Learning dynamic graffiti strokes with a compliant robot

Daniel Berio[1], Sylvain Calinon[2] and Frederic Fol Leymarie[1]

***Abstract*— We present an approach to generate rapid and fluid drawing movements on a compliant Baxter robot, by taking advantage of the kinematic redundancy and torque control capabilities of the robot. We concentrate on the task of reproducing graffiti-stylised letter-forms with a marker. For this purpose, we exploit a compact *lognormal*-stroke based representation of movement to generate natural drawing trajectories. An Expectation-Maximisation (EM) algorithm is used to iteratively improve tracking performance with low gain feedback control. The resulting system captures the aesthetic and dynamic features of the style under investigation and permits its reproduction with a compliant controller that is safe for users surrounding the robot.**

I. INTRODUCTION

Drawing and handwriting are challenging tasks for robots. Even for humans, a long time is required from childhood to learn how to manipulate a pen with fluidity to produce smooth and elegant sequences of letters such as in a signature or handwritten notes. Thus far, most approaches employed in robotics to mimic this human trait have relied on precise and stiff position-controlled robots, by using either manipulators [1], [2], [3] or Cartesian robots [4], [5]. The use of such platforms guarantees that the pen will be in contact with the paper but requires fine calibration of the drawing pad and only superficially emulates the essence of human drawing skills.

We present here the novel use of a robot with passive and active compliance to enable the transfer of more natural drawing skills while exploiting contact with the drawing pad. Furthermore, we focus on the robotic reproduction of *graffiti tags* (Fig. 1). With the term *graffiti*, we refer to the artistic movement that first emerged in the late 1960's on the surfaces of the New York City public transport system, and which revolves around different forms of stylisation and abstraction applied to letters of an alphabet [6]. In its most elementary and fundamental instantiation, graffiti art takes the form of a rapidly executed and highly stylised signature, which is commonly referred to as a *tag*. Tags are a *calligraphic* form of writing: the letters composing a tag are not important in their semantic meaning but rather are meant to impress the viewer with their figurative form and style. We investigate the generation of graffiti tags as it provides a useful testbed for robot drawing applications. Tag genesis permits to explore different aspects of human motor skills, including fast and dynamic movements, with an efficient combination of open and closed loop behaviours, strong variations in the letter forms and a personalised style that is potentially recognisable across different sets of letters [7], [6].

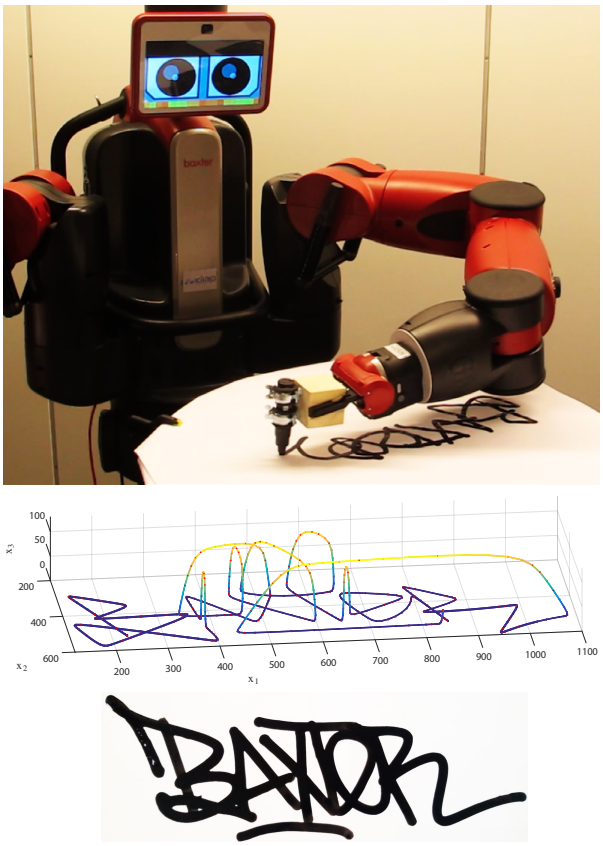


Fig. 1: *Top*: Baxter drawing its name as a graffiti tag. *Middle*: Corresponding trajectory with the 3D motion of the marker, including pen-up trajectories. *Bottom*: An example of a similar tag produced by a human artist.

