

Supplementary Information for
Model-free tracking control of complex dynamical trajectories with machine learning

Zheng-Meng Zhai, Mohammadamin Moradi, Ling-Wei Kong, Bryan Glaz, Mulugeta Haile, and
Ying-Cheng Lai

Corresponding author: Ying-Cheng Lai (Ying-Cheng.Lai@asu.edu)

CONTENTS

Supplementary Note 1: Reservoir computing	2
Supplementary Note 2: Generation of 15 types of reference trajectories	3
Chaotic Lorenz reference trajectory	3
Circular reference trajectory	4
Chaotic Mackey-Glass reference trajectory	4
Periodic figure-8 reference trajectory	4
Chaotic Rössler reference trajectory	5
Fermat's spiral reference trajectory	5
Heart-shaped reference trajectory	5
Astroid reference trajectory	5
Two types of chaotic trajectories from dynamical systems with a polynomial velocity field	6
Epitrochoid reference trajectory	6
Chaotic Chua-circuit reference trajectory	7
Talbot reference trajectory	7
Lissajous reference trajectory	7
Supplementary Note 3: Bridge to reference trajectory	8
Supplementary Note 4: Alternating and continuous tracking of a variety of reference trajectories	8
Tracking Control	8
Robustness against uncertainties	10
Effect of varying training parameters	11
Supplementary Note 5: Pertinent issues of tracking control	13
Safe region of initial conditions for control success	13
Tracking speed tolerance	14
Robustness against variations in training parameters	15
Supplementary Note 6: Tracking control with feed-forward neural networks	17
References	20

SUPPLEMENTARY NOTE 1: RESERVOIR COMPUTING

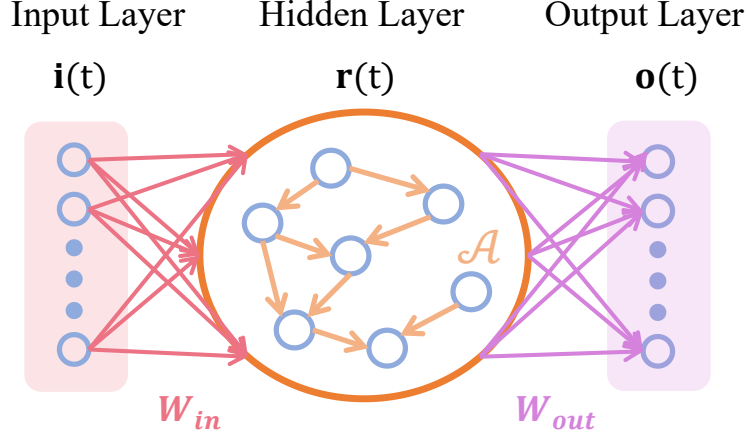


FIG. S1. Structure of reservoir computing.

The key requirement for selecting a machine-learning scheme for our tracking-control problem is to generate a control signal from a desired trajectory. When the trajectory is chaotic, the control signal can be quite complex. Reservoir computing [1–3], a type of recurrent neural network, is a suitable machine-learning approach for generating complex desired signals as output from complex input signals. Hence, it is an appropriate choice for our problem. A reservoir computer is composed of three layers: an input layer, a hidden recurrent layer, and an output layer, as shown in Fig. S1. The distinguishing feature of reservoir computing (RC) is that only the readout weights (W_{out}) are trained using a simple learning algorithm, e.g., linear regression, while the input weights (W_{in}) and recurrent connection weights within the reservoir ($\mathcal{A}$) are left untrained. This approach enables a significant reduction in computational costs for learning compared to conventional RNNs, which is the major advantage of RC. Since no backpropagation is required, the difficulty of exploding/vanishing gradients associated with the training of recurrent neural networks does not occur. As shown in Fig. S1, the matrix W_{in} maps the reservoir input signal $\mathbf{i}(t)$ into a high-dimensional hidden layer. The reservoir network adjacency matrix $\mathcal{A}$ usually satisfies “the echo-state property” where the dynamical state of the network is confined to a single attractor [1]. The echo-state property is referred to as the ability of the reservoir to retain past information from the input signal for a limited time. This property is essential to effectively process time-varying signals. Activated by the sequence of reservoir input signals $[\mathbf{i}(1), \mathbf{i}(2), \dots, \mathbf{i}(t)]$, the hidden layer state is updated step-by-step according to

$$\mathbf{r}(t + \delta t) = (1 - \alpha)\mathbf{r}(t) + \alpha \cdot \tanh[\mathcal{A} \cdot \mathbf{r}(t) + W_{in} \cdot \mathbf{i}(t) + W_{bias}], \quad (\text{S0.1})$$

where α is the leakage parameter that adjusts the learning rate, δt is the time step of the dynamical evolution of the network state, and the activation function is the hyperbolic tangent function ($\tanh$). The bias vector W_{bias} is composed of equal constants w_b , whose role inside the $\tanh$ function is to shift the small signals into the linear region in the activation function [4]. Let D_i , D_r and D_o be the dimensions of the input signal $\mathbf{i}(t)$, the hidden network state $\mathbf{r}(t)$, and the output signal $\mathbf{o}(t)$, respectively. The dimension of the input matrix W_{in} is $D_r * D_i$, whose elements are chosen uniformly from the interval $[-\gamma, \gamma]$ prior to training. The dimension of the network adjacency

matrix $\mathcal{A}$ in the hidden layer is $D_r * D_r$, which results in a high-dimensional, symmetric, sparse, random network with link probability p and spectral radius ρ . Typically, D_r needs to be much larger than the input dimension D_i to ensure that the network has a sufficient degree of complexity to “learn” and process complex input signals. The nonzero elements in $\mathcal{A}$ are generated by Gaussian distribution with zero mean and unit variance before training. The dimension of the output matrix $\mathcal{W}_o$ is $D_o * D_r$.

To facilitate training, we calculate, update, and concatenate the vector $\mathbf{r}(t)$ characterizing the dynamical state of the reservoir network into a matrix $\mathcal{R}$ of the dimension $D_r * T_{\text{train}}$, where T_{train} is the training length that includes all the episodes of the training phase. The input signal is concatenated into a matrix $\mathcal{U}$. The output matrix is determined by using Tikhonov regularization [5] as

$$\mathcal{W}_{\text{out}} = \mathcal{U} \cdot \mathcal{R}'(\mathcal{R}' \cdot \mathcal{R}' + \beta \mathcal{I})^{-1}, \quad (\text{S0.2})$$

where $\mathcal{I}$ is the identity matrix of dimension D_r , β is the regularization coefficient, and $\mathcal{R}'$ is the transpose of $\mathcal{R}$. In the testing phase, to measure the machine’s performance and tracking accuracy, we use the difference between the current Cartesian position $[C_x, C_y]$ and the reference position $[D_x, D_y]$ as the root mean square error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{2T_{\text{test}}} \sum_{t=1}^{T_{\text{test}}} [(C_x(t) - D_x(t))^2 + (C_y(t) - D_y(t))^2]}, \quad (\text{S0.3})$$

where T_{test} is the test length. The output of the reservoir neural network is

$$\mathbf{o}(t) = \mathcal{W}_{\text{out}} \mathbf{r}(t). \quad (\text{S0.4})$$

There are six hyperparameters to be optimized: the spectral radius of the reservoir ρ , the scaling factor of the input weights of the input matrix γ , the leakage parameter α , the regularization coefficient β , the link probability p of the reservoir network and the bias constant w_b of the bias matrix. These hyperparameters are selected by the surrogate optimization (surrogateopt) algorithm in Matlab [6].

SUPPMENETARY NOTE 2: GENERATION OF 15 TYPES OF REFERENCE TRAJECTORIES

We describe the 15 reference trajectories (eight periodic and seven chaotic) tested in our work. All the chaotic trajectories are from three-dimensional dynamical systems. For the planar tracking problem, we use two variables.

Chaotic Lorenz reference trajectory

The classic Lorenz chaotic system [7], a simplified model for atmospheric convection, is

$$\begin{aligned} \frac{dx_{1o}}{dt} &= \sigma_{1o}(y_{1o} - x_{1o}), \\ \frac{dy_{1o}}{dt} &= x_{1o}(\rho_{1o} - z_{1o}) - y_{1o}, \\ \frac{dz_{1o}}{dt} &= x_{1o}y_{1o} - \beta_{1o}z_{1o}, \end{aligned} \quad (\text{S0.5})$$

where x_{lo} , y_{lo} and z_{lo} are proportional to the rate of convection, the horizontal and vertical temperature variation, respectively, and $\sigma_{lo} > 0$, $\rho_{lo} > 0$ and $\beta_{lo} > 0$ are system parameters. By adjusting the three parameters, the Lorenz system can generate periodic or chaotic trajectories. We choose the parameter values as $\sigma_{lo} = 10$, $\rho_{lo} = 8/3$ and $\beta_{lo} = 26$ to generate a chaotic reference trajectory and use x_{lo} and y_{lo} to represent a planar trajectory. We scale x_{lo} and y_{lo} to within the range $[-0.5, 0.5]$ according to

$$x_{scaled} = a_s + (b_s - a_s) \cdot \frac{x_o - \min(x_o)}{\max(x_o) - \min(x_o)}, \quad (S0.6)$$

where x_o denotes the original data, and the rescaled data is in the interval $[a_s, b_s]$.

