interfaces [2].

In general, to allow non-expert users to conveniently use robots, coding duties have to be lightened or even avoided. This objective has been considered by Imitation Learning (IL) methods that undertake to learn a task from data, rather than explicitly coding it. Such a form of machine learning, otherwise known as programming-by-demonstration [3] or learning-from-demonstration [4], avoids to explicitly program the robotic task, thus facilitating the large deployment of robots. Among others, Dynamical Systems (DS)-based IL allows to implement the data imitation strategy preserving the stability properties of the original controllers [5].

For instance, consider a common insertion task with a robotic manipulator as shown in Fig. 1. The task can be kinesthetically taught to the robot through manual demonstrations. DSs can be used to learn the robot joints motion to realize the task in a stable way without coding it, i.e. imitating the stored demonstrations. An obvious improvement of this kind of methods is the inclusion of exteroception, such as vision, which adds adaptability to the learning framework. In the considered scenario, in fact, a visual feedback would inform the control about possible changes of the insertion location. In this context, Visual Servoing (VS) [6], [7] represents a very suitable strategy to realize adaptive DSs. Indeed, VS is a mature technique and widely used to control robots from visual information. Furthermore, its formalism can be easily interpreted as a DS and, thus, conveniently combined with this class of IL. In a previous work, we have shown that the integration of DS and VS brings advantages to both techniques [8]. On the one side, it allows to include visual exteroception into the DS, thus adding adaptability to environmental changes. On the other, it allows to avoid specific programming of additional tasks into the VS.

We here propose a novel strategy for DS-based Imitation Learning Visual Servoing (ILVS) built on the large null-space projection formulation originally proposed in [9]. After reviewing the related literature (Sec. II) and recalling the basic concepts used in our work (Sec. III), we describe the details of our method in Sec. IV. Results carried out with simulations and experiments are analyzed in Section V; Section VI concludes the papers.

II. Related Work

DS-based IL methods effectively generate stable kinematic motion of robots [10]–[15]. The simplest way to retrieve stability is the use of a *clock signal* to suppress possibly unstable dynamics [10]–[12]. Alternatively, stability constraints (e.g., derived from a proper Lyapunov function) can be enforced during the learning process using constrained optimization [5], [13]. In other approaches, the learned input-output mapping is forced to be a diffeomorphism, and a stable DS can be obtained using its Jacobian [14], [15].

DSs implementing closed-loop strategies with proprioception have adaptation skills as they are capable of passive

This work was supported by the European Union's Horizon 2020 (1-SWARM, Grant No. 871743), the Hasler foundation (EViRCo), and the Italian Ministry of the University (iNEST, Grant No. ECS 00000043).

[1]Dalle Molle Institute for Artificial Intelligence (IDSIA), USI-SUPSI, Lugano, Switzerland antonio.paolillo@idsia.ch

[2]CNRS, Univ Rennes, Inria, IRISA, Rennes, France prg@irisa.fr

[3]Department of Industrial Engineering (DII), University of Trento, Trento, Italy matteo.saveriano@unitn.it

interactions [16]. However, their adaptation potential can be fully reached using exteroception like vision. In this regard, VS laws [6], [7] can be interpreted as DSs and used to integrate visual feedback into IL schemes [8]. The combination of VS and DS allows to generate complex motion, adapt to environmental changes, and preserve stability.

On the VS side, the basic law performance can be augmented with higher adaptability and flexibility resorting to, e.g., planning [17], [18], model predictive control [19], [20] or optimization-based control [21]–[23]. Planning approaches allow to perform complex VS tasks, but they are normally computed offline and have limited adaptation capabilities. In predictive controllers, instead, one has to balance the anticipatory behavior with the computational cost. This balance can be loosened using a set of pre-computed solutions [24]. These optimization-based solutions need however to be specifically designed. Exploiting the IL paradigm in a VS scheme would instead allow to easily learn complex visual tasks from few demonstrations. Examples are: Gaussian Mixture Regression (GMR) to learn the pixel to camera motion mapping [25]; deep neural networks to imitate the surgeon in eye operations [26]; random forests queried to generate commands for clothes manipulation [27]. Vision-based IL is effective in learning fine operations [28] and can generalize the learned motion to different surfaces [29]. In contrast to black-box approaches, our method exploits the controller structure to accurately imitate a trajectory while ensuring stability. Stability guarantees can be obtained if the controller analytical structure is maintained and visual IL is used to provide a reference trajectory [30], [31]. In [32] the reference of a classical image-based VS is inferred from demonstrations and further refined by an optimizer.

In our previous work [8], we have integrated 3 existing DS-based IL methods in the VS scheme. Here, we exploit a specific task priority control paradigm, using the large projection formulation originally introduced in [9]. The proposed approach conveniently separates the data imitation from the maintenance of closed-loop stability. In practice, the higher priority task is designed so as to guarantee closed-loop stability; the data imitation is handled as secondary task. On the learning side, the scheme greatly simplifies the problem as it allows to use consolidated inference approaches without worrying about stability constraints. On the control side, a modified version of the large projection formulation grants enough degrees-of-freedom to perform complex VS tasks.