This work stems from our general interest in studying the perceptual processes and movement dynamics that underlie the production of various forms of art [8], [9], [10]. We aim at developing computational models that will enable the transfer of complex and personal artistic skills to robotic platforms. Our contribution is a novel application in robotics of a lognormal stroke based representation of movement [11] which so far has only been adopted in the handwriting analysis and synthesis domains. We exploit the compactness of this representation to (i) generatively capture the variations in trace and dynamics of complex tags and (ii) iteratively improve tracking performance with a compliant robotic arm. Our method achieves the desired types of movement dynamics and traces within a few iterations. Furthermore, the exploitation of active/passive compliance for generating fast

[1]Goldsmiths College, University of London. Department of Computing. (d.berio,ffl)@gold.ac.uk
[2]Idiap Research Institute, Martigny, Switzerland. sylvain.calinon@idiap.ch

writing and drawing motions in a robot is new.

The remaining of the paper is organised as follows: §II gives a brief background including related work; §III presents our approach for generating trajectories for the robot; §IV describes the controller used to drive the robot's motions; §V details the iterative correction scheme used to achieve a desired trajectory tracking performance.

II. BACKGROUND

A. Drawing robots

Drawing machines have inspired several generations of engineers and artists. One of the first such machines was designed by Jacquet Droz (1721-90) who created a mechanical automaton that could write sentences in cursive script. In the 20th Century, the *Algorists* were a group of pioneering artists and computer scientists who employed Cartesian robots (pen-plotters) to algorithmically generate artwork (mainly drawings) [12], [4]. Since the 1970's, Harold Cohen developed AARON, a computerised system that generates original compositional artworks derived from Cohen's own stylistic ideas. AARON has occasionally been embodied in the form of a large scale robotic platform and has performed in front of human audiences [13].

More recently, robotic manipulators are used to produce artworks ranging from portrait sketching applications driven by computer vision [14], [9] to systems that use optimisation methods to reproduce an image with acrylic paint on canvas at different levels of abstraction [1], [2]. Also in recent years, a number of studies have been carried out towards the reproduction of East Asian calligraphy strokes with a variety of robotic devices, ranging from articulated arms [15], [16], [3], [17], to specifically built Cartesian drawing systems [5], [18], [19], as well as Unmanned Aerial Vehicles (*aka* drones) [20]. While most of these works focus on a careful reproduction of the brush pressure and footprint, we put an emphasis on the study of the relationship between the movement employed in trajectory generation and the resulting trace. Mueller *et al.* use visual feedback to reproduce single calligraphic strokes with a robot by iteratively adjusting brush pressure and the control points of a spline [3]. We employ a similar iterative scheme for trajectory correction; however, we focus on the ballistic character of more complex writing trajectories and choose to use proprioceptive feedback for the task at hand. Shinoda *et al.* also propose an interactive system to generate synthetic Japanese cursive calligraphy trajectories using B-Splines together with an optimal control scheme [21] resulting in smooth motion trajectories [15]. Potkonjak studies biomechanical properties of handwriting in humans, such as fatigue and distribution of forces across joints, and develops mathematical models of these principles for the generation of handwriting motions in a robotic arm with redundant degrees of freedom [22]. De Santis *et al.* explore the reproduction of cursive handwriting movements with a simulated 7 DoF robotic arm [23]. The authors propose a weighted inverse kinematics (IK) solution, where the weights constrain the movements to approximate a human arm while writing. We use a similar IK solution, but with a different weighting scheme which is chosen based on the mechanical properties of the manipulator to improve tracking accuracy and stability during the drawing task.

B. Generative art through movement simulation

We have previously proposed that in order to interactively or procedurally generate convincing tags, it is essential to simulate the movement underlying its production [10]. The distinctive style that typically denotes tags is conferred by the execution of skilled and rapid writing motions. With experience and extensive practice, the gestures involved in the production of a tag become second nature; this results in rapid open loop movements which is reflected in the resulting traces and determines their aesthetic quality.

Consequently, our approach for the computer generation of tags and their reproduction with a robot is informed by research in movement science as well as work conducted within the field of *graphonomics* [24], which is primarily focused on the computational analysis and synthesis of handwriting traces and motions. It has generally been observed that the velocity profile of rapid and straight reaching motions can be described by a "bell shaped" velocity profile [25], [26]. The velocity profile can be variably asymmetric depending on the movement speed [27] and the best fit to empirical data is given by 3-parameter lognormals [28], [29]. Many studies propose that smooth motions can be described as the composition of a number of discrete "ballistic" movement units [25], [30], [31], which in turn can be modeled with the characteristic bell-shaped speed profile.