Circular reference trajectory

A circular reference trajectory is periodic, which can be described as

$$\begin{aligned} x_{ci} &= a_{ci} \cdot \cos\left(\frac{2\pi t}{T_p}\right), \\ y_{ci} &= a_{ci} \cdot \sin\left(\frac{2\pi t}{T_p}\right), \end{aligned} \quad (S0.7)$$

where a_{ci} limits the size of the trajectory and is set to be $a_{ci} = 0.5$, t is the time and $T_p = 150$ is a parameter determining the period of the trajectory.

Chaotic Mackey-Glass reference trajectory

The Mackey-Glass system [8, 9] is an infinite dimensional system described by a delayed differential equation:

$$\dot{x}_{mg}(t) = \frac{a_{mg}x_{mg}(t - \tau_{mg})}{1 + (x_{mg}(t - \tau_{mg}))^{c_{mg}}} - b_{mg}x_{mg}(t), \quad (S0.8)$$

where τ_{mg} is the delayed time, a_{mg} , b_{mg} and c_{mg} are parameters. We set $a_{mg} = 0.2$, $b_{mg} = 0.1$ and $c_{mg} = 10$ and choose two delayed times: $\tau_{mg} = 17$ and $\tau_{mg} = 30$. The system generates a low-dimensional chaotic attractor with one positive Lyapunov exponent for $\tau_{mg} = 17$ and a high-dimensional chaotic attractor with two positive Lyapunov exponents for $\tau_{mg} = 30$. We use Eq. (S0.6) to scale the data into the ranges $[-0.5, 0.5]$ and $[-0.4, 0.4]$ for $\tau_{mg} = 17$ and $\tau_{mg} = 30$, respectively. We use the variables $x_{mg}(t)$ and $x_{mg}(t - \tau_{mg})$ to generate a planar reference trajectory.

Periodic figure-8 reference trajectory

The figure of eight is a periodic trajectory described by

$$\begin{aligned} x_{ei} &= a_{ei} \cdot \sin\left(\frac{2\pi t}{T_p}\right), \\ y_{ei} &= b_{ei} \cdot \cos\left(\frac{2\pi t}{T_p/2}\right), \end{aligned} \quad (S0.9)$$

where a_{ei} and b_{ei} are the scaling factors to control the size of the figure and T_p is the parameter determining the period of the trajectory. We set $a_{ei} = 0.25$, $b_{ei} = 0.15$ and $T_p = 150$.

Chaotic Rössler reference trajectory

The Rössler system [10] is given by

$$\begin{aligned}\frac{dx_{ro}}{dt} &= -y_{ro} - z_{ro}, \\ \frac{dy_{ro}}{dt} &= x_{ro} + a_{ro}y_{ro}, \\ \frac{dz_{ro}}{dt} &= b_{ro} + z_{ro}(x_{ro} - c_{ro}),\end{aligned}\tag{S0.10}$$

where a_{ro} , b_{ro} and c_{ro} are system parameters. For $a_{ro} = 0.2$, $b_{ro} = 0.2$ and $c_{ro} = 5.7$, the system generates a chaotic attractor. We take x_{ro} and b_{ro} , normalized into the range $[-0.35, 0.35]$ according to Eq. S0.6, as the dynamical variables associated with the chaotic reference trajectory.

Fermat's spiral reference trajectory

Fermat's spiral [11] has the property that the area between two consecutive full turns around the spiral is invariant. It is described in the polar coordinates as

$$r_{fs}^2 = a_{fs}^2 \theta_{fs},\tag{S0.11}$$

where (r_{fs}, θ_{fs}) are the polar coordinates and a_{fs} is a scaling factor. We set $a_{fs} = 0.5$ and $\theta_{fs} = 2\pi t/f_{fs}$, where $f_{fs} = 100$ is set to adjust the speed of the trajectory. In the Cartesian coordinates, the trajectory is given by

$$\begin{aligned}x &= r \cdot \cos(\theta), \\ y &= r \cdot \sin(\theta),\end{aligned}\tag{S0.12}$$

Heart-shaped reference trajectory

A heart-shaped curve can be described in the polar coordinates as

$$r_{he}(\theta_{he}) = 1 - \sin(\theta_{he}).\tag{S0.13}$$

We set $\theta_{he} = 2\pi t/f_{he}$, where $f_{he} = 250$ is used to adjust the speed of the trajectory. In the Cartesian coordinates, the trajectory is described by Eq. (S0.12).

Astroid reference trajectory

In mathematics, an astroid is a type of roulette curves and a hypocycloid with four cusps. The curve can be described parametrically as

$$\begin{aligned}x_{as} &= a_{as} \cos^3(\theta_{as}), \\ y_{as} &= a_{as} \sin^3(\theta_{as}),\end{aligned}\tag{S0.14}$$

where a_{as} is a scaling factor. we choose $a_{\text{as}} = 0.4$ and $\theta_{\text{as}} = 2\pi t/f_{\text{as}}$, where $f_{\text{as}} = 250$ is used to adjust the speed of the trajectory.

Two types of chaotic trajectories from dynamical systems with a polynomial velocity field

In Ref. [12], 19 distinct examples of chaotic attractors are listed. We choose two examples with the corresponding differential equations given by

$$\begin{aligned}\frac{dx_{\text{sp}}}{dt} &= y_{\text{sp}}, \\ \frac{dy_{\text{sp}}}{dt} &= -x_{\text{sp}} + y_{\text{sp}}z_{\text{sp}}, \\ \frac{dz_{\text{sp}}}{dt} &= 1 - y_{\text{sp}}^2,\end{aligned}\tag{S0.15}$$

and

$$\begin{aligned}\frac{dx_{\text{sp}}}{dt} &= -y_{\text{sp}}, \\ \frac{dy_{\text{sp}}}{dt} &= x_{\text{sp}} - y_{\text{sp}}, \\ \frac{dz_{\text{sp}}}{dt} &= x_{\text{sp}}z_{\text{sp}} + 3y_{\text{sp}}^2.\end{aligned}\tag{S0.16}$$

We take the x_{sp} and b_{sp} as the dynamical variables for the chaotic reference trajectories in the plane and scale them into the range $[-0.4, 0.4]$ according to Eq. S0.6.

Epitrochoid reference trajectory

Epitrochoid is a kind of roulette curve generated by tracing the center of the circle with radius r which rolls around the outside of a fixed circle of radius R . Parametrically, an epitrochoid can be described as

$$\begin{aligned}x_{\text{ep}} &= (a_{\text{ep}} + b_{\text{ep}}) \cos(\theta_{\text{ep}}) - c_{\text{ep}} \cos\left(\left(\frac{a_{\text{ep}}}{b_{\text{ep}}} + 1\right)\theta_{\text{ep}}\right), \\ y_{\text{ep}} &= (a_{\text{ep}} + b_{\text{ep}}) \sin(\theta_{\text{ep}}) - c_{\text{ep}} \sin\left(\left(\frac{a_{\text{ep}}}{b_{\text{ep}}} + 1\right)\theta_{\text{ep}}\right),\end{aligned}\tag{S0.17}$$

where a_{ep} , b_{ep} and c_{ep} are parameters. We choose $a_{\text{ep}} = 5$, $b_{\text{ep}} = 3$, $c_{\text{ep}} = 5$, and $\theta_{\text{ep}} = 2\pi t/f_{\text{as}}$ with $f_{\text{as}} = 200$ to tune the speed of the trajectory.

Chaotic Chua-circuit reference trajectory

Chua's circuit can generate a double-scroll chaotic attractor [13]. One set of differential equations describing the Chua's circuit is

$$\begin{aligned}\frac{dx_{\text{ch}}}{dt} &= \alpha_{\text{ch}}(y_{\text{ch}} - x_{\text{ch}} - h_{\text{ch}}), \\ \frac{dy_{\text{ch}}}{dt} &= x_{\text{ch}} - y_{\text{ch}} + z_{\text{ch}}, \\ \frac{dz_{\text{ch}}}{dt} &= -\beta_{\text{ch}}y_{\text{ch}},\end{aligned}\tag{S0.18}$$

where

$$h_{\text{ch}} = \mu_{\text{ch}_1}x_{\text{ch}} + 0.5(\mu_{\text{ch}_0} - \mu_{\text{ch}_1}) \cdot (|x_{\text{ch}} + 1| - |x_{\text{ch}} - 1|),$$

α_{ch} , β_{ch} , μ_{ch_0} and μ_{ch_1} are parameters. We choose $\alpha_{\text{ch}} = 14.405$, $\beta_{\text{ch}} = 28$, $\mu_{\text{ch}_0} = -1.143$ and $\mu_{\text{ch}_1} = -0.714$ to generate a chaotic attractor, with the variables x_{ch} and y_{ch} scaled into the respective ranges $[-0.5, 0.5]$ and $[-0.3, 0.3]$ representing the reference trajectory.

