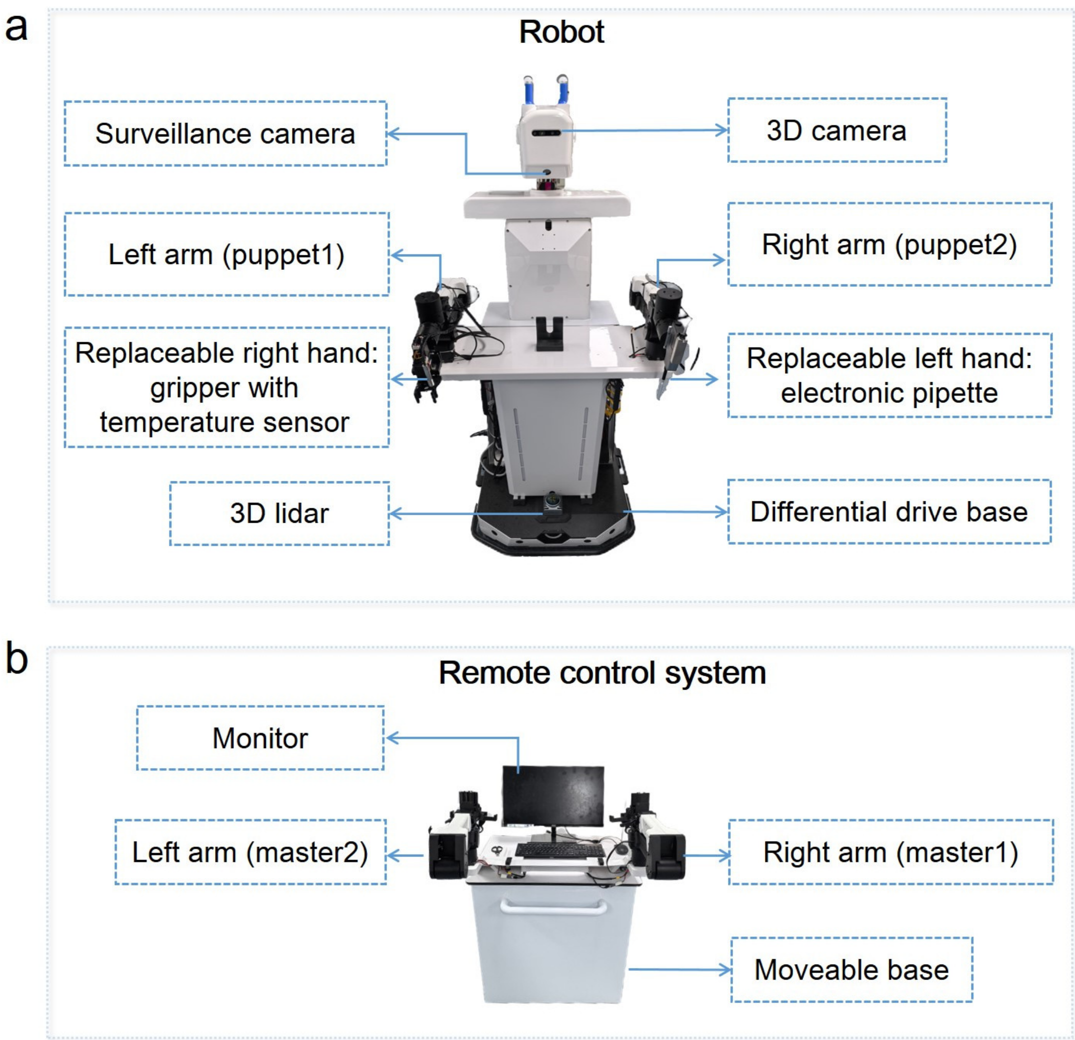
Supplementary Information

This file includes:

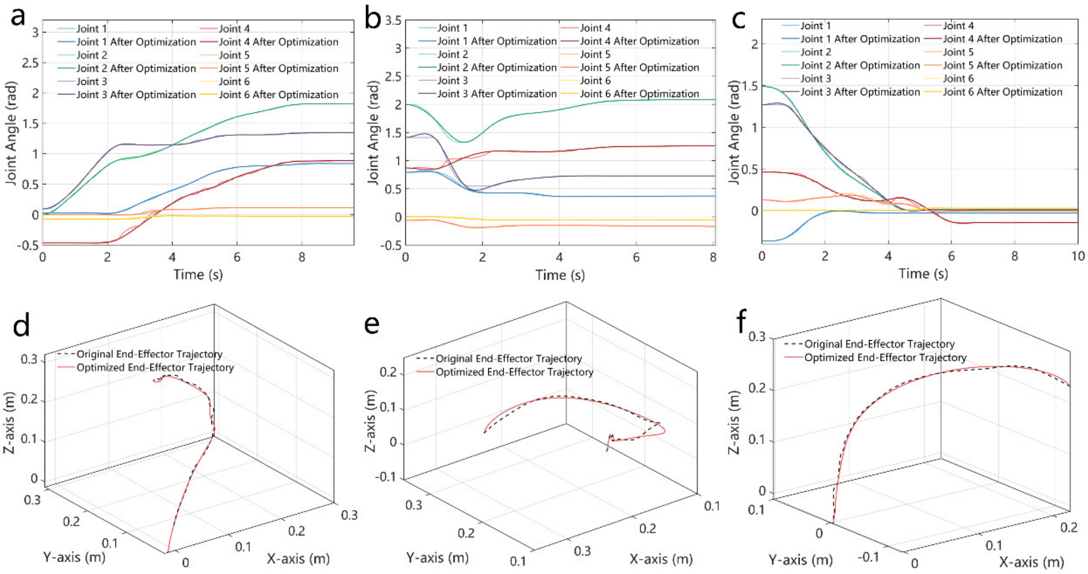
Supplementary Figures 1-11.

Supplementary Tables 1-3.