III. TRAJECTORY GENERATION

For the task of generating tag trajectories, we rely on the Sigma Lognormal ($\Sigma\Lambda$) model developed by Plamondon *et al.* [11], which describes complex hand trajectories via the linear combination in time of a number of stroke primitives. The speed profile of each i^{th} stroke is defined with a 3-parameter lognormal

$$\Lambda_i(t) = \frac{1}{\sigma\sqrt{2\pi}(t-t_{0i})} \exp\left(-\frac{(\ln(t-t_{0i})-\mu_i)^2}{2\sigma_i^2}\right), \quad (1)$$

where t_0 is the time of occurrence of the input command for the stroke and (μ, σ) determine the overall shape of the lognormal. μ is the time-delay in logarithmic time scale (referred to as logtime delay) and indicates the rapidity of the system to react to the input command. σ is the response-time in a logarithmic time scale (*aka* logresponse time) and determines the spread and asymmetry of the lognormal. For an in-depth discussion of the effects and biological interpretation of the lognormal parameters we refer the reader to the work of Plamondon *et al.* [32].

The spatial evolution of a trajectory is defined in the form of an *action plan* (or "motor program") which is described by M *virtual targets* $\{\boldsymbol{v}_i\}_{i=1}^{M}$ each joined by $M-1$ strokes. With the assumption that curved handwriting movements are done by rotating around a pivot (*e.g.* the wrist), the curvature evolution of a stroke is described with a circular arc. The smoothness of trajectories can be defined by adjusting the

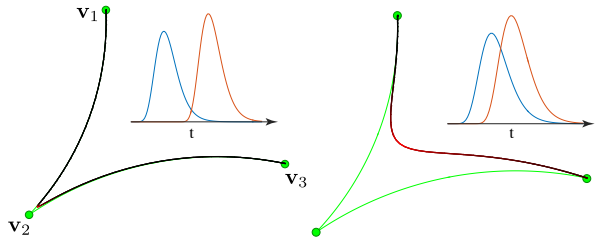


Fig. 2: Effect of varying the third lognormal parameter t_0. In green the action plan and the corresponding virtual targets. The red gradient indicates the part of the trajectory in which the influence of two consecutive strokes overlaps.

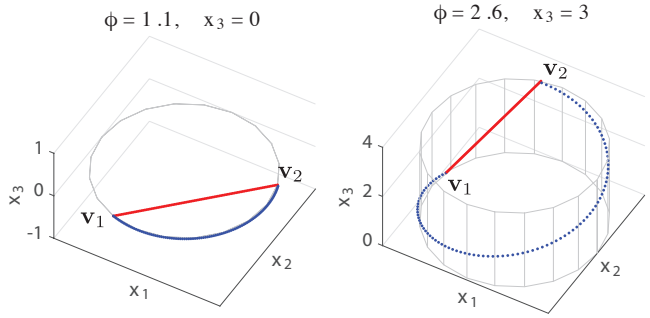


Fig. 3: Different types of strokes and the corresponding cylinder.

time overlap of strokes (t_0 parameter), where a greater overlap results in a smoother trajectory (Fig. 2).

While in its original formulation the $\Sigma\Lambda$ model describes a trajectory with series of planar orientation/magnitude pairs, we use a reparametrisation of the model in which we explicitly define the virtual target positions. This is advantageous both for the interactive manipulation of trajectories [10], as well as for the iterative correction scheme that will be later described. In order to allow for smooth up and down pen movements, we add a third dimension to the virtual target positions. This effectively results in a helical stroke primitive that evolves along the surface of a cylinder, the height of which is defined by the x_3 offset between consecutive virtual targets (Fig. 3). The curvilinear evolution in time for a stroke with curvature θ_i is computed via

$$\phi_i(t) = \theta_i + \theta_i \left[1 + \text{erf}\left(\frac{\ln(t - t_{0i}) - \mu_i}{\sigma_i\sqrt{2}}\right)\right]. \quad (2)$$

The $\Sigma\Lambda$ equation parameterised by the virtual target positions becomes

$$\boldsymbol{p}(t) = \boldsymbol{v}_1 + \int_0^t d\tau \Lambda_j(\tau) \sum_{i=1}^{M-1} \boldsymbol{\Phi}_i(\tau)\,(\boldsymbol{v}_{i+1} - \boldsymbol{v}_i), \quad (3)$$

$$\text{with} \quad \boldsymbol{\Phi}_i(t) = \begin{bmatrix} s(\theta_i)\cos\phi_i(t) & -s(\theta_i)\sin\phi_i(t) & 0 \\ s(\theta_i)\sin\phi_i(t) & -s(\theta_i)\cos\phi_i(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

$$\text{and} \quad s(\theta_i) = \begin{cases} \frac{2\theta_i}{2\sin\theta_i} & \text{if } |\sin\theta_i| > 0, \\ 1 & \text{otherwise,} \end{cases} \quad (5)$$

which scales the extent of the stroke based on the arc length of the helical stroke. With this parametrisation, we can easily specify trajectories with an intuitive *point and click procedure* [10], similarly to the one used in CAD software to define splines, except that it also generates a smooth velocity profile that is a desirable property for natural movement generation.