III. Background

A. Dynamical systems for imitation learning

In IL, control affine DS can be used to map a state vector $\boldsymbol{x} \in \mathbb{R}^n$ (e.g., the end-effector position), a control input $\boldsymbol{u} \in \mathbb{R}^n$, the time $t \in \mathbb{R}^+$ to the state time derivative $\dot{\boldsymbol{x}} \in \mathbb{R}^n$:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{u}(\boldsymbol{x}, t). \tag{1}$$

When DSs are used to generate point-to-point motions, one needs to guarantee the stability of (1). The approach in [13] learns $\boldsymbol{f}$ in (1) to reproduce a set of demonstrations and then compute a stabilizing controller $\boldsymbol{u}$ using a control Lyapunov function. Other methods constrain $\boldsymbol{f}$ to be stable and thus set $\boldsymbol{u} = \boldsymbol{0}$ [5], [33]. The work in [10], [11] assume linear and stable $\boldsymbol{f}$, and use $\boldsymbol{u}$ to *reshape* the linear motion and follow the demonstrations. Making $\boldsymbol{u}$ vanish with time allows to retrieve the stability. In this work, we use a control law inspired by (1) to guarantee convergence and follow demonstrated trajectories together.

B. Visual servoing using the large projection operator

VS enables robot control using the feedback encoded in sets of visual features [6], [7]. In image-based VS, such features are directly extracted from the images, using image processing [34] or more sophisticated methods, e.g. neural models [35]. With eye-in-hand configurations, fixed targets and non-moving scenes, the classic VS writes as $\boldsymbol{v} = -\lambda \hat{\boldsymbol{L}}^{+} \boldsymbol{e}$. This law generates the velocity $\boldsymbol{v} \in \mathbb{R}^6$ that drives the camera to a desired pose, zeroing the error $\boldsymbol{e} \in \mathbb{R}^f$ between the visual features and their corresponding targets. In the VS law, λ is a positive scalar gain tuning the exponential decay of the error. Instead, $\hat{\boldsymbol{L}}^{+} \in \mathbb{R}^{6 \times f}$ is the pseudo-inverse of an approximation of the so-called interaction matrix [6] that relates the features motion to the camera velocity. The approximation, denoted with the hat over the matrix, is in general due to unknown 3D quantities such as the feature depths. Other tasks can be achieved in parallel to VS using the null-space projection [36]:

$$\boldsymbol{v}_e = -\lambda \hat{\boldsymbol{L}}^{+} \boldsymbol{e} + \boldsymbol{P} \boldsymbol{\sigma} \tag{2}$$

where $\boldsymbol{P} = (\boldsymbol{I}_6 - \hat{\boldsymbol{L}}^{+} \hat{\boldsymbol{L}})$ is the projector in the null-space of the VS task, being $\boldsymbol{I}_6$ is the 6×6 identity matrix; $\boldsymbol{\sigma}$ is the velocity command needed to realize a secondary task. The number and the type of features used in the VS scheme are chosen in accordance with the desired task to be executed. In general, the dimension of $\boldsymbol{e}$ needs to be large enough in order to control all the 6 camera degrees of freedom. Therefore, the interaction matrix needs to have full rank and, thus, $f \geq 6$. However, full rank of $\hat{\boldsymbol{L}}$ implies no room for projection in its null-space. A possibility to circumvent this problem is to consider the regulation of the error norm $\eta = \|\boldsymbol{e}\|$ in lieu of the full vector $\boldsymbol{e}$ in the control law, as proposed in [9], [37]:

$$\boldsymbol{v}_\eta = -\lambda \, \eta \, \hat{\boldsymbol{L}}_\eta^{+} + \boldsymbol{P}_\eta \, \boldsymbol{\sigma} \tag{3}$$

where both $\hat{\boldsymbol{L}}_\eta^{+}$ and $\boldsymbol{P}_\eta$ can be computed in closed-form, i.e.:

$$\hat{\boldsymbol{L}}_\eta^{+} = \eta \frac{\hat{\boldsymbol{L}}^\top \boldsymbol{e}}{\boldsymbol{e}^\top \hat{\boldsymbol{L}} \hat{\boldsymbol{L}}^\top \boldsymbol{e}} \quad \text{and} \quad \boldsymbol{P}_\eta = \boldsymbol{I}_6 - \frac{\hat{\boldsymbol{L}}^\top \boldsymbol{e} \boldsymbol{e}^\top \hat{\boldsymbol{L}}}{\boldsymbol{e}^\top \hat{\boldsymbol{L}} \hat{\boldsymbol{L}}^\top \boldsymbol{e}}. \tag{4}$$

The advantage of (3) is that the primary task has dimension 1, and the remaining 5 degrees of freedom can be exploited to achieve a secondary task. Indeed, the VS task is actually achieved since, at steady state, $\boldsymbol{e}$ converges to zero for $\eta \to 0$. However, during the transient, the camera motion is not fully constrained, leaving space to other tasks to be accomplished. The main drawback is that (3) is singular if $\boldsymbol{e}$ is null. Thus, a switching strategy is needed to replace (3) with the classic law (2) in the neighborhood of the singularities [9]:

$$\boldsymbol{v} = \alpha(\eta) \, \boldsymbol{v}_\eta + \big(1 - \alpha(\eta)\big) \, \boldsymbol{v}_e \tag{5}$$

where the switching function $\alpha(\eta)$ is defined as follows:

$$\alpha(\eta) = \begin{cases} 1 & \text{if } \eta_1 < \eta \\ \big(\bar{\alpha}(\eta) - \alpha_0\big) / \left(\alpha_1 - \alpha_0\right) & \text{if } \eta_0 \leq \eta \leq \eta_1 \\ 0 & \text{if } \eta < \eta_0 \end{cases} \quad (6)$$

with $\bar{\alpha}(\eta) = (1 + \exp(-\bar{\alpha}_0 \frac{\eta - \eta_0}{\eta_1 - \eta_0}) + \bar{\alpha}_1)^{-1}$. The hyper-parameters $\bar{\alpha}_0$, $\bar{\alpha}_1$, α_0, α_1, η_0, η_1 are set to design a smooth transition of α in $[0, 1]$. See [9] and Sec. V for more details.