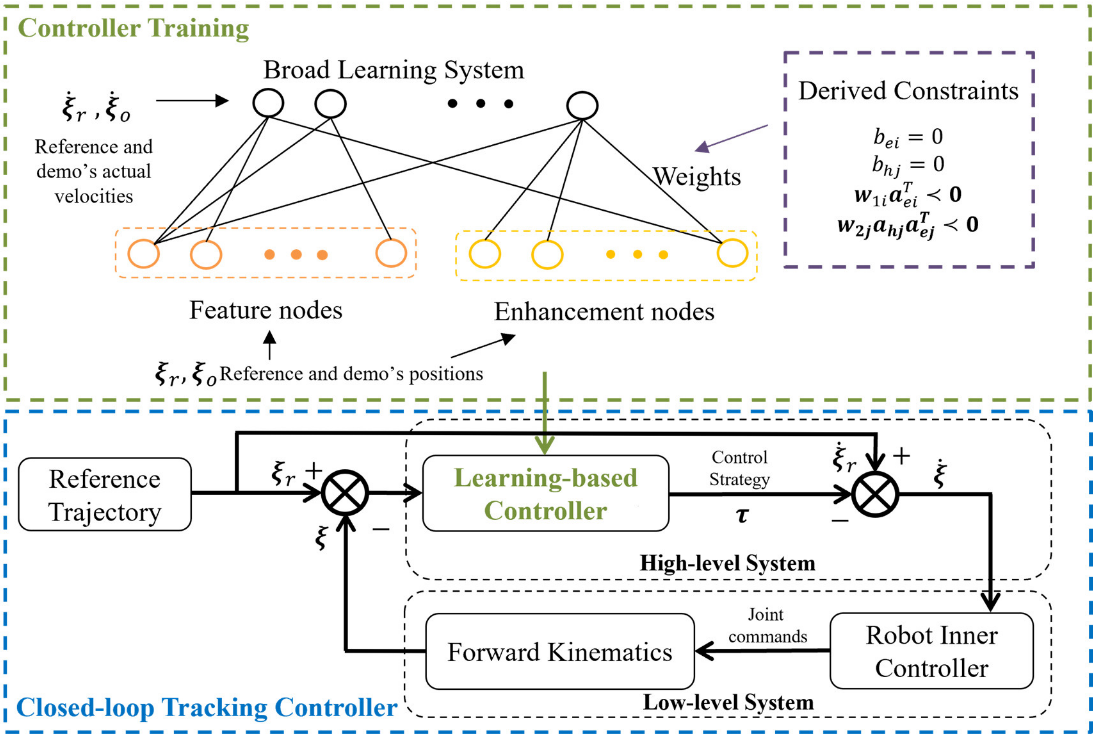
Supplementary Notes 1-5.



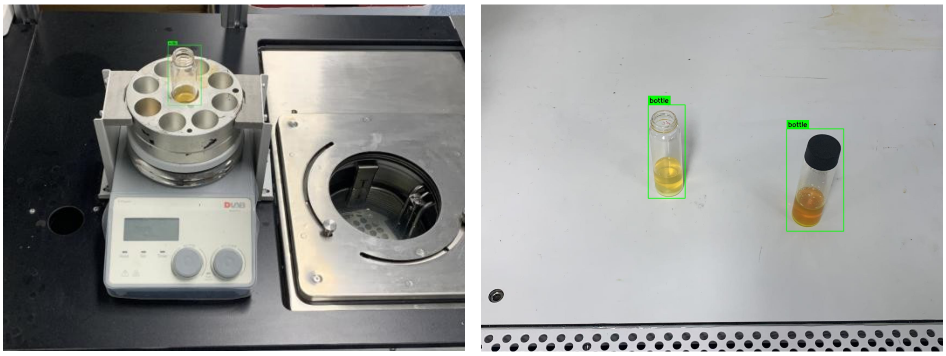
Supplementary Fig. 1: EI architecture. Structure of the EI robot and remote control system of the EI platform.



Supplementary Fig. 2: Comparison of trajectories before and after optimization.
a-c, Joint trajectories; d-f, End-effector trajectories.



Supplementary Fig. 3: Diagram of the proposed control system. Controller training and closed-loop tracking controller.



Supplementary Fig. 4: Bottle detection via YOLO.

Supplementary Note 1: Embodied Action (EA) Library

The EA library, learned directly from human movements, facilitates flexible, compliant, and large-range trajectory planning and control during robotic experiments. It includes motions such as moving to the bottle opening, liquid dispensing position, and bottle placement position.

To implement the motion control modules in the EA library, a remote control system is employed to provide manual teaching to the robot. After trajectories are recorded, a smoothing optimization module is applied to reduce noise from manual demonstrations. For relatively fixed experimental positions, these trajectories can be directly utilized to instantiate the corresponding action modules. Furthermore, a data-driven learning framework that learns from trajectory-following demonstrations is implemented to enable adaptive path-tracking across diverse environments while enhancing robustness and generality. Compared to conventional robot programming, greater flexibility and improved replication of operational expertise are accomplished by this EI teaching and control strategy, which mirrors the proficiency of seasoned experimenters. Although frequently neglected in prior work, there is significant importance with regard to the capability in certain experimental scenarios.

This paper presents a method for demonstration, learning, and optimized control of robotic actions. Supplementary Fig. 2 displays the original trajectories from manual demonstrations of representative Embodied Actions in the experiments and optimized trajectories after processing.

The detailed implementation procedures and algorithms include:

Demonstration

Kinesthetic demonstrations are performed by a human operator physically to guide the master manipulator arms through desired motions while visually monitoring the response of the follower arms. During teaching, the high-precision master arms capture 6-DoF trajectories in real time, synchronized with end-effector signals (pipette plunger position or gripper width). All trajectory data are time-stamped at 200 Hz and stored in ROS bag files, while corresponding commands are streamed to the follower arms at the same frequency. Multiple trials are performed per task type to account for execution-style variations and environmental differences. After collection, trajectories are optimized to extract consistent and reusable motion primitives. The optimized trajectories form the basis of our EA Library, which is a comprehensive repository of motor skills, are capable of flexible combination and adaptation to execute autonomous tasks.