IV. ROBOT CONTROL

Given the trajectory $\boldsymbol{p}(t)$ generated by the $\Sigma\Lambda$ model, we first compute an affine transformation $\boldsymbol{A}$ that maps the trajectory to the drawing plane with respect to the reference frame of the robot. The plane and the extents of the robot's drawing workspace are computed by demonstrating a series of drawing movements by accompanying the manipulator along the drawing surface through kinesthetic teaching [33]. The trajectory to be tracked by the robot is then computed with $\hat{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{p}(t)$. We then proceed with computing the corresponding joint velocities that will be used to control the robot's drawing motions.

Given the *position* and *orientation* Jacobians for the Cartesian $\dot{\boldsymbol{x}}$ and angular $\dot{\boldsymbol{\psi}}$ components of the end-effector velocity

$$\boldsymbol{J}_x = \left(\frac{\delta x_i}{\delta q_j}\right)_{i,j} \in \mathbb{R}^{3\times 7} \quad \text{and} \quad \boldsymbol{J}_\psi = \left(\frac{\delta \psi_i}{\delta q_j}\right)_{i,j} \in \mathbb{R}^{3\times 7},$$

we compute a least norm IK solution with

$$\dot{\hat{\boldsymbol{q}}} = \overbrace{\boldsymbol{J}_x^* \dot{\hat{\boldsymbol{x}}}}^{\dot{\boldsymbol{q}}_x} + \overbrace{\boldsymbol{N}(\boldsymbol{J}_x)\, k_\psi \boldsymbol{J}_\psi^* \boldsymbol{e}_\psi}^{\dot{\boldsymbol{q}}_\psi} + \overbrace{\boldsymbol{N}(\boldsymbol{J}_x)\, k_d \nabla H}^{\dot{\boldsymbol{q}}_d}. \quad (6)$$

The controller exploits the redundancy of the 7 DoF arm to enforce three tasks: one primary and two competing secondary tasks characterized by gains k_d and k_ψ.

a) The principal task $\dot{\boldsymbol{q}}_x$: It tracks the desired Cartesian velocity $\dot{\boldsymbol{x}}$ using a weighted pseudo-inverse of the position Jacobian computed with

$$\boldsymbol{J}_x^* = \boldsymbol{W}^{-1}\boldsymbol{J}_x^\top\left(\boldsymbol{J}_x\boldsymbol{W}^{-1}\boldsymbol{J}_x^\top + \lambda\boldsymbol{I}\right)^{-1}, \quad (7)$$

where λ is a regularisation term that avoids singularities in the vicinity of joint limits and $\boldsymbol{W}$ is a positive definite weight matrix (here, diagonal), where high weight in the diagonal penalises the movement of the corresponding joint. We empirically choose to penalise the movement of the 2 upper joints of the manipulator, as they are driven by more powerful and less precise motors which tend to produce jerkier motions.

b) The secondary task $\dot{\boldsymbol{q}}_\psi$: It uses the *gradient projection* operator [34] to enforce a soft orientation constraint which keeps the pen approximately perpendicular to the drawing surface as a secondary task described by $\boldsymbol{J}_\psi^*\boldsymbol{e}_\psi$, where $\boldsymbol{e}_\psi$ is the error between the desired and current end-effector orientation and $\boldsymbol{J}_\psi^*$ is the weighted pseudoinverse of the orientation Jacobian which is computed identically to (7). This is done by projecting the joint velocity given by $\boldsymbol{J}_\psi^*\boldsymbol{e}_\psi$ onto the null space of the position Jacobian

$$\boldsymbol{N}(\boldsymbol{J}_x) = \boldsymbol{I} - \boldsymbol{J}_x^\top\boldsymbol{J}_x^*. \quad (8)$$

Tracking the pen orientation as a secondary task permits to extend the workspace of the robot to a larger area. Note here that we use a marker with a round nib and that the rotation around the x_3 axis in the tool frame does not need to be controlled.

c) The secondary task $\dot{\boldsymbol{q}}_d$: It also uses *gradient projection* to map a joint velocity onto the null space of the position Jacobian, which results in a *self-motion* of the limb which does not affect the end-effector position. This permits us to enforce, as a secondary task, a joint configuration $\boldsymbol{q}_d$ approximately parallel to the drawing pad, which is achieved by computing the gradient of the quadratic cost function

$$H = \frac{1}{7}\sum_{i=1}^{7}\left(\frac{q_i - q_{di}}{q_i^{\max} - q_i^{\min}}\right)^2, \tag{9}$$

where q_i is the i^{th} joint angle and (q_i^{min}, q_i^{max}) are the joint limits.