IV. Proposed Approach

We build on the VS law (5)–and thus (3)–to realize our ILVS strategy, achieving the desired visual task while following demonstrated trajectories. We posit that the rationale behind (5) is particularly suitable to integrate DS and VS, since it allows to fulfill the visual task at steady-state and imitate the demonstrations during the transient thanks to the large redundancy granted by (5). In practice, the projected quantity $\boldsymbol{P}_\eta \boldsymbol{\sigma}$ can result in complex camera trajectories as evoked by the DS paradigm. Indeed, the velocity term $\boldsymbol{\sigma}$ can be inferred from the demonstrations using a proper regression technique and inserted in (3) to realize our DS-based ILVS.

We illustrate how to imitate complex camera trajectories from demonstrations using the VS law with the large projection formulation. In particular, we show the adaptation needed to accurately reproduce the stored trajectories, while preserving the controller stability. Thus, we introduce both the control and learning components of our framework, along with the dataset required to perform the imitation strategy.

A. Error norm tracking with velocity imitation

As explained in Sec. III-B, the space left by the VS task using the scalar error norm η allows to realize a large class of secondary tasks specified by the velocity $\boldsymbol{\sigma}$. Thus, we propose to use this secondary task to imitate the demonstrations. However, very complex demonstrations, having a non-exponential (or even non-monotonic) decrease of η, could not be accurately followed as they would conflict with the primary VS task that, instead, forces a "simple" exponential decay of η. To overcome this issue, we propose to track a desired norm trajectory $\eta_d(t)$ that also needs to be learned:

$$\boldsymbol{v}_\eta = (\dot{\eta}_d - \lambda\varepsilon)\hat{\boldsymbol{L}}_\eta^+ + \boldsymbol{P}_\eta \boldsymbol{\sigma}, \quad (7)$$

thus allowing to implement more complex behaviors for the norm trajectory $\eta_d(t)$. In the proposed law, we define $\varepsilon = \eta - \eta_d$. The desired norm tracking (7) does not affect the stability of the original controller (3) proposed in [9]. In fact, by choosing $V = \varepsilon^2$ as Lyapunov candidate, it is

$$\begin{aligned} \dot{V} &= 2\,\varepsilon\,\dot{\varepsilon} = 2\,\varepsilon\,(\boldsymbol{L}_\eta \boldsymbol{v}_\eta - \dot{\eta}_d) \\ &= 2\varepsilon(\dot{\eta}_d - \lambda\varepsilon)\boldsymbol{L}_\eta \hat{\boldsymbol{L}}_\eta^+ + 2\,\varepsilon \boldsymbol{L}_\eta \boldsymbol{P}_\eta \boldsymbol{\sigma} - 2\,\varepsilon\,\dot{\eta}_d. \end{aligned} \quad (8)$$

Assuming (as usual) a good approximation of the uncertain 3D parameters in interaction matrix, it follows that $\boldsymbol{L}_\eta \hat{\boldsymbol{L}}_\eta^+ = 1$, $\boldsymbol{L}_\eta \boldsymbol{P}_\eta = \boldsymbol{0}$, and (8) reduces to $\dot{V} = -2\lambda\varepsilon^2$ that is strictly negative for $\eta_d \neq \eta$. Thus, $\varepsilon = 0$ is an asymptotically stable equilibrium point in its neighborhood: (7) remains locally stable as the norm regulation (3) [9] and the classic VS [6]. Note that $\boldsymbol{\sigma}$ does not impact the stability and can be learned with no particular constraint, while the learned $\eta_d(t)$ needs to converge to a proper target, as explained in the next section.

Algorithm 1: DS-based ILVS

1: Data processing phase
  $\mathcal{D}, \mathcal{H} \longleftarrow$ `collect_and_process_demos`
2: Learning phase
  $\boldsymbol{r}_\sigma \longleftarrow$ `train_regressor`$(\mathcal{D})$
  $\boldsymbol{r}_\eta \longleftarrow$ `train_regressor`$(\mathcal{H})$
3: Execution phase
  $(\boldsymbol{e}, \eta) \longleftarrow$ `compute_error(image)`
  $\boldsymbol{\sigma} \longleftarrow \boldsymbol{r}_\sigma(\boldsymbol{e}|\mathcal{D})$
  $\dot{\eta}_d \longleftarrow \boldsymbol{r}_\eta(\tau|\mathcal{H})$
  $\eta_d \longleftarrow$ `numerical_integration`$(\dot{\eta}_d)$
  $\boldsymbol{v}_\eta \longleftarrow$ `compute_proposed_law`$(\eta, \eta_d, \dot{\eta}_d, \boldsymbol{\sigma})$
  $\boldsymbol{v}_e \longleftarrow$ `compute_classic_law`$(\boldsymbol{e}, \boldsymbol{\sigma})$
  $\boldsymbol{v} \longleftarrow$ `compute_switching_law`$(\boldsymbol{v}_e, \boldsymbol{v}_\eta)$
  `send_commands`$(\boldsymbol{v})$
  Repeat until $\boldsymbol{e} \to \boldsymbol{0}$