Trajectory Optimization Algorithm

Owing to the inevitable tremor introduced by manual operation, the recorded trajectories are refined by cubic B-spline interpolation. The process involves several steps to ensure that the optimized trajectory is smooth, continuous, and dynamically consistent while preserving the essential motion features captured during the demonstration phase. The implementation functions of the algorithm are located in the file ysmooth.py.

The raw trajectory data are uniformly down-sampled to derive a reduced set of N time stamps $T = \{t_i\}_{i=0}^{N}$ together with the corresponding six-dimensional joint-angle vectors $\boldsymbol{q}_i = [q_{i1}, q_{i2}, q_{i3}, \dots, q_{i6}]^T$. This down-sampling step helps reduce noise and computational complexity without losing critical information about the task execution. Additionally, important key points, such as the start and end points of the trajectory, can be explicitly included in the sampling set to ensure that the interpolated trajectory passes through these critical locations. This guarantees that the optimized trajectory adheres to the fundamental constraints imposed by the task requirements. To achieve smoothness while preserving these constraints, a clamped cubic B-spline is fitted to the sampled data. The clamped cubic B-spline is defined as:

$$\hat{\boldsymbol{q}}(t) = \sum_r \boldsymbol{c}_r B_{r,3}(t),$$

where control points $\boldsymbol{c}_r$ fulfil $\hat{\boldsymbol{q}}(t_i) = \boldsymbol{q}_i$, and endpoint-derivative constraints [1]. The degree-p cubic basis $B_{r,p}(t)$ is generated recursively by the Cox-de Boor formula:

$$B_{r,0}(t) = \begin{cases} 1, t_r \leq t < t_{r+1}, \\ 0, otherwise, \end{cases} p = 0,$$

$$B_{r,p}(t) = \frac{t - t_r}{t_{r+p} - t_r} B_{r,p-1}(t) + \frac{t_{r+p+1} - t}{t_{r+p+1} - t_{r+1}} B_{r+1,p-1}(t), p \geq 1.$$

The smoothed velocity profile $\dot{\hat{\boldsymbol{q}}}(t)$ is obtained analytically from the spline derivative. The above procedure substantially attenuates high-frequency noise while preserving critical motion features, thus providing a continuous, differentiable, and dynamically consistent reference trajectory for downstream controller design. The effect of trajectory optimization is visually demonstrated in Supplementary Fig. 3, where the original noisy trajectory is compared with the refined, smooth trajectory after cubic B-spline interpolation.

Learning-Based Tracking Control Algorithm

For long-range transfers, such as moving from one manipulation site to another, the follower arms are required to track the optimized trajectory with high precision. Classical control techniques, such as Proportional-Integral-Derivative (PID) control, fuzzy control, and sliding mode control, often require complex procedures to determine controller parameters, meaning that the user typically needs some professional knowledge of automatic control. Accordingly, we implement a data-driven control method based on the broad learning system (BLS) that learns from demonstration control strategy $\boldsymbol{\tau}_o$ generated by tracking a known trajectory with a well-tuned controller [2-4]. This control method has several advantages, including a simple structure and not requiring retraining with new demonstration data. Moreover, this method exhibits strong generalization ability and can successfully track the desired trajectory from any starting point. By incorporating Lyapunov theory as a constraint in solving the controller parameters, the trained controller exhibits strong error convergence performance. The implementation functions of the algorithm are located in the file BLScontroller.py. The learning-based control diagram is shown in Supplementary Fig. 3.

Given a task-specific reference trajectory $\{\boldsymbol{\xi}_r, \dot{\boldsymbol{\xi}}_r\}$ and the actual positions and velocities $\{\boldsymbol{\xi}, \dot{\boldsymbol{\xi}}\}$, the difference between the desired state and the actual state of the physical system is defined as the error:

$$\boldsymbol{e} = \boldsymbol{\xi}_r - \boldsymbol{\xi}.$$

In servo control problems, both velocity and position errors should converge to zero. Therefore, we design an appropriate control strategy $\boldsymbol{\tau}$ to reduce system errors. The actual velocity command $\dot{\boldsymbol{\xi}}$ after compensation is given by the following expression

$$\dot{\boldsymbol{\xi}} = \dot{\boldsymbol{\xi}}_r - \boldsymbol{\tau}.$$

Therefore, the position at the next time step can be represented as

$$\boldsymbol{\xi}(t+1) = \boldsymbol{\xi}(t) + \int_t^{t+\Delta t} \dot{\boldsymbol{\xi}}(t)\mathrm{d}t,$$

where Δt is the time difference between the time indices t and $t+1$.

Based on the broad learning system [5], the intermediate layer of the network can be divided into feature nodes $\boldsymbol{Z} = [z_1, z_2, \dots, z_N]$ and enhancement nodes $\boldsymbol{H} = [h_1, h_2, \dots, h_N]$. The weights and biases $\boldsymbol{a}_{ei}$, a_{hj}, b_{ej}, b_{hj} from the input layer to the intermediate layers are initialized randomly. These two types of nodes are calculated as follows

$$z_i = \phi(\boldsymbol{a}_{ei}^T \boldsymbol{e} + b_{ei})$$

$$h_j = \varepsilon\left(a_{hj} z_j + b_{hj}\right).$$

$\phi(\cdot)$ and $\varepsilon(\cdot)$ are continuously differentiable activation functions sharing the same form, as shown below

$$\phi(s_i) = \frac{2}{1 + exp(-2s_i)} - 1,$$

$$\varepsilon(c_j) = \frac{2}{1 + exp(-2c_j)} - 1.$$

Both of these activation functions have the following characteristics

$$\phi(0) = 0, \qquad \dot{\phi}(s_i) > 0, \qquad s_i \neq 0$$

$$\varepsilon(0) = 0,\ \dot{\epsilon}(c_j) > 0,\ c_j \neq 0.$$

The control strategy is formulated in response to the position error $\boldsymbol{e} = \boldsymbol{\xi}_r - \boldsymbol{\xi}$ at each time step, taking the form of