Given the desired joint positions $\hat{\boldsymbol{q}}$ and velocities $\hat{\dot{\boldsymbol{q}}}$ we finally use a proportional-derivative controller to track the trajectory with[1]

$$\boldsymbol{\tau} = \boldsymbol{K}_p(\hat{\boldsymbol{q}} - \boldsymbol{q}) + \boldsymbol{K}_v(\hat{\dot{\boldsymbol{q}}} - \dot{\boldsymbol{q}}) + \boldsymbol{J}_x^{\top}\boldsymbol{f} + \boldsymbol{\tau}_g, \tag{10}$$

where $\boldsymbol{K}_p$ and $\boldsymbol{K}_v$ are proportional and derivative positive-definite gain matrices (diagonal in our case), $\boldsymbol{f}$ is a constant force perpendicular to the drawing surface that is used as a small command offset to adjust the pressure of the pen, and $\boldsymbol{\tau}_g$ is a gravity compensation term computed based on a dynamic model of the robot.

V. ITERATIVE CORRECTION SCHEME

While a suitable tracking performance could be achieved by setting high controller gains and a critically damped system, we instead choose to employ low feedback gains so that the system can: (1) be compliant and behave more similarly to the human arm; (2) be safer in a human-robot collaboration scenario; (3) be more energy efficient and safer for the robotic hardware; (4) exploit the contact with the drawing surface to stabilise the movements. In return, low controller gains will degrade the tracking performance due to the multiple nonlinearities caused by imprecisions in the model and unmodeled external factors such as friction and system delays [35]. For recurring movements, several generic low-level approaches exist, such as iterative learning control (ILC), which compensate for such errors while keeping the compliant capability of the system [36]. We propose here a higher-level task-specific approach that exploits the ballistic nature of the task under investigation. It directly relies on the compact stroke-based representation used to generate the trajectories to iteratively compensate for tracking errors.

Each stroke can be considered as an aiming motion towards a virtual target. Intuitively we can shift the virtual target positions in order to compensate for errors in the direction and extent of strokes. While virtual targets do not usually lie along the trace of the trajectory (Fig. 4), we can identify a series of *key-points* $\{\hat{z}_i\}_{i=1}^{M}$ along the trajectory generated by the model and corresponding points $\{z_i\}_{i=1}^{M}$ along the trajectory reproduced by the robot, which can then be used to iteratively compute corrective terms.

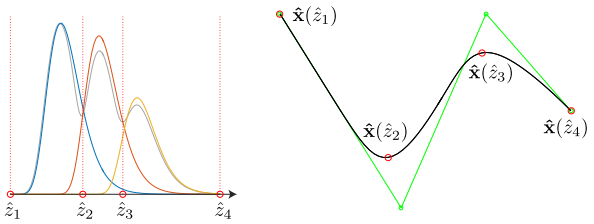


Fig. 4: Key-point estimation by finding intersection between lognormal components (*left*) and the corresponding points along the trajectory (*right*).

In our representation, key-points are given by the starting and ending time of the trajectory together with the time occurrences at which the influence of a virtual target exceeds that of the previous target. These points will approximately correspond to maxima of curvature and minima in velocity along the trajectory. The key-points for the desired trajectory are easily identified by numerically finding the intersection points between each pair of consecutive lognormals. However, due to the nonlinearities induced by the low feedback gains and contact with the environment, the key-points in the reproduced trajectory can be misaligned with respect to the original key-points and we will have to compute an estimate given the data recorded through the robot's encoders (Fig. 5).

Given a sufficiently accurate estimate of the key-points in the reproduced trajectory, we improve the tracking performance by iteratively offsetting each virtual target at each reproduction step with

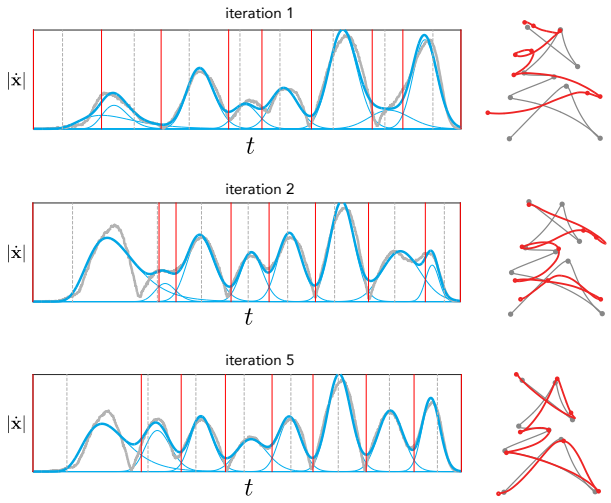
$$\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + \alpha\Big(\hat{\boldsymbol{x}}(\hat{z}_i) - \boldsymbol{A}^{-1}\boldsymbol{x}(z_i)\Big), \tag{11}$$

where α is a *learning rate* parameter that linearly scales the correction applied at each iteration (which we empirically set to 1 in the first iteration and 0.5 in the successive ones), and $\boldsymbol{A}$ is the transformation matrix that maps the generated trajectory to the drawing plane of the robot (Fig. 6).