B. Learning VS tasks

We consider data from D demonstrations of VS tasks composed of N samples, consisting in pairs of visual error $\boldsymbol{e}_n^d$ and the corresponding desired camera commands $\boldsymbol{v}_n^d$:

$$\mathcal{D} = \left\{\boldsymbol{e}_n^d,\, \boldsymbol{v}_n^d\right\}_{n=1,d=1}^{N,D}. \quad (9)$$

In computing the visual error, we assume that the target features are known and decided in advance. Its dimension, chosen to meet the requirements detailed in Sec. III-B, is also decided beforehand. Thus, using $\mathcal{D}$, a camera velocity can be simply inferred through the current measurement of the visual error. Recalling that our control paradigm (7) requires also information about the error norm, the following dataset

$$\mathcal{H} = \left\{\tau_n^d,\, \dot{\eta}_n^d\right\}_{n=1,d=1}^{N,D} \quad (10)$$

is derived from $\mathcal{D}$ by computing the desired error norm $\eta_n^d = \|e_n^d\|$ and its time derivative $\dot{\eta}_n^d$ for each value of n and d. The dataset $\mathcal{H}$ also contains τ_n^d, a time-like variable that linearly increases from $\tau_0^d = 0$ to $\tau_N^d = 1$ for each demonstration d.

The law (7) greatly simplifies the learning of closed-loop stable VS, as it decouples the velocity imitation from the law convergence. Thus, one can learn separately desired camera velocity and error norm with standard regression techniques.

Using a proper regression function, the velocity $\boldsymbol{\sigma}$ required to follow the demonstrations can be inferred from the dataset (9) and realized by our strategy (7) as secondary task. Such a regression function is actually a map that, given the dataset $\mathcal{D}$, returns an approximation of the velocity imitating the demonstrations for a certain value of the visual error:

$$\boldsymbol{\sigma} = \boldsymbol{r}_\sigma\left(\boldsymbol{e} \,|\, \mathcal{D}\right). \quad (11)$$

The regression in (11) allows to retrieve the desired camera velocity $\boldsymbol{\sigma}$ in closed-loop, i.e., starting from the measured

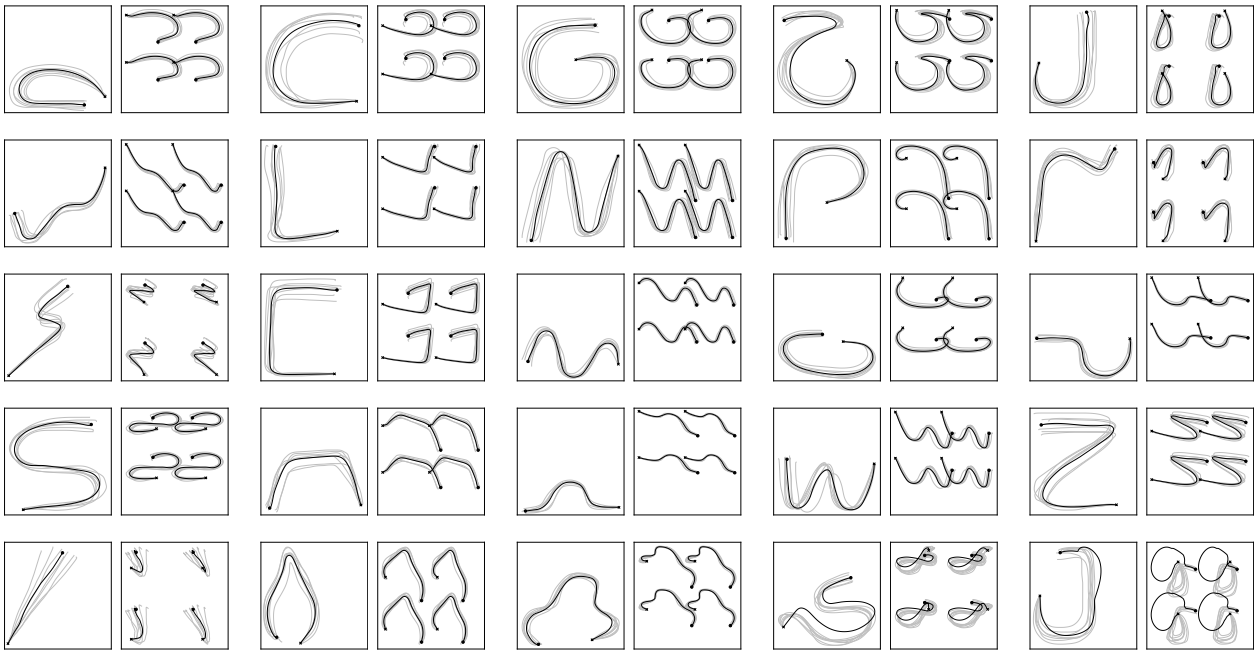

Fig. 2. Simulation analysis: demonstrated motion samples contained in the dataset (gray lines) and an exemplar execution of our method (black line); each sample is composed of camera (left) and visual features motion (right). For visualization purposes, only the top view of the camera motion is shown.

error $\boldsymbol{e}$. This aspect is beneficial as the learning algorithm can adjust the desired velocity to cope with an unexpected variation of the environment (e.g., a change of the target location) or disturb occurring on the camera motion. Furthermore, the proposed control scheme (7) converges to the desired features even if $\boldsymbol{\sigma}$ has a stable equilibrium at $\boldsymbol{e} \neq \mathbf{0}$ or it diverges. The only requirement is that $\boldsymbol{r}_\sigma$ is smooth. Thus, any regression techniques can be used to learn and retrieve it. Notably, we train directly on $\mathcal{D}$ without additional stability constraints that significantly increase the training time and the possibility to fall in poor local minima. In this work, we choose GMR to implement $\boldsymbol{r}_\sigma$ because it produces smooth outputs, is fast enough for online control loops, and the only hyper-parameter (the number of Gaussians) is easy to tune.