$$\begin{aligned}\boldsymbol{\tau} &= \sum_{i=1}^{N} \boldsymbol{w}_{1i} \phi(\boldsymbol{a}_{ei}^T \boldsymbol{e} + b_{ei}) + \sum_{j=1}^{N} \boldsymbol{w}_{2j} \epsilon\left(a_{hj} z_j + b_{hj}\right) \\ &= \sum_{i=1}^{N} \boldsymbol{w}_{1i} \phi(\boldsymbol{a}_{ei}^T \boldsymbol{e} + b_{ei}) + \sum_{j=1}^{N} \boldsymbol{w}_{2j} \varepsilon\left(a_{hj} \phi\left(\boldsymbol{a}_{ej}^T \boldsymbol{e} + b_{ej}\right) + b_{hj}\right),\end{aligned}$$

where $\boldsymbol{W} = [\boldsymbol{w}_{1i}^T, \boldsymbol{w}_{2j}^T]^T$ are the weights from the intermediate layer to the output layer, N is the number of both the feature and enhancement nodes.

We apply the Lyapunov theory [2] to guarantee the stability of the control system. Specifically, when the continuous and continuously differentiable Lyapunov candidate function satisfies

$$
\begin{aligned}
&V(\boldsymbol{e}, \dot{\boldsymbol{e}}) > 0, \qquad \forall[\boldsymbol{e}, \dot{\boldsymbol{e}}] \in \mathbb{R}^d, \qquad [\boldsymbol{e}, \dot{\boldsymbol{e}}] \neq [\boldsymbol{e}^*, \dot{\boldsymbol{e}^*}], \\
&\dot{V}(\boldsymbol{e}, \dot{\boldsymbol{e}}) < 0, \qquad \forall[\boldsymbol{e}, \dot{\boldsymbol{e}}] \in \mathbb{R}^d, \qquad [\boldsymbol{e}, \dot{\boldsymbol{e}}] \neq [\boldsymbol{e}^*, \dot{\boldsymbol{e}^*}], \\
&V(\boldsymbol{e}^*, \dot{\boldsymbol{e}^*}) = 0, \\
&\lim_{\|\boldsymbol{e}, \dot{\boldsymbol{e}}\| \to \infty} V(\boldsymbol{e}, \dot{\boldsymbol{e}}) = \infty,
\end{aligned}
$$

the system state will globally asymptotically stabilize at $[\boldsymbol{e}^*, \dot{\boldsymbol{e}}^*] = \mathbf{0}$. According to the Lyapunov theory, we derive that the network parameters need to satisfy

$$
\begin{aligned}
&b_{ei} = 0, \\
&b_{hj} = 0, \\
&\boldsymbol{w}_{1i} \boldsymbol{a}_{ei}^T \prec \mathbf{0}. \\
&\boldsymbol{w}_{2j} \boldsymbol{a}_{hj} \boldsymbol{a}_{ej}^T \prec \mathbf{0}.
\end{aligned}
$$

Constraints on the network parameters derived from Lyapunov theory ensure global asymptotic stability of the control system and enable the robot to reliably reach the target position. Training is cast as a constrained optimization problem, minimizing the deviation between the learned control strategy and the demonstration data, that is

$$
\min_{\boldsymbol{W}} \sum_{o=1}^{M} \left\| \sum_{i=1}^{N} \boldsymbol{w}_{1i} \phi(\boldsymbol{a}_{ei}^T \boldsymbol{e} + b_{ei}) + \sum_{j=1}^{N} \boldsymbol{w}_{2j} \varepsilon\big(a_{hj} \phi\big(\boldsymbol{a}_{ej}^T \boldsymbol{e} + b_{ej}\big) + b_{hj}\big) - \boldsymbol{\tau}_o \right\|,
$$

subject to, $i, j = 1, 2, \cdots, N$

$$
\begin{aligned}
&b_{ei} = 0, \\
&b_{hj} = 0, \\
&\boldsymbol{w}_{1i} \boldsymbol{a}_{ei}^T \prec \mathbf{0}, \\
&\boldsymbol{w}_{2j} \boldsymbol{a}_{hj} \boldsymbol{a}_{ej}^T \prec \mathbf{0},
\end{aligned}
$$

where M is the number of demonstration data points. This method obviates laborious manual tuning and streamlines the deployment of conventional control strategies while generalizing effectively to a variety of different trajectories through a single round of parameter learning. Meanwhile, through this method, we achieve superior fidelity in replicating the acquired embodied intelligent actions.

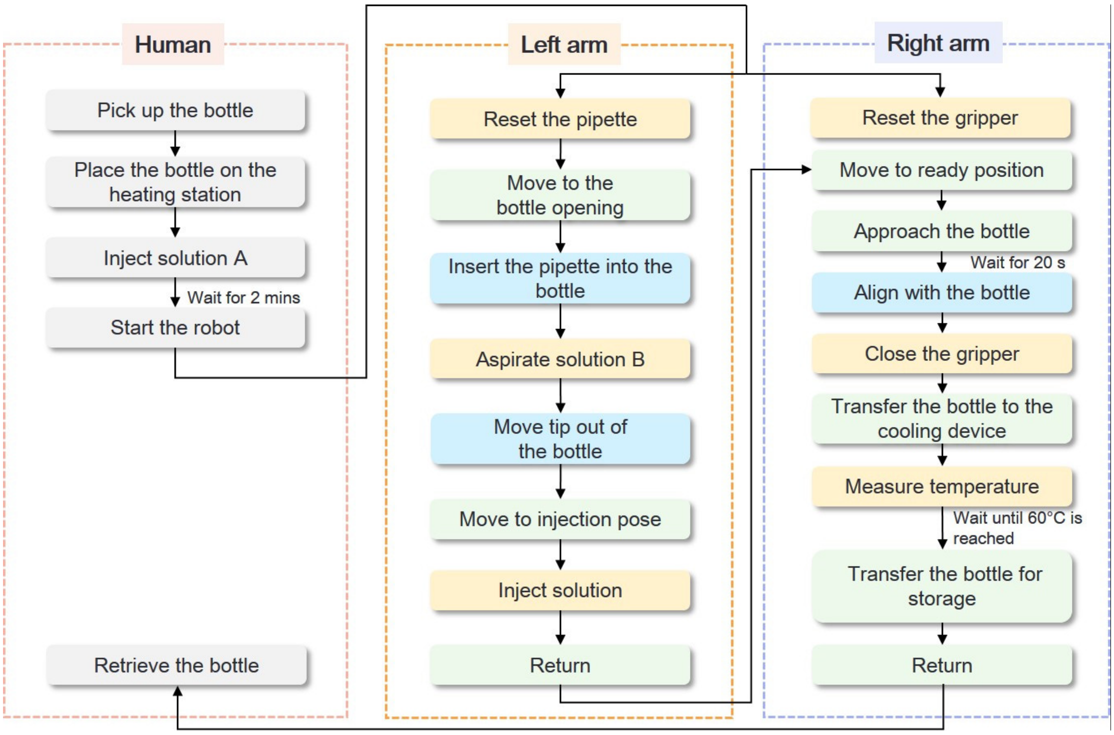
Supplementary Note 2: Precise Control Action (PA) Library