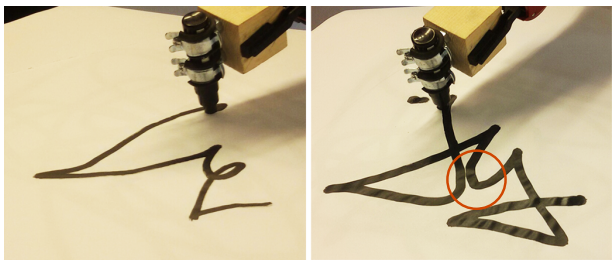
A. Keypoints Estimation

An accurate identification of key-points in the reproduced trajectory is crucial for a successful application of the iterative correction method. If we treat the speed profile reproduced by the robot as a probability density function, we can then use statistical methods to estimate the positions of key-points. We use a version of Expectation-Maximisation (EM) [37] in which each datapoint is associated with a weight influencing its importance in the process. This allows us to model speed profiles as mixture of parametric distributions with EM. Consequently, we can use the intersection between mixture components to identify key-points in a process that is identical to the one we use to find key-points in the generated

[1]Our first tests employed computed torque control but the inertial parameters of the robot model were not accurate enough to obtain a stable controller.



(a)



(b)

Fig. 5: *(a)* Three steps of the iterative correction scheme. Left, the lognormal mixture computed from the reproduced speed signal with the key-points marked in red; right, the corresponding trajectories and key-points (desired in gray, reproduced in red). *(b)* Corresponding reproductions achieved by the robot. The red circle indicates an imprecise motion part due to configuration dependent errors.

trajectory, with $K = M - 1$ the desired number of mixture components (M is the number of virtual targets).

We have tested different methods for estimating key-points in the reproduced trajectory, including various types of mixtures (Laplace, Gaussian, Lognormal), Dynamic Time Warping and estimating minima in the reproduction speed signal; estimating a mixture of lognormals resulted in more consistent results across trials and in a faster convergence of the iterative correction algorithm. While in practice it would be desirable to estimate a mixture of 3-parameter lognormals, this is a known difficult problem [38], [39] due to the risk of EM converging to local optima corresponding to an overall good fit but that only poorly reflect the activation signal property we are looking for. In practice, it would be interesting to find offsets corresponding to activation signals triggering the lognormal profiles. The problem is that there are other local solutions very close to this optimum, and it is then likely that EM can get trapped in these local solutions instead of the one we seek. These local optima also provide a good fit in terms of log-likelihood, but some

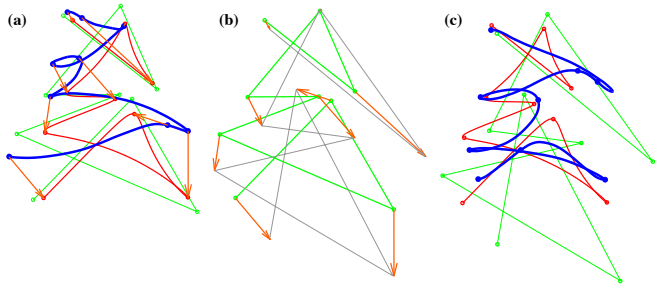


Fig. 6: One iteration of the corrective scheme, with $\alpha = 1$. *(a)* The offsets (orange vectors) between key-points in the reproduced trajectory (blue) and the desired (red) are computed. *(b)* The same offsets are applied to the virtual targets, resulting in a new action plan *(c)* which is then reproduced for the next iteration.

of the local optima only poorly exploit the activation signal property (t_0) of the distributions. In the proposed use-case, we only identify the intersection points between mixture components, and it is thus sufficient to fit 2 parameter lognormals (*i.e.* (1) with $t_0 = 0$), which can be implemented in a straightforward manner by normalising the duration to the range $[0, 1]$, transforming the input data into the log-space and then performing EM with a Gaussian Mixture Model. We employed the procedure described in [40] that consists of starting from random initialisations followed by only a few EM iterations to select the model on which we will apply the complete EM process until convergence.

B. Results and discussion

We tested our method on different trajectories in both horizontal and vertical planes, including complex movements with up and down pen motions. The iterative correction method converged to an overall satisfactory result in a few iterations: typically 4 to 10 depending on trajectory complexity (Fig. 5). Due to the nonlinearities and compliance of the robot, the first reproduction attempt only barely matched the desired result, but after a single correction step the trajectory was already recognisable. Furthermore the trajectory generation method resulted in natural and rapid motions (see accompanying video [41]). This is reflected in the high quality of the resulting traces that we evaluated qualitatively based on the feedback of expert graffiti artists.