Similarly, the desired value of the time derivative of the error norm $\dot{\eta}_d$ is inferred from data as

$$\dot{\eta}_d = \boldsymbol{r}_\eta \left(\tau \,|\, \mathcal{H} \right), \tag{12}$$

and η_d is obtained by numerical integration. In principle, $\dot{\eta}_d$ could also be algebraically retrieved from the learned $\boldsymbol{\sigma}$, but this choice cannot ensure convergence as the output of (11) may diverge or stay in local minima. Note that $\boldsymbol{r}_\eta$, instead, takes as input the monotonically increasing time-like signal $\tau \in [0, 1]$, enabling reliable learning contrary to considering η_d as input. Also, being η_d a scalar, it is easy to obtain multi-modal behaviors, i.e., different values of $\dot{\eta}_d$ for the same η_d, which are difficult to learn. The use of τ solves this issue as there is a unique pair $(\eta_d, \dot{\eta}_d)$ for each τ. Also for $\boldsymbol{r}_\eta$, any proper regressor may be used and we consider GMR.

C. Imitation Learning-based Visual Servoing framework

The whole framework implementing our ILVS strategy is summarized in Algorithm 1. The procedure consists of three main phases; the first two are carried out offline, whereas the latter is executed online. The initial phase regards the data collection and processing required to build the datasets $\mathcal{D}$ and $\mathcal{H}$ used in the learning steps. Thus, as explained in Sec. IV-B, two regression functions (GMRs in our work) are trained to provide the framework with the required demonstrated information. The execution phase is the actual online control loop that realizes our ILVS. Initially, the visual error (and its norm) is computed from the acquired camera image, and used to infer the velocity $\boldsymbol{\sigma}$ imitating the demonstrations. Similarly, also $\dot{\eta}_d$ is inferred from data, and η_d computed through numerical integration. These values are used to compute the velocity command as in the law (7). Using the visual error $\boldsymbol{e}$, also the classic VS law (2) is computed. Thus, the two laws are combined together as in (5) using the switching strategy (6). Finally, the commands are executed and the loop runs again until convergence.

V. Results

A. Simulations

Our simulations are performed on the LASA dataset [5]. This dataset contains 26 uni-modal handwriting motion samples, used as camera lateral trajectories (the vertical displacement is implemented as a constant velocity motion) and augmented with corresponding visual features trajectories as in [8]. The visual features are 4 points, corresponding to the image of the 4 vertexes of a squared visual pattern placed at a known position in front of the camera. Each motion sample contains 7 demonstrations; each demonstration has 1000 points. We heuristically set the switching strategy with $\bar{\alpha}_0 = 12$, $\bar{\alpha}_1 = 6$, $\eta_0 = 10$, $\eta_1 = 25$, $\alpha_0 = 0$ and $\alpha_1 = 1$; the number of Gaussians of the GMRs to 7; the VS gain λ to

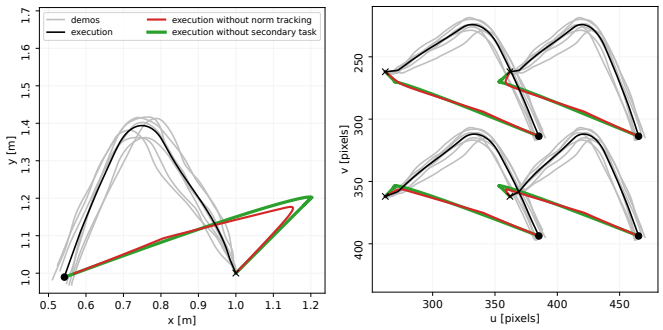

Fig. 3. Simulation analysis of one motion sample: lateral camera (left) and visual features (right) trajectories.

3. Each execution is given a length of 1050 samples. In our study, we consider only the linear motion of the camera, which means that both control and learning steps of our framework deal with only position variables.

The qualitative performance of our method in imitating the dataset trajectories is shown in Fig. 2 for 25 of the samples. For each one, the planar camera motion (on the left) and the visual features motion on the image (right) are shown. In each plot, the black line refers to the trajectory executed by our method; the gray traces are the demonstrations; the black dot indicates the starting point (taken as the average of the demonstration starts); the cross is the final point. The plots show that our strategy succeeds in imitating the demonstrations while converging to the target in most samples, with the same learning and control parameters. For two of them (the last plots in Fig. 2), the performance deteriorates due to the presence of loops in all the demonstrated features motion.

For the sake of clarity of the presentation, plots concerning the remaining 26th sample of the dataset are reported in Fig. 3–6. More into details, Fig. 3 shows that the proposed method allows to achieve the VS task, bringing the camera to convergence, while imitating the dataset trajectories. As explained in Sec. IV, this is made possible thanks to information extracted from the dataset, as shown in Fig. 4 and 5, where the gray, black dashed and black solid lines denote the demonstrations, the GMR inference and the execution, respectively. Note that we run the GMR for 1000 samples, which essentially set the maximum time given to the imitation strategy. After about 900 samples, the system switches to the classic VS law (see the plot of the switch variable α in Fig. 6) and there is no room for null-space tasks. The trajectory of the norm is properly tracked, being it the primary task. Instead, as expected, the inferred velocity is followed with less accuracy since it is considered into the secondary task. At around the 900th sample, the tracking performance degrades as the framework starts switching to the classic VS law. From there on, the behavior converges to the target using the classical VS exponential trend.