Talbot reference trajectory

The Talbot curve is a negative pedal of an ellipse with respect to the ellipse's center [14]. Parametrically, the curve is described as

$$\begin{aligned}x_{\text{ta}} &= \frac{(a_{\text{ta}}^2 + c_{\text{ta}}^2 \sin^2(\theta_{\text{ta}})) \cos(\theta_{\text{ta}})}{a_{\text{ta}}}, \\ y_{\text{ta}} &= \frac{(a_{\text{ta}}^2 - 2c_{\text{ta}}^2 + c_{\text{ta}}^2 \sin^2(\theta_{\text{ta}})) \sin(\theta_{\text{ta}})}{b_{\text{ta}}},\end{aligned}\tag{S0.19}$$

where a_{ta} , b_{ta} and c_{ta} are parameters. We choose $a_{\text{ta}} = 1.1$, $b_{\text{ta}} = 0.666$, $c_{\text{ta}} = 1$, and $\theta_{\text{ta}} = 2\pi t/f_{\text{as}}$ with $f_{\text{as}} = 400$ to set the speed of the curve. The coordinates x_{ta} and y_{ta} are scaled into the respective ranges $[-0.4, 0.4]$ and $[-0.3, 0.3]$.

Lissajous reference trajectory

A Lissajous curve, also called a Lissajous figure or Bowditch curve, is described by the following parametric equations

$$\begin{aligned}x_{\text{li}} &= A_{\text{li}} \sin(a_{\text{li}}\theta_{\text{li}} + c_{\text{li}}), \\ y_{\text{li}} &= B_{\text{li}} \sin(b_{\text{li}}\theta_{\text{li}}),\end{aligned}\tag{S0.20}$$

where A_{li} , B_{li} , a_{li} , b_{li} and c_{li} are parameters that control the shape of the Lissajous curve. We choose $A_{\text{li}} = 1$, $B_{\text{li}} = 1$, $a_{\text{li}} = 1$, $b_{\text{li}} = 3$, $c_{\text{li}} = \pi/4$, and $\theta_{\text{li}} = 2\pi t/f_{\text{li}}$ with $f_{\text{li}} = 300$ to set the speed of the trajectory. The coordinates x_{li} and y_{li} are scaled into the range $[-0.3, 0.3]$.

SUPPLEMENTARY NOTE 3: BRIDGE TO REFERENCE TRAJECTORY

It is often the case that the starting position of the end effector is not on the given reference trajectory, requiring a “bridge” to drive the end effector from the starting position to the trajectory. Let T_x and T_y be the closest points of the reference trajectory to the starting position C_x, C_y of the end effector. The required time to “cross the bridge” is $t_b = \sqrt{(C_x - T_x)^2 + (C_y - T_y)^2} / dt$. Since a smooth trajectory in the Cartesian coordinates may not lead to a smooth trajectory in the polar coordinates, we convert the two points into angular positions according to Eqs. (7) and (8) in the main text. This way, the bridge will be in the angular space, i.e., we generate a series of points traveling from the initial point to the reference trajectory in terms of the angles.

A method to build the bridge is through cubic polynomial trajectory generation [15]. In particular, consider the cubic polynomial of the form

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3, \quad (\text{S0.21})$$

where $q(t) = [q_1(t), q_2(t)]$ is the angular position. Setting the velocities of the initial and the end point as $\dot{q}(0)$ and $\dot{q}(t_b)$, we can write the equations to describe the cubic polynomial as

$$\begin{aligned} q(0) &= a_0, \\ q(t_b) &= a_0 + a_1 t_b + a_2 t_b^2 + a_3 t_b^3, \\ \dot{q}(0) &= a_1, \\ \dot{q}(t_b) &= a_1 + 2a_2 t_b + 3a_3 t_b^2. \end{aligned} \quad (\text{S0.22})$$

The solutions are

$$\begin{aligned} a_0 &= q(0), \\ a_1 &= \dot{q}(0), \\ a_2 &= \frac{3}{t_b^2}(q(t_b) - q(0)) - \frac{2}{t_b}\dot{q}(0) - \frac{1}{t_b}\dot{q}(t_b), \\ a_3 &= -\frac{2}{t_b^3}(q(t_b) - q(0)) + \frac{1}{t_b^2}(\dot{q}(t_b) + \dot{q}(0)). \end{aligned} \quad (\text{S0.23})$$

Given the angular positions and velocities of both robot arms, we can then build a smooth “bridge” in the angular space using Eq. (S0.23) to make the robot arm move smoothly and gradually approach the starting section of the trajectory.

SUPPLEMENTARY NOTE 4: ALTERNATING AND CONTINUOUS TRACKING OF A VARIETY OF REFERENCE TRAJECTORIES

We present results of continuously tracking 15 different types of regular or chaotic reference trajectories, one after another, and address pertinent issues such as robustness, tracking speed tolerance, and variations in the training parameters.

Tracking Control

Figure S2 shows the results of continuous tracking of 15 different types of reference trajectories, regular or chaotic. For the first trajectory, we set all initial states including that of the reservoir

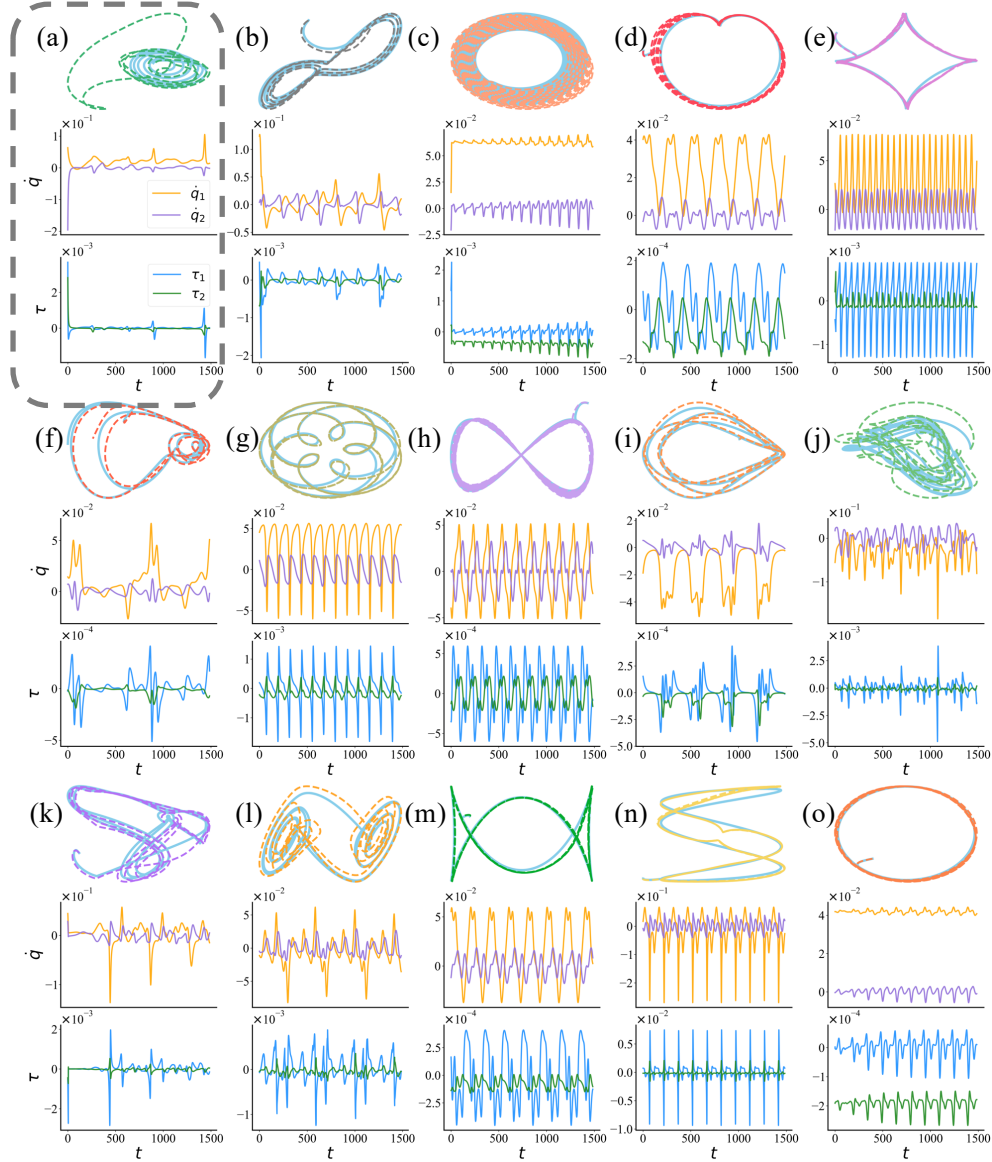


FIG. S2. Tracking control of 15 different kinds of reference trajectories: (a) chaotic Rössler, (b) chaotic Lorenz, (c) Fermat's spiral, (d) heart, (e) astroid, (f) polynomial chaotic, (g) epitrochoid, (h) eight symbol, (i) a different type of polynomial chaotic, (j) chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$, (k) chaotic Mackey-Glass with $\tau_{\text{mg}} = 30$, (l) chaotic Chua, (m) Talbot, (n) Lissajous, and (o) circle. For each reference trajectory, Tracking of the position, speed, and the control signal from the reservoir controller are shown.

controller to zero. For the remaining 14 reference trajectories, the initial states are the end states associated with the preceding trajectory. For each trajectory, we build a bridge to guide the robot arms to move forward to the trajectory. These results demonstrate the power and great flexibility of our reservoir-computing based tracking control.