Precise control is critical for experiments, as it directly impacts accuracy and reproducibility. Such control is often infeasible *via* direct manual remote control and thus requires specialized control algorithms or modules. The PA library implemented in this study comprises two categories.

The first category pertains to small-range, high-precision robot arm control during experiments, thus requiring the integration of algorithms such as robot kinematics and visual feedback. For example, when grasping and placing bottles, the system must precisely locate the target and perform vertical movements accurately to ensure insertion into holes with gaps smaller than 0.5 mm. The implementation of this method relies on visual detection and robot feedback control. In order to perform tasks such as grasping bottles at variable positions, YOLO-based bottle detection is utilized to facilitate grasping control. Some visual detection results are shown in Supplementary Fig. 4.

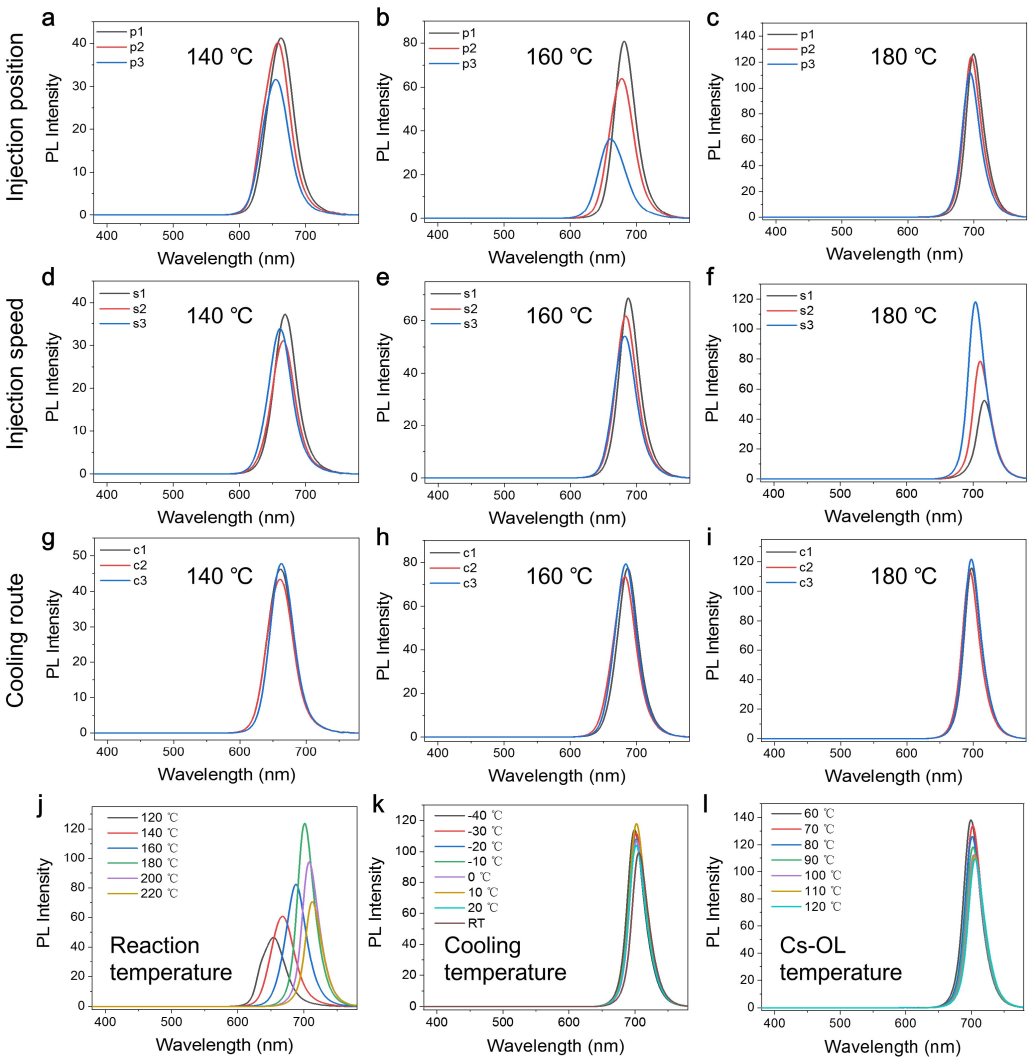
The second category involves the management of specialized execution devices, such as precise liquid dispensing with an electronic pipette. All these operations are encapsulated into dedicated execution commands.

The PA library includes the following key actions: end-effector precise point control, end-effector precise posture control, precise liquid transfer control, and condition-based control using temperature measurements, etc.