Crucially, the tracking of the error norm and the consistent execution of demonstrated velocities as secondary task allow the achievement of the full desired task. In Fig. 3 and 4, the green and red lines are the executions without the secondary

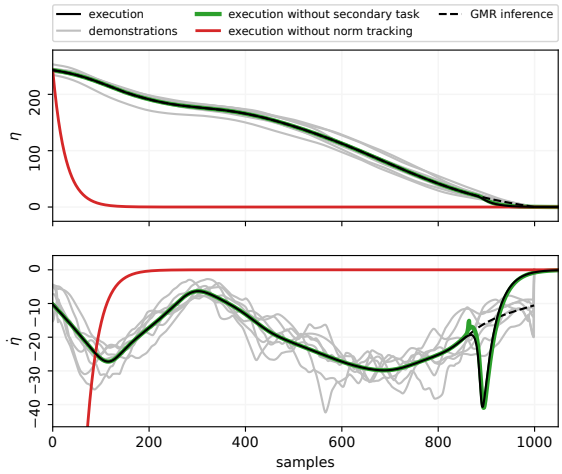

Fig. 4. Simulation analysis of one motion sample: norm of the error (top) and its time derivative (bottom).

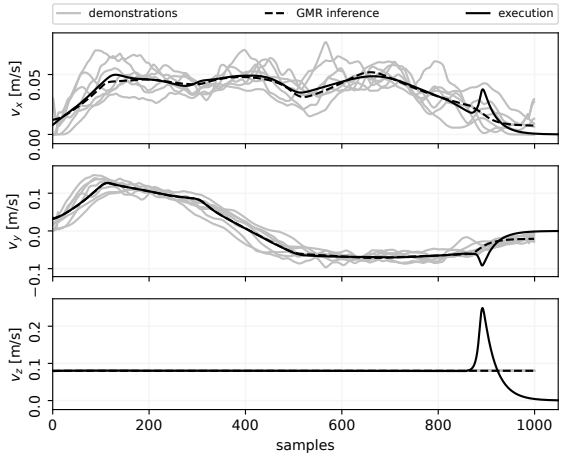

Fig. 5. Simulation analysis of one motion sample: camera velocity.

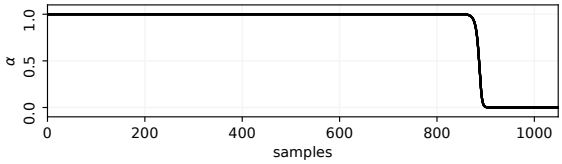

Fig. 6. Simulation analysis of one motion sample: switching variable.

task, i.e., (7) with $\boldsymbol{\sigma} = \mathbf{0}$, and without the norm tracking, i.e., simply (3), respectively. As shown, the combination of these components is essential for the imitation of complex trajectories, i.e. having a non-exponential decay of the η.

Fig. 7 shows the capabilities of the framework to cope with external disturbances. During the execution of the task, between samples 300 and 400, a smooth perturbation of maximum amplitude of 0.1 m/s is applied to the three components of the velocity, as visible in the plot on the right in Fig. 7. Nevertheless, our method well handles the unexpected event converging to the target while imitating the demonstrations, properly recovering from the disturbance.

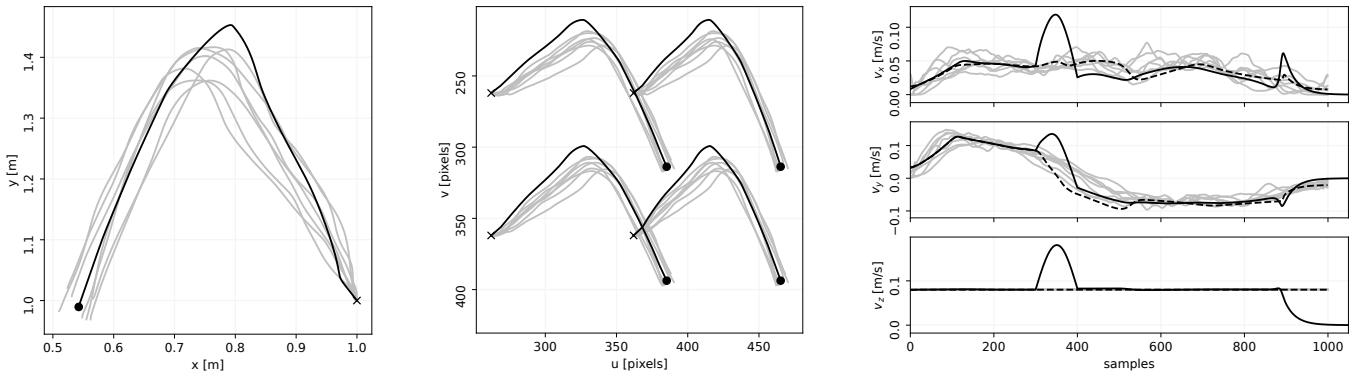

Fig. 7. Simulation analysis on one motion sample with disturb: lateral camera (left) and visual features (center) trajectories, and camera velocity (right).