We present the results of robustness against disturbance and noises for three reference trajectories: circle, chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$, and the eight symbol, as shown in Fig. S3. The RMSE threshold is set to be $\text{RMSE}_{\text{thre}} = 0.18$. It can be seen that, for the three trajectories, the

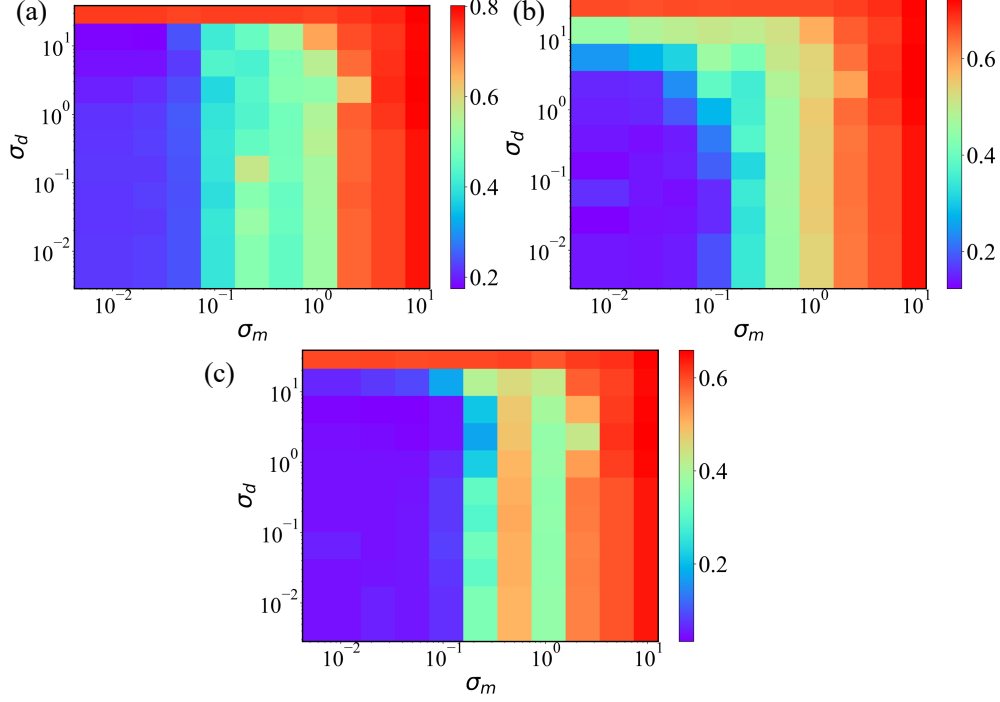


FIG. S3. Robustness test under disturbance and measurement noise. (a-c) Results of robustness test with respect to the circular, chaotic Mackey-Glass with $\tau_{mg} = 17$ and eight symbol reference trajectories, respectively. The color represents the ensemble-averaged RMSE values from 50 independent realizations. The results indicate that the reservoir-computing based tracking control framework is robust against disturbance and noise.

control system is robust against measurement noise of amplitude below 10^{-1} and disturbance of magnitude below 0.5×10^1 .

Robustness against uncertainties

We further test the robustness of tracking control against parameter uncertainties. To be concrete, we assume that the uncertainty occurs in the lengths of the robot arms. Figures S4(a) and S4(b) show the results of tracking a circular and the chaotic Mackey-Glass with $\tau_{mg} = 17$ reference trajectories, respectively. For the circular trajectory, the system is able to maintain stable tracking in spite of the difference in the robot arm lengths. For the chaotic Mackey-Glass trajectory, the parameter uncertainty has a devastating effect on tracking control. The reason is that this trajectory can be quite to the origin, so even a small difference in the arm length will make it impossible for the end effector to come close to the origin. (For the figure-eight reference trajectory, since it crosses the origin, even a slight difference in the arm lengths l_1 and l_2 will make it impossible for the end effector to approach the origin.)

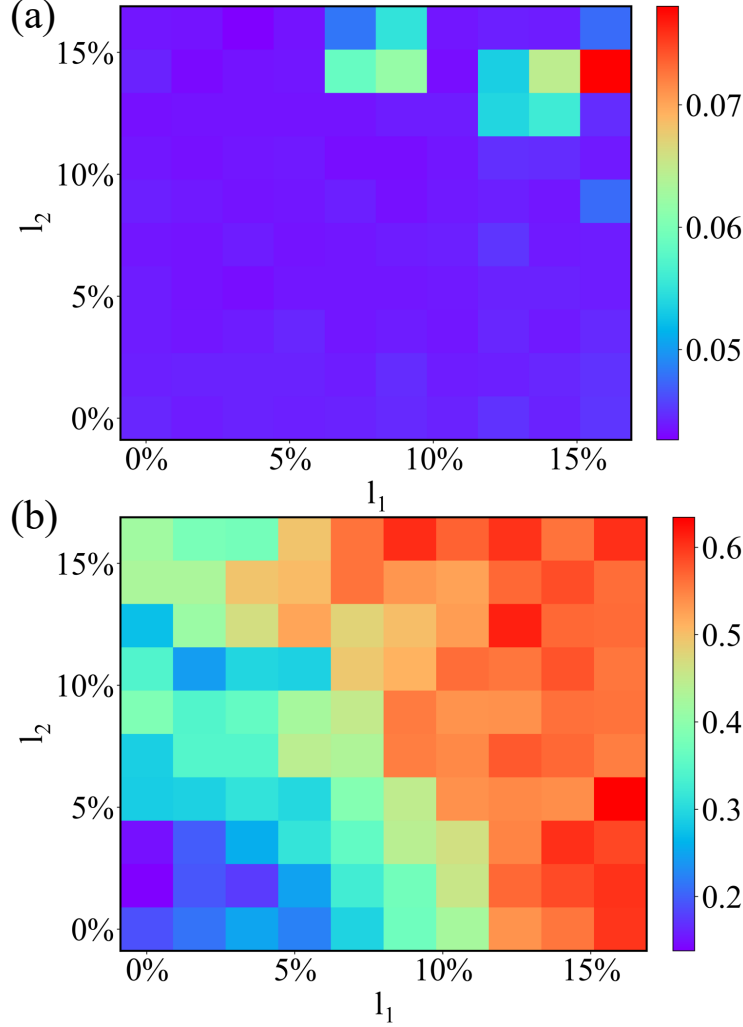


FIG. S4. Robustness against parameter uncertainty. (a,b) Tracking a circular and the chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$ reference trajectories, respectively. The controller is trained with the arm lengths $l_1 = 0.5$ and $l_2 = 0.5$, but tested with different values of l_1 and l_2 . The color represents the ensemble averaged values of RMSE (from 50 realizations). Insofar as the reference trajectory does not come arbitrarily close to the origin, stable tracking-control performance can be achieved.

Effect of varying training parameters

We evaluate the computational time and the performance of the machine-learning based tracking controller with respect to variations in the training parameters. Figure S5(a) shows, the color-coded computation time required for the training phase in the parameter plane of training length T_{train} and network size N , which is averaged over the four reference trajectories in Fig. 6 in the main text. As either T_{train} or N increases, the computation time grows in an algebraic (not exponential) fashion. Figure S5(b) shows the computation time in the parameter plane (T_{ep}, η) . It can be seen that there is little dependence of the time on these parameters. Figures S6(a-c) show the performance versus N and T_{train} , for the circular, chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$ and eight-figure reference trajectories, respectively. It can be seen that a larger network size in general will