Supplementary Fig. 5: Flowchart of experimental steps. Green boxes represent tracking control, blue boxes represent precise control, and yellow boxes represent commands sent to the end effectors.

Supplementary Fig. 5 presents a flowchart illustrating the experimental workflow integrating human collaboration and robotic operations. The process comprises three sections: human operator tasks (preparatory/retrieval work), left-arm tasks (solution injection), and right-arm tasks (cooling). Each section details the sequence of actions performed, with specific color coding to differentiate between tracking control, precise control, and commands sent to the end effectors. Among all actions, movements over longer distances are performed by tracking control (green boxes in Supplementary Fig. 5), while adjustments requiring high positional accuracy are executed by precise control (blue boxes in Supplementary Fig. 5). Commands sent to the end-effectors control the aspiration and injection of the pipette, as well as the opening and closing of the gripper (yellow boxes in Supplementary Fig. 5). During cooling, the gripper concurrently performs temperature measurements until 60 °C is reached. This dual-arm robotic system significantly enhances experimental reproducibility and consistency by executing tasks with high precision and repeatability, minimizing variability from human subjective decision-making inherent in manual operations.



Supplementary Fig. 6: PL spectra of EI-synthesized $CsPbI_3$ QDs under different actions and conditions.

Supplementary Note 3 LLM suggestion for target $CsPbI_3$ QDs

Question for LLMs:

In the table below, the first three columns, Peak position (nm), FWHM (nm), and PL Intensity, are experimental results, while the following six columns, Injection position, Injection speed, Cooling route, Reaction temperature, Cooling temperature, and Cs-OL temperature, are experimental influencing factors. The table is analyzed in depth by adjusting the temperature in order to recommend a new synthesis scheme not present in the table, with requirements of peak = 700 nm, FWHM ≤ 30, and PL intensity ≥ 130.

Gpt4.1 suggested chosen parameters:

Answer:

Analysis

According to the table you provided, **adjusting temperature** (mainly Reaction temperature, Cooling temperature, Cs-OL temperature) has a significant effect on Peak, FWHM, and PL intensity.

Target Parameters
- **Peak position** = 700 nm
- **FWHM** ≤ 30 nm
- **PL Intensity** ≥ 130

Summary of Existing Data Patterns

1. **Peak and Reaction Temperature**
    - As reaction temperature increases from 140→160→180→200→220°C, the peak redshifts (value increases).
    - At 180°C, there are many data points with peaks close to 700 nm (e.g., 699.1, 695.8, 700.9, etc.).

2. **FWHM and Temperature**
    - At 180°C, FWHM drops to around 30 nm, meeting the requirement.

3. **PL Intensity and Temperature**
    - Around 180°C, PL intensity can reach up to 138.1 (699.1, 30.3, 138.1).
    - If reaction temperature is too high (e.g., 200, 220°C), PL drops significantly.

4. **Cooling Temperature and Cs-OL Temperature**
    - As cooling temperature increases, PL intensity drops slightly and peak redshifts slightly.
    - As Cs-OL temperature varies from 60~120°C, PL intensity first increases, then decreases, peaking at 80°C.

5. **Injection position/speed/Cooling route**

- s3, c3, p1 are the most commonly used and effective (smallest FWHM, highest PL, peak closest to target).

Closest Experimental Results to Target
Peak	FWHM	PL Intensity	Reaction T	Cooling T	Cs-OL T
699.1	30.3	138.1	180	0	60
700.9	30	123.8	180	0	80
701.8	30	133.9	180	0	70

But **none simultaneously achieve Peak=700, FWHM≤30, and PL≥130** (PL highest is 138.1 but Peak is 699.1, FWHM is 30.3, slightly above 30).

Recommended New Synthesis Plan

1. Fine-tune Reaction and Cs-OL Temperatures
- In the existing data, **Cs-OL temperature at 60°C yields the highest PL, but peak is slightly below 700 (699.1 nm)**.
- **At Cs-OL temperature of 70°C, Peak=701.8, PL=133.9, FWHM=30**.
- With **Reaction temperature at 180°C, Cs-OL at 70°C, Cooling at 0°C, and p1/s3/c3 combination**, the peak is right at 700-702nm, and PL is above 130.

Plan A (Recommended)
- **Injection position: p1**
- **Injection speed: s3**
- **Cooling route: c3**
- **Reaction temperature: 180°C**
- **Cooling temperature: 0°C**
- **Cs-OL temperature: 65°C** (between 60 and 70°C, theoretically the peak can be adjusted to 700 nm, PL between 133~138)

Plan B (Alternative)
- Slightly **shorten reaction time**, or lower reaction temperature to 178°C (just below 180°C), peak may stabilize at 700 nm, and FWHM stays within 30.

Summary and Suggestions

Recommended New Experimental Formula
Parameter	Recommended Value

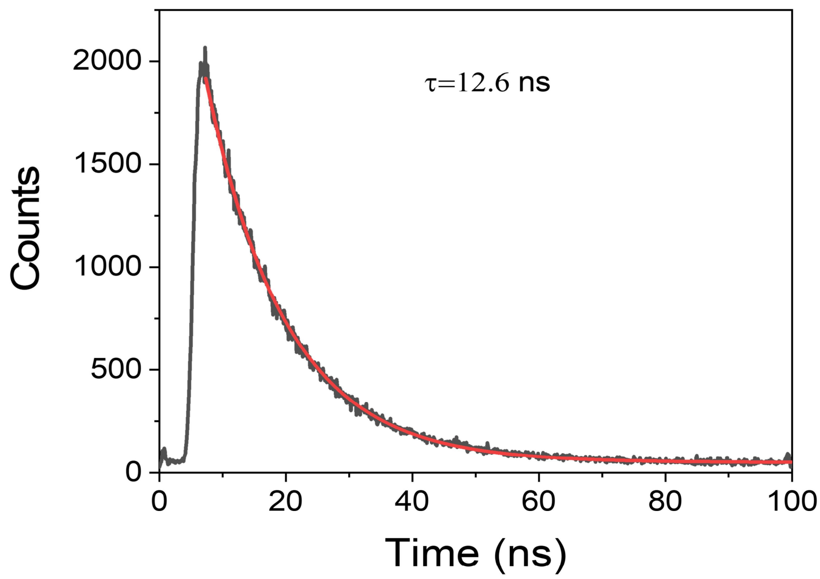
Injection position	p1
Injection speed	s3
Cooling route	c3
Reaction temperature	180°C
Cooling temperature	0°C
Cs-OL temperature	**65°C** (or run several experiments within 62~68°C)