Future work will investigate potential ways of evaluating quantitatively the aesthetics quality of the reproduced graffiti tags. The problem is very challenging and involves relating the properties of the static result with the underlying dynamics of the movements that was used to produce them. Defining such metric will also require to take into account subtle and specialised elements and will likely benefit from the integration of proprioceptive feedback with vision. We also plan in future work to compare our method with more traditional corrective approaches such as ILC.

Note that despite the good results in movement and trajectory reproduction, some configuration dependent errors

still persist, see red circle in Fig. 5. In future work, we plan to investigate potential ways of correcting these types of errors by adjusting the $\Sigma\Lambda$ parameters in addition to the virtual target positions.

VI. CONCLUSION

We have presented a simple yet effective method for generating natural drawing and calligraphic motions, and reproducing them on a compliant robot. Such scheme is useful in robotic drawing applications that involve the reproduction of human-like calligraphic styles (or more generally stroke-based image styles), in which it is necessary to generate the ballistic nature of movement units rather than to achieve industrial grade tracking accuracy. In future work we plan to further exploit the model parameters to correct a wider range of local errors that can currently occur. The proposed method is currently trajectory and configuration specific; we plan to investigate if the $\Sigma\Lambda$ movement representation can be exploited to generalize the approach to corrective terms across different trajectories and configurations. Finally we plan to complement proprioceptive feedback with visual feedback in order to explore more complex artist-robot collaborative scenarios.

ACKNOWLEDGMENT

We thank Prof. Réjean Plamondon (Polytechnique Montréal) for the useful comments and feedback on an earlier version of the manuscript. This work has been partly supported by UK's EPSRC Centre for Doctoral Training in Intelligent Games and Game Intelligence (IGGI; grant EP/L015846/1).

REFERENCES

[1] C. Aguilar and H. Lipson, "A robotic system for interpreting images into painted artwork," in *Int'l Conf. on Generative Art*, 2008.

[2] O. Deussen, T. Lindemeier, S. Pirk, and M. Tautzenberger, "Feedback-guided stroke placement for a painting machine," in *Proc. of 8th Symp. on Comp. Aesthetics in Graphics, Visualization, and Imaging*, 2012, pp. 25–33.

[3] S. Mueller, N. Huebel, M. Waibel, and R. D'Andrea, "Robotic calligraphy — learning how to write single strokes of Chinese and Japanese characters," in *IEEE Proc. of IROS*, 2013, pp. 1734–9.

[4] F. Dietrich, "Visual intelligence: The first decade of computer art (1965-75)," *Leonardo*, vol. 19, no. 2, pp. 159–69, 1986.

[5] K. W. Kwok, K. W. Lo, S. M. Wong, and Y. Yam, "Evolutionary replication of calligraphic characters by a robot drawing platform," in *IEEE Int'l Conf. Automation Sci. & Engineering*, 2006, pp. 466–71.

[6] J. Kimvall, *The G-word*. Stockholm: Dokument, 2014.

[7] L. Gottlieb, *Graffiti art styles: A classification system and theoretical analysis*. McFarland, 2008.

[8] F. F. Leymarie, "Aesthetic computing and shape," in *Aesthetic Computing*, P. Fishwick, Ed. MIT Press, April 2006, ch. 14.

[9] P. Tresset and F. Fol Leymarie, "Portrait drawing by Paul the robot," *Computers & Graphics*, vol. 37, no. 5, pp. 348–63, 2013.

[10] D. Berio and F. F. Leymarie, "Computational Models for the Analysis and Synthesis of Graffiti Tag Strokes," in *Computational Aesthetics*, P. Rosin, Ed. Eurographics Association, 2015, pp. 35–47.

[11] R. Plamondon, C. O'Reilly, J. Galbally, A. Almaksour, and É. Anquetil, "Recent developments in the study of rapid human movements with the kinematic theory," *Pattern Recognition Letters*, vol. 35, pp. 225–35, 2014.

[12] R. Leavitt, *Artist and computer*. Harmony Books, 1976.

[13] P. McCorduck, *AARON's Code — Meta-Art, Artificial Intelligence, and the Work of Harold Cohen*. Freeman, 1991.

[14] S. Calinon, J. Epiney, and A. Billard, "A humanoid robot drawing human portraits," in *5th IEEE-RAS Int'l Conf. on Humanoid Robots*, 2005, pp. 161–6.

[15] H. Shinoda, H. Fujioka, and H. Kano, "Generation of cursive characters using minimum jerk model," in *IEEE Proc. SICE*, vol. 1, 2003, pp. 730–3.