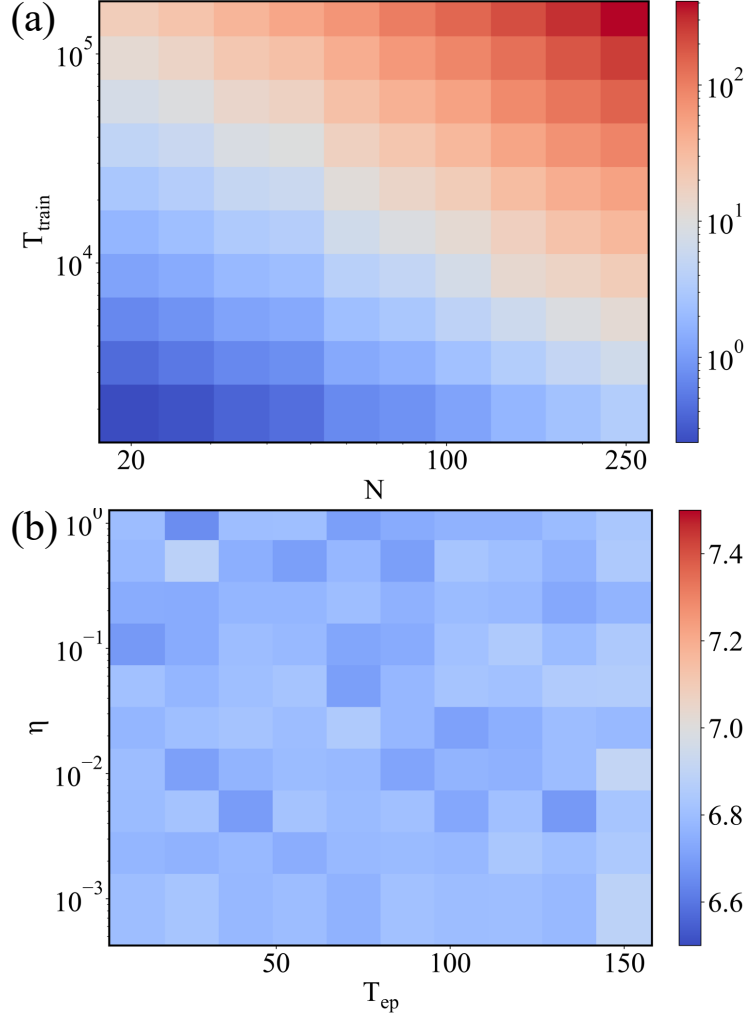


FIG. S5. Required computational time with different parameters in the training phase, averaged over the four reference trajectories in Fig. 6 in the main text. Shown is the training time required for 50 runs with variations (a) in the neural-network size N and training time T_{train} , and (b) reset time T_{ep} and the amplitude η of the stochastic, uniformly distributed control signal employed in the training. With proper choices of the parameters in machine learning and training, satisfactory training can be realized.

lead to better tracking performance. The training length T_{train} , insofar it is reasonable, has little effect on the performance. Likewise, the changes in the training reset time T_{ep} and different values of the amplitude η of the stochastic control input used in training, insofar as they are in a reasonable range, tend not to have a severe effect on the training performance. Figures S7(a-c) show the performance with respect to varying the reset time T and the amplitude η of the stochastic control input signal employed in the training phase for the circular, chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$ and eight-figure reference trajectories, respectively. It can be seen that appropriate combinations of T and η should be chosen to guarantee satisfactory performance.

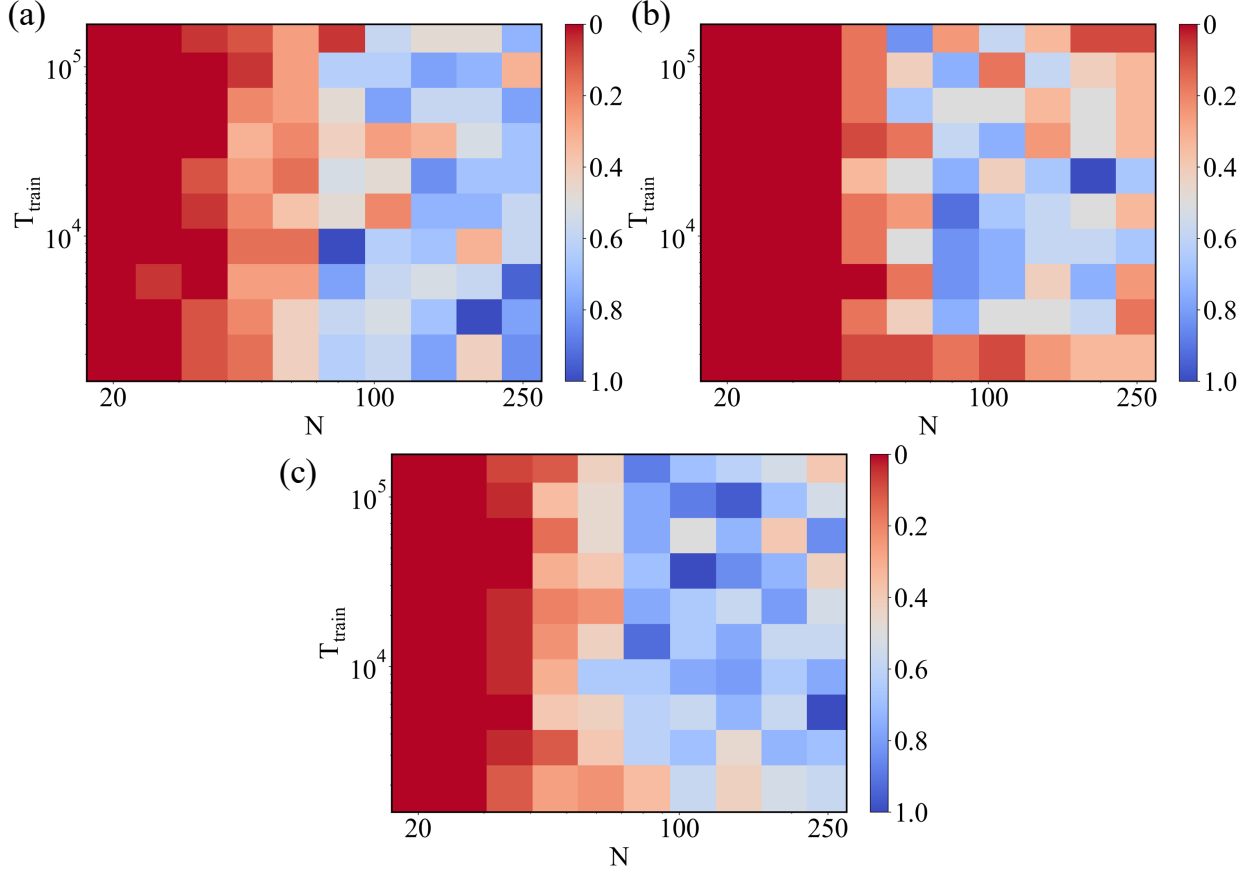


FIG. S6. Effects of network size and training time on performance. (a-c) Relative success rate obtained from 50 trials versus changes in the network size N and training time T_{train} for the circular, chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$ and eight-figure reference trajectories, respectively. There exist appropriate parameter ranges for satisfactory performance.

SUPPLEMENTARY NOTE 5: PERTINENT ISSUES OF TRACKING CONTROL

Safe region of initial conditions for control success

For each episode of training, the initial arm positions are randomly selected. In general, the tracking success rate in the testing phase would depend on the choice of the initial arm positions. It is desired that control tracking be successful for arbitrary choices of the initial condition. Here we introduce the concept of “safe region” to meet this requirement, where any initial condition from this region can guarantee a higher tracking success probability. We find that, if the end effector is initialized in the third or fourth quadrant, the success rate will increase dramatically compared with initial conditions from other regions. Figures S8(a) and S8(b) illustrate a random initial state of the two-arm system and a state from the safe region, respectively. In each case, the q_1 and q_2 coordinates are confined in the yellow and purple region with the ranges $[4, 6]$ and $[-0.1, -2.5]$, respectively, for 15 different types of reference trajectories. Figure S8(c) shows the tracking success rate of these trajectories for random (orange) and safe (blue) region initialization. For each trajectory, tracking control is carried out 100 times and the success rate is calculated