Expected Results
- **Peak position** ≈ 700 nm
- **FWHM** ≤ 30 nm
- **PL Intensity** ≥ 133

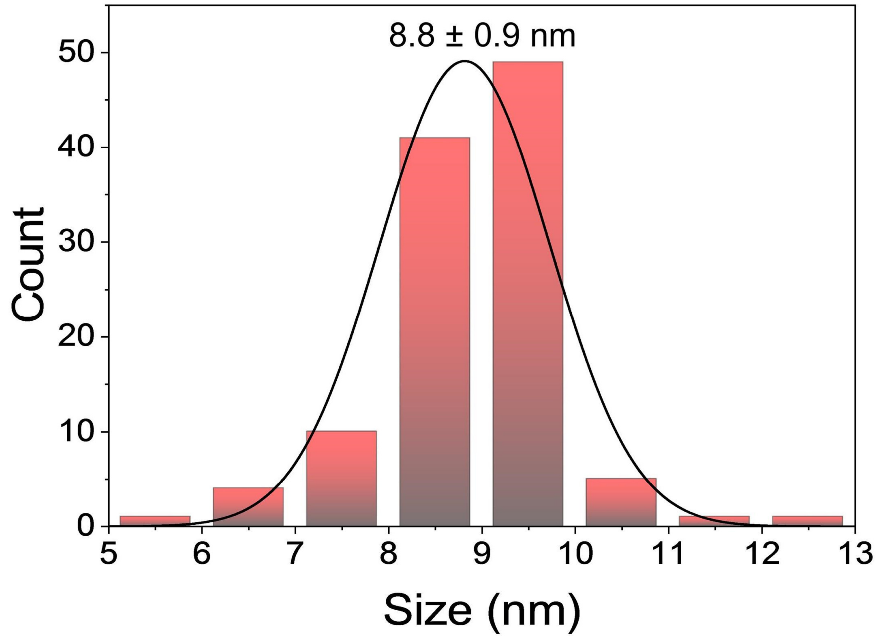
Further Optimization Suggestions
- If peak is slightly below 700, try increasing Cs-OL temperature;
- If FWHM is slightly above 30, try optimizing injection speed or shortening reaction time;
- If PL does not reach 130, try fine-tuning Cs-OL temperature between 60~70°C.

Conclusion:
> **Recommended new plan**: Under the experimental conditions of Injection position p1, Injection speed s3, Cooling route c3, Reaction temperature 180°C, Cooling temperature 0°C, and Cs-OL temperature 65°C, it is expected to achieve Peak=700 nm, FWHM≤30 nm, and PL Intensity≥130.

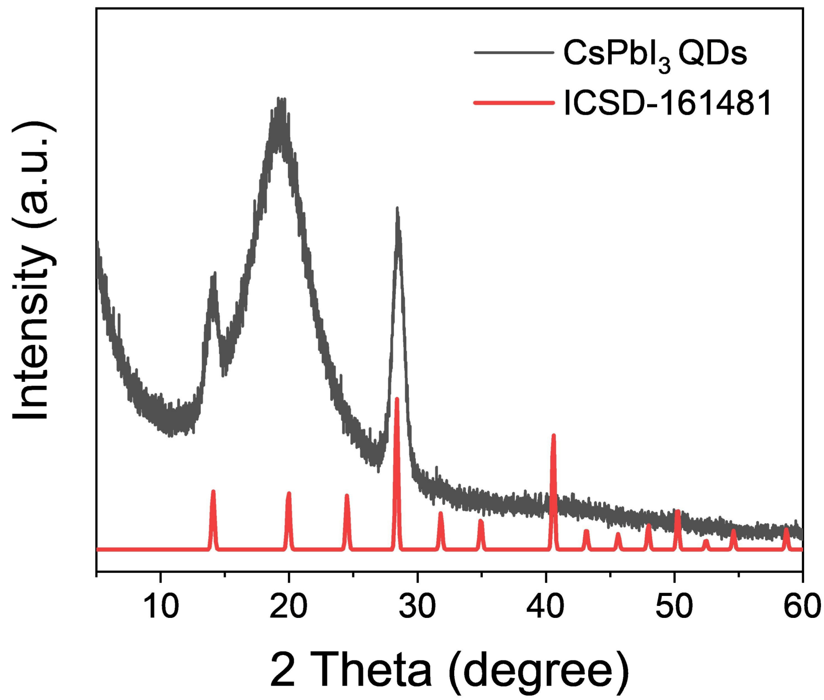
For further fine-tuning, you can experiment with Cs-OL temperature between 60~70°C in 2°C increments, or slightly adjust reaction temperature (178~182°C).