[16] Y. Sun and Y. Xu, "A calligraphy robot — Callibot," in *IEEE Proc. ROBIO*, 2013, pp. 185–90.

[17] J. Li, W. Sun, M. Zhou, and X. Dai, "Teaching a calligraphy robot via a touch screen," in *IEEE Proc. CASE*, 2014, pp. 221–6.

[18] J. H. Lam and Y. Yam, "Stroke trajectory generation experiment for a robotic Chinese calligrapher using a geometric brush footprint model," in *IEEE Proc. of IROS*, 2009, pp. 2315–20.

[19] Y. Man, C. Bian, H. Zhao, C. Xu, and S. Ren, "A kind of calligraphy robot," in *IEEE Proc. of ICIS*, 2010, pp. 635–8.

[20] S. K. Phang, S. Lai, F. Wang, M. Lan, and B. M. Chen, "Systems design & implementation with jerk-optimized trajectory generation for UAV calligraphy," *Mechatronics*, vol. 30, pp. 65–75, 2015.

[21] T. Flash and N. Hogan, "The coordination of arm movements," *Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–703, 1985.

[22] V. Potkonjak, "Robotic handwriting," *International Journal of Humanoid Robotics*, vol. 2, no. 01, pp. 105–24, 2005.

[23] A. De Santis, V. Caggiano, B. Siciliano, L. Villani, and G. Boccignone, "Anthropic inverse kinematics of robot manipulators in handwriting tasks," in *12th Conference of the International Graphonomics Society*, 2005.

[24] H. S. Kao, R. Hoosain, and G. Van Galen, *Graphonomics: Contemporary research in handwriting*. Elsevier, 1986.

[25] P. Morasso, "Spatial control of arm movements," *Experimental Brain Research*, vol. 42, no. 2, pp. 223–7, 1981.

[26] T. Flash and B. Hochner, "Motor primitives in vertebrates and invertebrates," *Current opinion in neurobiology*, vol. 15, no. 6, pp. 660–6, 2005.

[27] H. Nagasaki, "Asymmetric velocity and acceleration profiles of human arm movements," *Experimental Brain Research*, vol. 74, no. 2, pp. 319–26, 1989.

[28] R. Plamondon, A. M. Alimi, P. Yergeau, and F. Leclerc, "Modelling velocity profiles of rapid movements: A comparative study," *Biological cybernetics*, vol. 69, no. 2, pp. 119–28, 1993.

[29] B. Rohrer and N. Hogan, "Avoiding spurious submovement decompositions II," *Biological cybernetics*, vol. 94, no. 5, pp. 409–14, 2006.

[30] P. Morasso, "Understanding cursive script as a trajectory formation paradigm," *Graphonomics*, vol. 37, pp. 137–67, 1986.

[31] H.-L. Teulings and L. R. Schomaker, "Invariant properties between stroke features in handwriting," *Acta psychologica*, vol. 82, no. 1, pp. 69–88, 1993.

[32] R. Plamondon, C. Feng, and A. Woch, "A kinematic theory of rapid human movement. Part IV," *Biological Cybernetics*, vol. 89, no. 2, pp. 126–38, 2003.

[33] P. Kormushev, D. N. Nenchev, S. Calinon, and D. G. Caldwell, "Upper-body kinesthetic teaching of a free-standing humanoid robot," in *IEEE Proc. of ICRA*, Shanghai, China, 2011, pp. 3970–5.

[34] A. Liegeois, "Automatic supervisory control of the config. & behavior of multibody mechanisms," *IEEE Trans. SMC*, vol. 7, no. 12, pp. 868–71, 1977.

[35] D. Nguyen-Tuong and J. Peters, "Learning robot dynamics for computed torque control using local Gaussian processes regression," in *IEEE Proc. LAB-RS*, 2008, pp. 59–64.

[36] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Systems*, vol. 26, no. 3, pp. 96–114, 2006.

[37] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Statistical Society B*, vol. 39, no. 1, pp. 1–38, 1977.

[38] R. J. Aristizabal, "Estimating the parameters of the three-parameter lognormal distribution," FIU theses 575, Florida International University, 2012.

[39] P. Basak, I. Basak, and N. Balakrishnan, "Estimation for the three-parameter lognormal distribution based on progressively censored data," *Computational Statistics & Data Analysis*, vol. 53, no. 10, pp. 3580–92, 2009.

[40] C. Biernacki, G. Celeux, and G. Govaert, "Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models," *Computational Statistics & Data Analysis*, vol. 41, no. 34, pp. 561–75, 2003.

[41] D. Berio, S. Calinon, and F. Fol Leymarie, "Learning dynamic graffiti strokes with a compliant robot, accompanying video," https://youtu.be/yl5u9NjB8YU, 2016.