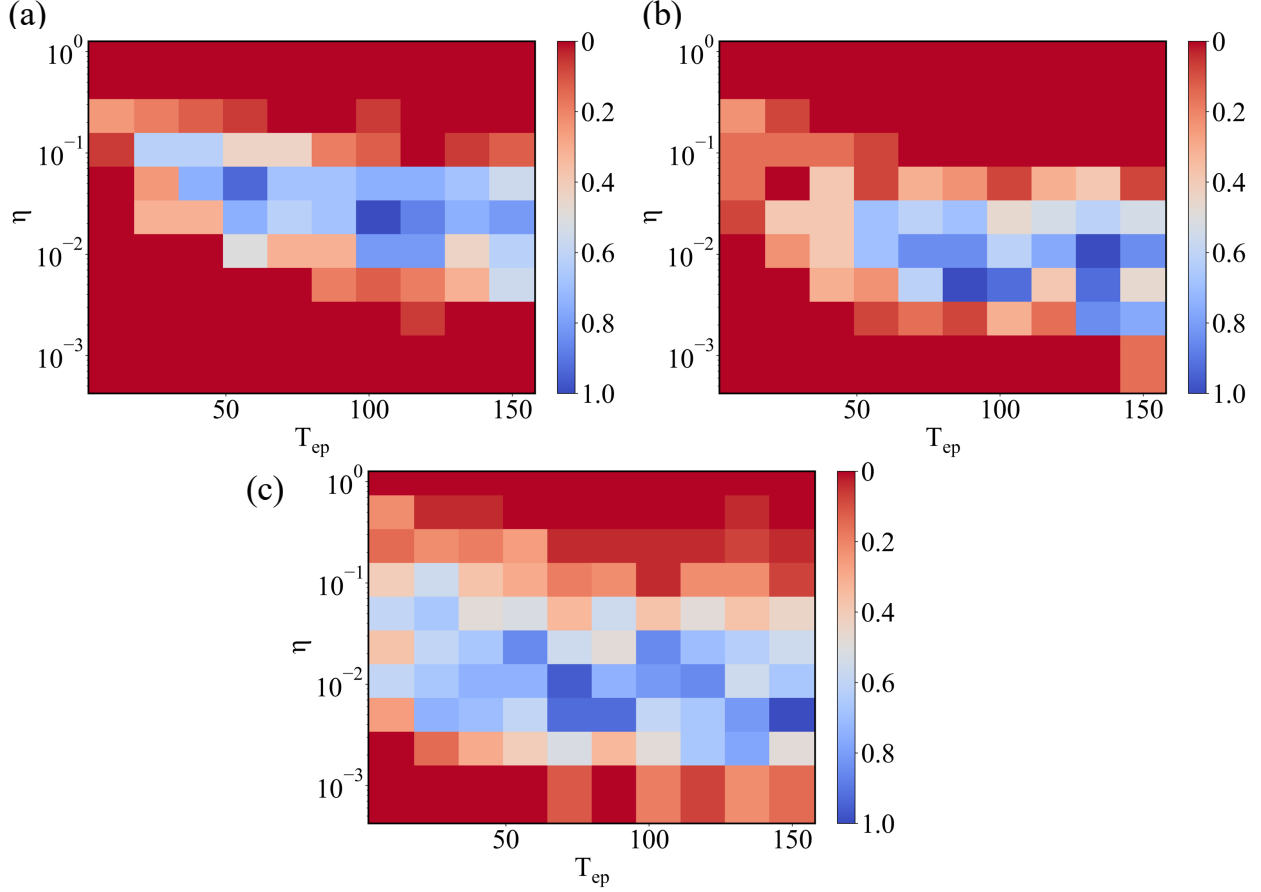


FIG. S7. Effects of reset time and stochastic input control signal on performance. (a-c) Relative success rate obtained from 50 trials versus changes in the reset time T and the amplitude η of the stochastic control input signal employed in the training phase for the circular, chaotic Mackey-Glass with $\tau_{\text{mg}} = 17$ and eight-figure reference trajectories, respectively. There exist appropriate parameter ranges for satisfactory performance.

as $P = N_s/100$, where N_s is the number of trials in which the RMSE is smaller than a small threshold value $\text{RMSE}_{\text{thre}} = 0.18$. It can be seen that, if the robot arms are prepared from an initial condition in the safe region, near perfect success rate can be achieved.

Tracking speed tolerance

During the training, the amplitude of the generated noise fed into the two-arm system as the control input varies in the range of $[-2 \times 10^{-2}, 2 \times 10^2]$. How fast can the noisy torques be applied? That is, what is the limit to the tracking speed? To test the tolerance of the trajectory tracking speed, we take two periodic trajectories: circle and the eight symbol with period T_p as shown in Figs. S9(a) and S9(b), respectively, for three different values of T_p . For a fixed period, we conduct tracking control 50 times, each of duration $15,000dt$. For $T_p = 200$, tracking is accurate. However, for $T_p = 20$ and $T_p = 400$, significant tracking errors arise. The ensemble-averaged RMSE and the standard deviation are shown in Fig. S9(c). It can be seen that when the period is within the range $T_p \in [100, 300]$, acceptable performance is achieved, where an acceptable case

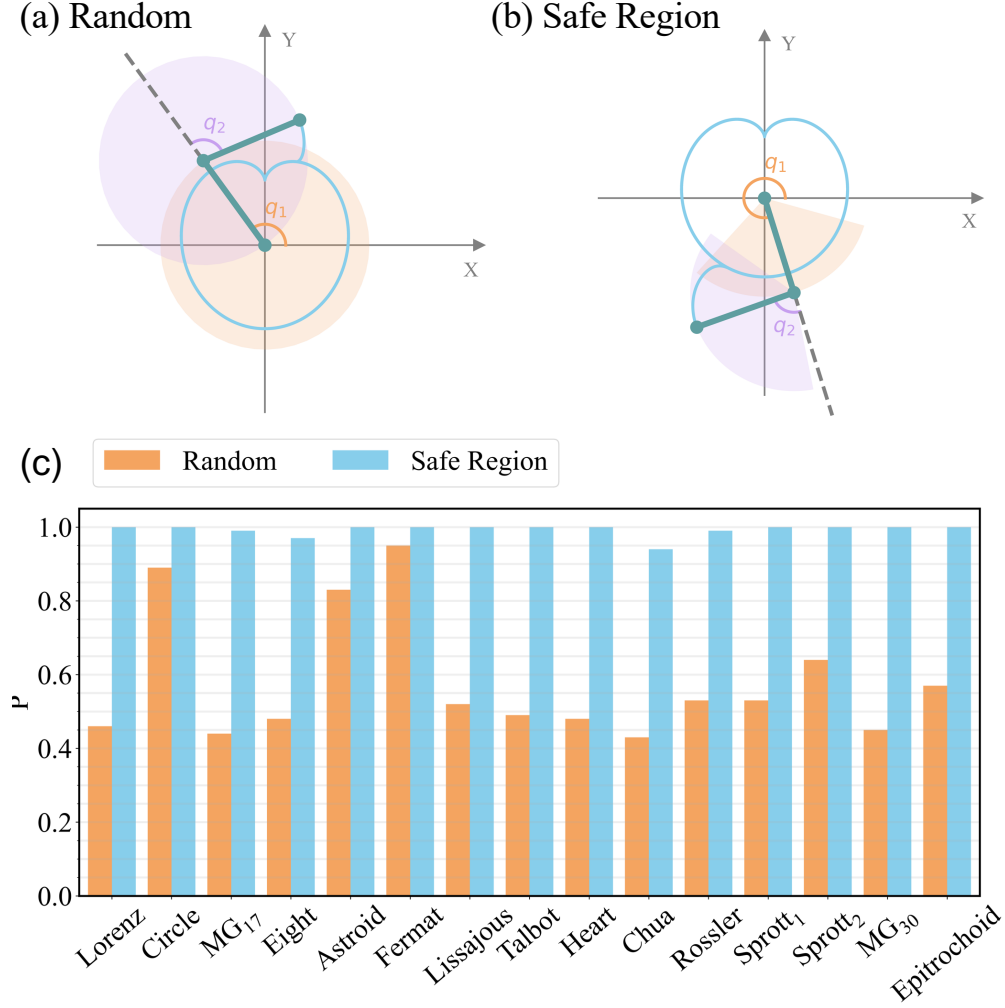


FIG. S8. Safe region for initializing the robot arms and tracking success. (a) A random initial state of the two-arm system. (b) An initial state in the safe region. (c) Tracking success rate for initial conditions from the safe region (blue) compared with that for random initial conditions (orange). Preparing the dynamical system from initial conditions from the safe region for training the reservoir controller can guarantee nearly perfect success rate.

is one in which the average RMSE is below some empirical threshold, e.g., $\text{RMSE}_{thre} = 0.18$ (the horizontal dotted line). The ability to track a fast reference signal depends on the amplitude of the generated noise and the episode reset time T_{ep} . The reservoir machine can learn more relevant information from tracking a reference with a specific speed range. However, this necessitates a larger exploration space.

Robustness against variations in training parameters

A proper choice of the key parameters involved in the training is essential to the working of the reservoir controller, and it is important to assess the impact of the parameter variations on the tracking performance. Here we present results with varying four such parameters for tracking