B. Experiments

We consider the task of inserting a screwdriver in an hole with the 7-axis manipulator by Franka Emika [38], see Fig. 1. The robot is equipped with a camera streaming images of 640×480 pixels. A pattern of 4 points is attached to the insertion location, so that the robot can use the image of these points to drive the inserting motion. A classic VS would bring the screwdriver, hold by the robot gripper, to collide with the environment. Classic DSs using proprioception, instead, would fail if the insertion location changed from the demonstrated one. Our experiments show the benefit of integrating VS and DSs using the proposed strategy.

We collect 3 demonstrations by kinestetically showing the robot right insertion moves. Each demonstration has 200 samples; the online execution is given 300 iterations. A greedy search to obtain enough accuracy suggests to use 11 GMR components for $\dot{\eta}_d$, 5 for $\boldsymbol{\sigma}$. Also for the experiments, we only consider the camera position. The λ gain is set to 0.5; the switching rule has $\eta_0 = 25$, $\eta_1 = 50$ and the other parameters as in Sec. V-A. The computed camera velocity is converted to joint velocities commands using the end-effector Jacobian and the camera extrinsic calibration (to transform the velocity from the camera to the end-effector frame).

Fig. 8 shows the performance of our approach in reaching the convergence while imitating the demonstrations. The accompanying video shows the robot performing this task along with the online camera stream; the comparison with the simple norm regulation; and the execution of our framework with the change of the insertion location.

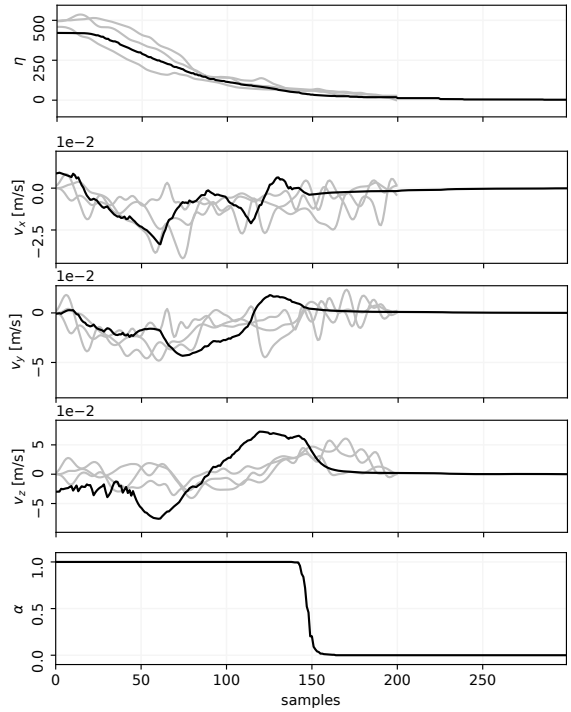

Fig. 8. Experiment with the robotic manipulator: norm of the error (top), camera velocity components (middle) and switching variable (bottom).

VI. Conclusions

We presented a dynamical systems-based imitation learning for visual servoing. Our method is built on the large projection formulation, where the scalar norm of the visual error is considered as feedback. Thus, the null-space of the visual task is large enough to project other tasks. We use this paradigm to establish priorities between the realization of stable visual servoing and the imitation of manual demonstrations. Such a strategy enables complex visual tasks with no explicit implementation of complicated trajectory generators.

The effectiveness of our approach has been shown through a simulation analysis, presenting the performance on a dataset of different motion samples. Experiments with a robotic manipulator have challenged the framework in a real-world scenario. These results have shown the benefits of integrating dynamical systems with visual servoing. On the one side, it allows to easily realize complex visual trajectories without explicit coding. On the other, the visual feedback enables adaptation capabilities to environmental changes.

Future work will extend and validate the generality of our approach. Indeed, the large projection formulation might be also applied to classic kinematic dynamical systems. Furthermore, the learning process could be improved by collecting better demonstrations, e.g. using a planner. More complex experiments, including the orientation control, larger motion, and multiple insertions, will be designed. Finally, we aim at realizing adaptive manipulation tasks, to pave the road to the deployment of our strategy in industrial environments.

REFERENCES

[1] A. Paolillo, G. Abbate, A. Giusti, Š. Trakić, H. Dzafic, A. Fritz, and J. Guzzi, "Towards the integration of a pointing-based human-machine interface in an industrial control system compliant with the IEC 61499 standard," *Procedia CIRP*, vol. 107, pp. 1077–1082, 2022.

[2] D. Lee, "Gesture, posture, facial interfaces," in *Encyclopedia of Robotics*, M. H. Ang, O. Khatib, and B. Siciliano, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2020, pp. 1–10.

[3] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds., 2008, pp. 1371–1394.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[5] S. M. Khansari-Zadeh and A. Billard, "Learning stable non-linear dynamical systems with gaussian mixture models," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 943–957, 2011.

[6] F. Chaumette and S. Hutchinson, "Visual servo control. Part I: Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, 2006.

[7] ——, "Visual servo control. Part II: Advanced approaches," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 109–118, 2007.

[8] A. Paolillo and M. Saveriano, "Learning stable dynamical systems for visual servoing," in *IEEE Int. Conf. on Robotics and Automation*, 2022, pp. 8636–8642.

[9] M. Marey and F. Chaumette, "A new large projection operator for the redundancy framework," in *IEEE Int. Conf. on Robotics and Automation*, 2010, pp. 3727–3732.