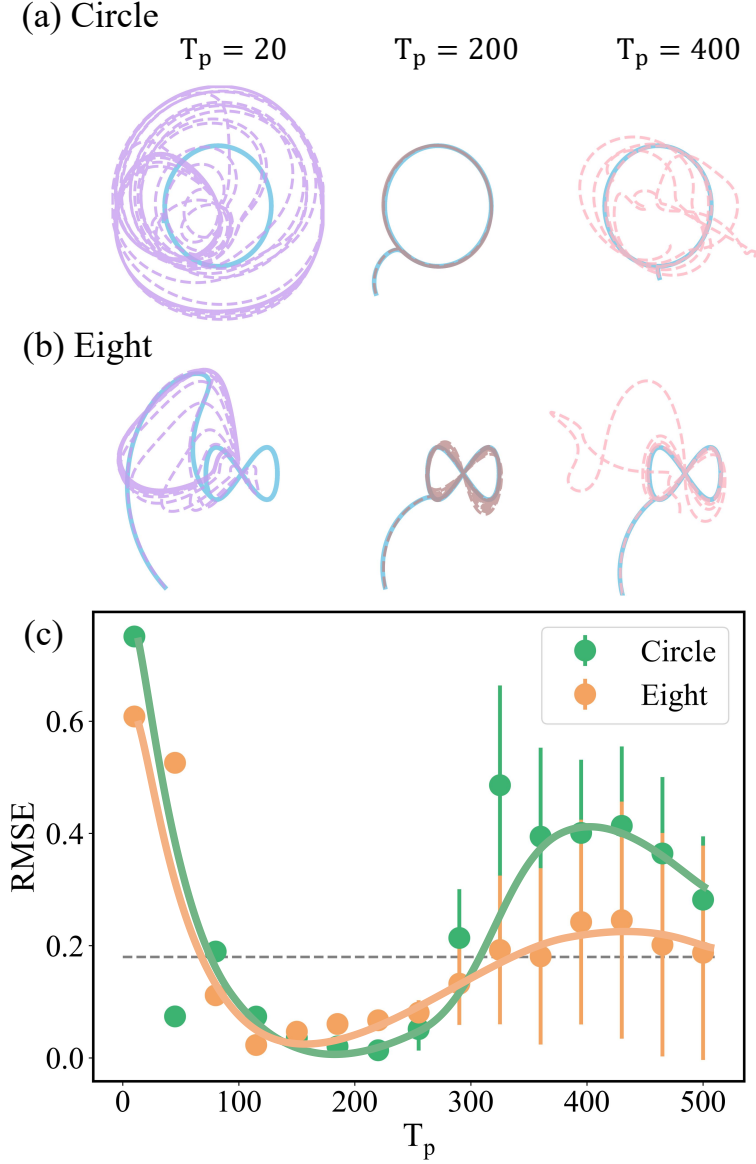


FIG. S9. Speed tolerance for successful tracking control. (a,b) Two types of periodic reference trajectories (circle and eight figure, respectively), for three different values of the period T_p . (c) Ensemble-averaged RMSE versus the period T_p for the two types of trajectories.

a chaotic Lorenz trajectory: reservoir controller network size N , training length T_{train} , episode length T_{ep} , and amplitude η of the stochastic input control signal. We consider two scenarios where, in each case, we fix two parameters, vary the other two systematically, and analyze the tracking performance. Figure S10(a) shows, for fixed $T_{\text{ep}} = 80$ and $\eta = 2 \times 10^{-2}$, color-coded rate of successful tracking control defined as $\text{RMSE} < 0.18$ in the parameter plane (N, T_{train}) , where each value of the success rate is obtained from 50 independent trials. It can be seen that the control performance depends more on the network size N : if the network size is too small (e.g., $\lesssim 100$, the success rate is low. In fact, insofar as the network size is above about 100, the success rate is high for a wide range of T_{train} values. Figure S10(b) shows, for fixed $N = 100$ and $T_{\text{train}} = 10000$, the success rate in the plane (T_{ep}, η) . The result suggests that the choice of the

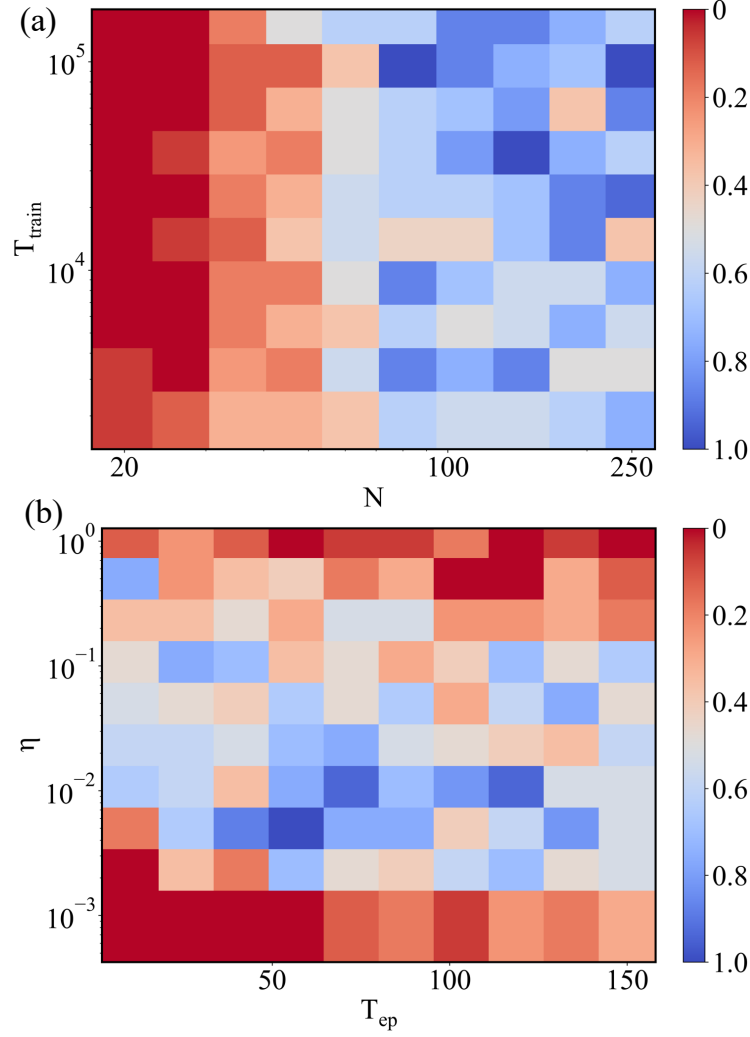


FIG. S10. Impact of variations of certain training parameters on tracking-control performance. (a) For fixed episode length $T_{\text{ep}} = 80$ and input stochastic control amplitude $\eta = 2 \times 10^{-2}$, relative success rate as the network size N and training time T_{train} vary. (b) For fixed $N = 100$ and $T_{\text{train}} = 10000$, success rate in the parameter plane (T_{ep}, η) . In both (a) and (b), each value of the success rate is obtained from 50 independent experiments.

episode length T_{ep} and the noise amplitude η is important for the training success.

SUPPLEMENTARY NOTE 6: TRACKING CONTROL WITH FEED-FORWARD NEURAL NETWORKS

Feed-forward neural networks (FNNs) are networks that have directed connections between artificial neurons and do not possess any loop structure [16]: information flows only in the forward direction. In artificial intelligence, FNNs have been widely used. Here we test if FNNs can replace reservoir computing for tracking control. In particular, we consider a standard FNN with a classic layered architecture: an input layer, several hidden layers, and an output layer, as shown

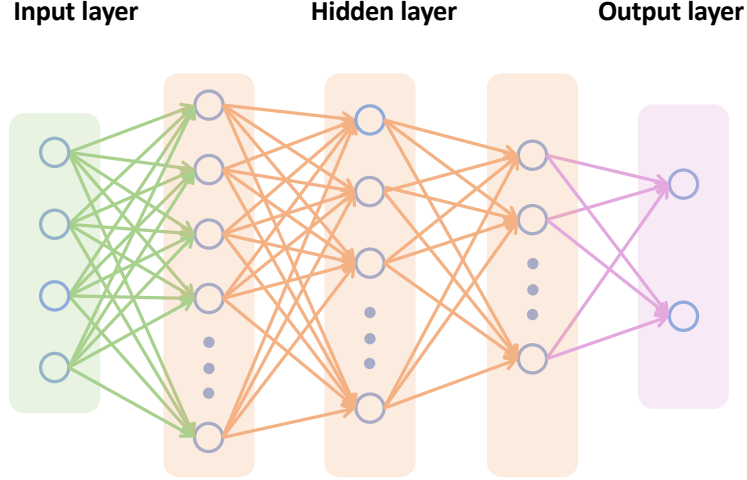


FIG. S11. Architecture of the feed-forward neural network to replace reservoir computing for tracking control. The FNN includes one input layer, three hidden layers, and one output layer.

in Fig. S11. The forward direction is the direction from the input to the output layer. Each layer consists of a number of neurons and there are only inter-layer connections, which are directed and weighted, from neurons in one layer to another in the forward direction. A sufficiently large FNN can perform as a universal approximator, which can capture the hidden complex relationship between the input and output data through a training process in which the connection weights are adjusted to achieve the desired output. We use the following hyperbolic tangent sigmoid as the activation function that compresses the output into the range $[-1, 1]$:

$$\text{tansig}(x_{tan}) = \frac{2}{1 + e^{-2x_{tan}}} - 1. \quad (\text{S0.24})$$

We exploit FNN to construct a mapping between the error terms and the control signal required for tracking, similar to the system configuration in Fig. 4(a) in the main text. We use scaled conjugate gradient backpropagation (SCG) [17] to perform the training. In the three hidden layers, the respective number of neurons are $[50, 30, 10]$. Subtracting the observed state $\mathbf{y}(t)$ from the desired observation state $\mathbf{y}_d(t + dt)$, we obtain four inputs for the FNN: E_x , E_y , $E_{\dot{q}_1}$, and $E_{\dot{q}_2}$. At each step, we have two scalar outputs: the control signal $u = [\tau_1, \tau_2]$. Since the Cartesian positions of the end effector and the angular velocities have different magnitudes, it is useful to normalize the data for training, which will ensure that the FNN learns the signal in the input channels uniformly to avoid bias towards learning any particular input [18]. We use z -score normalization [19]: $z = (x_z - \bar{x}_z)/S_z$ to normalize both the input and output, where $\bar{x}_z$ and S_z are the mean and standard deviation of the data x , respectively. We divide the data into a training and a testing segment, with 80% and 20% of the total data, respectively. In the training phase, we take 20% of the training data as the validation data for calculating the loss function. The training and testing results are shown in Fig. S12, where the four rows represent four examples of reference trajectories: chaotic Lorenz, circular, chaotic Mackey-Glass with $\tau_{mg} = 17$, and the eight symbol. The first column records the loss function and the mean absolute error (MAE) [20] versus the training epoch, which can be calculated by the sum of the absolute errors divided by the sample size. From the first column, it can be seen that the training and validation MAEs decrease and then gradually approach a steady value. The second column of Fig. S12 shows the testing phase

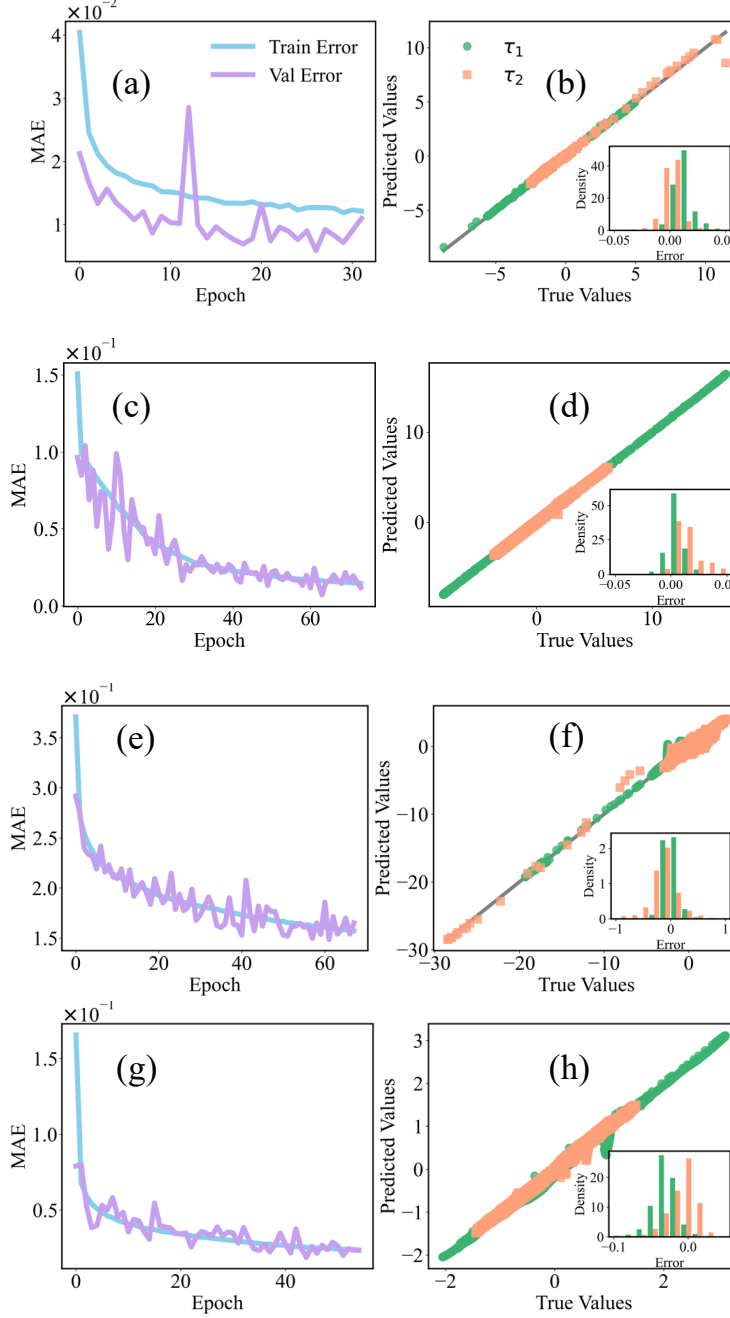


FIG. S12. Examples of successful tracking control using FNNs. The four rows show four examples of trajectories: chaotic Lorenz, circular, chaotic Mackey-Glass system with $\tau_{mg} = 17$ and the eight-symbol. (a,c,e,g) Training MAE of the loss function. (b,d,f,h) Predicted versus true values, with the insets displaying the statistics of the error between the predicted and true values. There exists a map between the state error terms and the control signal, and the FNN controller has the ability to output the proper control signal and track different trajectories.

results: the normalized true values of the control signal $u = [\tau_1, \tau_2]$ and the predicted values as the output of the FNN, where the green circles and orange squares represent τ_1 and τ_2 , respectively. The grey lines are for facilitating the interpretation of the results. Subtracting the predicted from

the true values, we obtain the statistics of the errors, as shown in the four insets. These results indicate that, by designating the error terms as the input and the control signal as the output for the FNN, satisfactory learning and prediction can be achieved. That is, the FNN can successfully yield the mapping between the state errors and the control signal.

In general, the tracking and control of dynamic systems require a neural architecture that can maintain a memory of past inputs to process future inputs - time-series analysis. Recurrent neural networks (RNNs) have demonstrated success in such scenarios. However, RNNs and their advanced versions, such as LSTM and GRU, possess a high computational time complexity and may not be suitable for online applications. Reservoir computing (RC) overcomes the computational complexity by excluding the reservoir weights from training. Instead, the reservoir weights are fixed, and training is only necessary to tune the output matrix or the readout layer, which can be achieved quickly using linear regression.

FNNs are essentially RNNs unfolded in time and thus face the same challenges as RNNs. While FNNs may work well for the example of the two-link robotic arm with two degrees of freedom (2-DoF) demonstrated in this paper, RC is the preferred architecture for tracking higher degree-of-freedom dynamic systems operating at high speeds.

-
- [1] Jaeger, H. The “echo state” approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* **148**, 13 (2001).
 - [2] Maass, W., Natschläger, T. & Markram, H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neu. Comp.* **14**, 2531–2560 (2002).
 - [3] Appeltant, L. *et al.* Information processing using a single dynamical node as complex system. *Nat. Commun.* **2**, 1–6 (2011).
 - [4] Carroll, T. L. Optimizing memory in reservoir computers. *Chaos* **32**, 023123 (2022).
 - [5] Bishop, C. M. Training with noise is equivalent to tikhonov regularization. *Neural computation* **7**, 108–116 (1995).
 - [6] Surrogate optimization for global minimization of time-consuming objective functions - MATLAB. <https://www.mathworks.com/help/gads/surrogateopt.html>.
 - [7] Lorenz, E. N. Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, 130–141 (1963).
 - [8] Mackey, M. C. & Glass, L. Oscillation and chaos in physiological control systems. *Science* **197**, 287–289 (1977).
 - [9] Wernecke, H., Sándor, B. & Gros, C. Chaos in time delay systems: An educational review. *Phys. Rep.* **824**, 1–40 (2019).
 - [10] Rössler, O. E. Equation for continuous chaos. *Phys. Lett. A* **57**, 397–398 (1976).
 - [11] Zhao, H. *et al.* Connected fermat spirals for layered fabrication. *ACM Trans. Graph. (TOG)* **35**, 1–10 (2016).
 - [12] Sprott, J. C. Some simple chaotic flows. *Phys. Rev. E* **50**, R647 (1994).
 - [13] Lü, J. & Chen, G. Generating multiscroll chaotic attractors: theories, methods and applications. *Int. J. Bif. Chaos* **16**, 775–858 (2006).
 - [14] Lockwood, E. H. *A book of curves* (Cambridge University Press, 1967).
 - [15] Kucuk, S. Optimal trajectory generation algorithm for serial and parallel manipulators. *Robot. Comp.-*

- Inte. Manufact.* **48**, 219–232 (2017).
- [16] Svozil, D., Kvasnicka, V. & Pospichal, J. Introduction to multi-layer feed-forward neural networks. *Chemome. Intell. Lab. Sys.* **39**, 43–62 (1997).
 - [17] Møller, M. F. A scaled conjugate gradient algorithm for fast supervised learning. *Neu. Net.* **6**, 525–533 (1993).
 - [18] Ormiston, R., Nguyen, T., Coughlin, M., Adhikari, R. X. & Katsavounidis, E. Noise reduction in gravitational-wave data via deep learning. *Phys. Rev. Res.* **2**, 033066 (2020).
 - [19] Jain, A., Nandakumar, K. & Ross, A. Score normalization in multimodal biometric systems. *Pat. Recog.* **38**, 2270–2285 (2005).
 - [20] Ji, L. & Peters, A. J. Assessing vegetation response to drought in the northern great plains using vegetation and drought indices. *Remote Sen. Environ.* **87**, 85–98 (2003).