[10] A. Ijspeert, J. Nakanishi, P. Pastor, H. Hoffmann, and S. Schaal, "Dynamical Movement Primitives: learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[11] M. Saveriano and D. Lee, "Incremental skill learning of stable dynamical systems," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2018, pp. 6574–6581.

[12] M. Saveriano, "An energy-based approach to ensure the stability of learned dynamical systems," in *IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 4407–4413.

[13] S. M. Khansari-Zadeh and A. Billard, "Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robot. Auton. Syst.*, vol. 62, no. 6, pp. 752–765, 2014.

[14] N. Perrin and P. Schlehuber-Caissier, "Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems," *Systems & Control Letters*, vol. 96, pp. 51–59, 2016.

[15] J. Urain, M. Ginesi, D. Tateo, and J. Peters, "Imitationflow: Learning deep stable stochastic dynamic systems by normalizing flows," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2020, pp. 5231–5237.

[16] K. Kronander and A. Billard, "Passive interaction control with dynamical systems," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 106–113, 2015.

[17] G. Chesi and Y. S. Hung, "Global path-planning for constrained and optimal visual servoing," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1050–1060, 2007.

[18] Y. Mezouar and F. Chaumette, "Path planning for robust image-based control," *IEEE Trans. Robot.*, vol. 18, no. 4, pp. 534–549, 2002.

[19] M. Sauvee, P. Poignet, E. Dombre, and E. Courtial, "Image based visual servoing through nonlinear model predictive control," in *IEEE Conference on Decision and Control*, 2006, pp. 1776–1781.

[20] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *IEEE Trans. Robot.*, vol. 26, no. 5, pp. 933–939, 2010.

[21] D. J. Agravante, G. Claudio, F. Spindler, and F. Chaumette, "Visual servoing in an optimization framework for the whole-body control of humanoid robots," *IEEE Robot. and Autom. Lett.*, vol. 2, no. 2, pp. 608–615, 2017.

[22] E. Mingo Hoffman and A. Paolillo, "Exploiting visual servoing and centroidal momentum for whole-body motion control of humanoid robots in absence of contacts and gravity," in *IEEE Int. Conf. on Robotics and Automation*, 2021, pp. 2979–2985.

[23] A. Paolillo, K. Chappellet, A. Bolotnikova, and A. Kheddar, "Interlinked visual tracking and robotic manipulation of articulated objects," *IEEE Robot. and Autom. Lett.*, vol. 3, no. 4, pp. 2746–2753, 2018.

[24] A. Paolillo, T. S. Lembono, and S. Calinon, "A memory of motion for visual predictive control tasks," in *IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 9014–9020.

[25] E. Pignat and S. Calinon, "Bayesian Gaussian mixture model for robotic policy imitation," *IEEE Robot. and Autom. Lett.*, vol. 4, no. 4, pp. 4452–4458, 2019.

[26] J. W. Kim, C. He, M. Urias, P. Gehlbach, G. D. Hager, I. Iordachita, and M. Kobilarov, "Autonomously navigating a surgical tool inside the eye by learning from demonstration," in *IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 7351–7357.

[27] B. Jia, Z. Pan, Z. Hu, J. Pan, and D. Manocha, "Cloth manipulation using random-forest-based imitation learning," *IEEE Robot. and Autom. Lett.*, vol. 4, no. 2, pp. 2086–2093, 2019.

[28] S. Paradis, M. Hwang, B. Thananjeyan, J. Ichnowski, D. Seita, D. Fer, T. Low, J. E. Gonzalez, and K. Goldberg, "Intermittent visual servoing: Efficiently learning policies robust to instrument changes for high-precision surgical manipulation," in *IEEE Int. Conf. on Robotics and Automation*, 2021.

[29] A. C. Dometios, Y. Zhou, X. S. Papageorgiou, C. S. Tzafestas, and T. Asfour, "Vision-based online adaptation of motion primitives to dynamic surfaces: application to an interactive robotic wiping task," *IEEE Robot. and Autom. Lett.*, vol. 3, no. 3, pp. 1410–1417, 2018.

[30] E. Johns, "Coarse-to-fine imitation learning: Robot manipulation from a single demonstration," in *IEEE Int. Conf. on Robotics and Automation*, 2021, pp. 4613–4619.

[31] A. Phung, J. Malzahn, F. Hoffmann, and T. Bertram, "Tool centered learning from demonstration for robotic arms with visual feedback," in *IEEE Int. Conf. on Robotics and Biomimetics*, 2012, pp. 1117–1122.

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

[33] K. Neumann and J. J. Steil, "Learning robot motions with stable dynamical systems under diffeomorphic transformations," *Robot. Auton. Syst.*, vol. 70, pp. 1–15, 2015.

[34] É. Marchand and F. Chaumette, "Feature tracking for visual servoing purposes," *Robot. Auton. Syst.*, vol. 52, no. 1, pp. 53–70, 2005.

[35] A. Paolillo, M. Nava, D. Piga, and A. Giusti, "Visual servoing with geometrically interpretable neural perception," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2022, pp. 5300–5306.

[36] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[37] R. Spica, P. Robuffo Giordano, and F. Chaumette, "Coupling active depth estimation and visual servoing via a large projection operator," *The International Journal of Robotics Research*, vol. 36, no. 11, pp. 1177–1194, 2017.

[38] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jaehne, L. Hausperger, and S. Haddadin, "The Franka Emika robot: A reference platform for robotics research and education," *IEEE Robot. Autom. Mag.*, 2022.