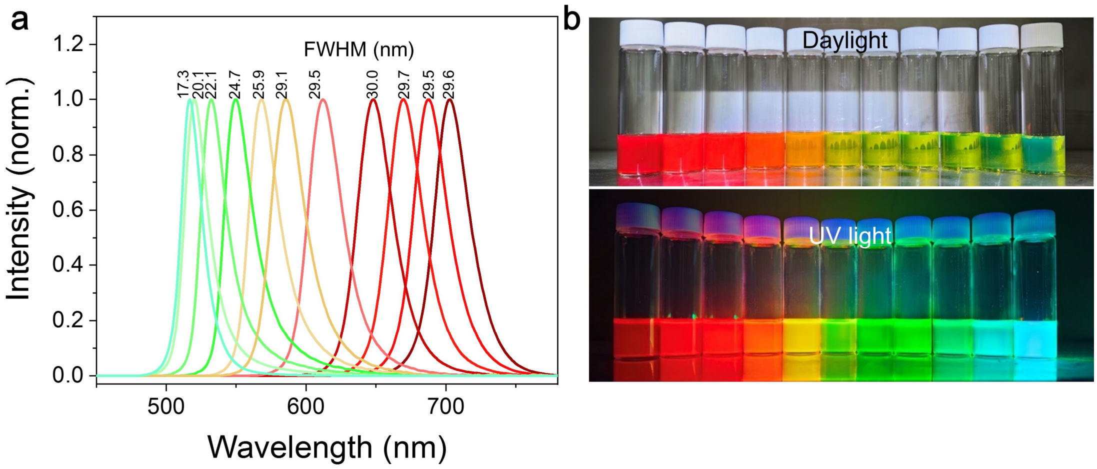
Supplementary Fig. 7: PL lifetime of target $CsPbI_3$ QDs (Single exponential decay).



Supplementary Fig. 8: Size distribution of target $CsPbI_3$ QDs.



Supplementary Fig. 9: XRD pattern of target $CsPbI_3$ QDs.



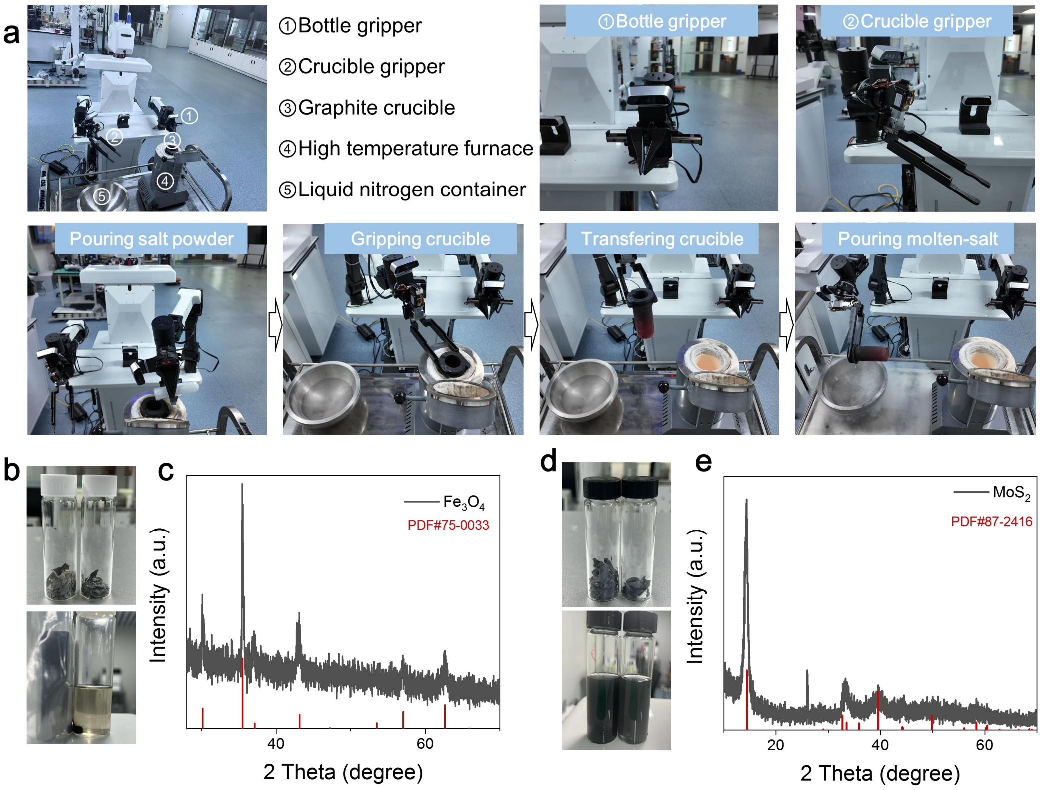
Supplementary Fig. 10: a, PL spectra of EI synthesized perovskite QDs with different halide compositions. b, Photos of colloidal dispersions under daylight and UV light.

Supplementary Note 4: Synthesis of perovskite QDs diversification

To demonstrate precise control over perovskite QDs growth dynamics, the optoelectronic properties are tailored. The EI platform is used to synthesize halogen hybridization perovskite QDs by referring to the synthesis parameters of $CsPbI_3$ QDs. The details are described below. The Cs-OL precursor is prepared as mentioned in the manuscript. The PbX_2 precursor is prepared by dissolving PbX_2 (Table below) in 25 ml of ODE, 2.5 ml of OA, and 2.5 ml of OAm. The reaction temperature is 180 °C. The Cs-OL precursor is kept at 65 °C, and the cooling temperature is 10 °C. The EI actions are p1, s3, and c3. Supplementary Fig. 8 shows the PL spectra of as-synthesized $CsPbBr_xI_{3-x}$ QDs showing peaks from 516 to 702 nm and with a narrow FWHMs from 17.3 to 30.0 nm. The corresponding photo under UV light shows bright photoluminescence from green to red.

PbX_2 precursor prepared with different halide compositions.

	1	2	3	4	5	6	7	8	9	10	11
PbI_2(g)	0.435	0.435	0.435	0.435	0.435	0.435	0.348	0.261	0.174	0.087	0
$PbBr_2$(g)	0	0.069	0.138	0.207	0.276	0.345	0.345	0.345	0.345	0.345	0.345



Supplementary Fig. 11: a, Illustration of the EI platform for extreme-condition synthesis. b, Photos of molten-salt composites and the dispersion of synthesized Fe_3O_4. c, XRD pattern of synthesized Fe_3O_4. d, Photos of molten-salt composites and the dispersion of synthesized MoS_2. e, XRD pattern of synthesized MoS_2.

Supplementary Note 5: Synthesis with EI under extreme conditions

Synthesis of Fe_3O_4 by the molten salt method

6.25 g of LiCl, 7.65 g of KCl, 1 g of $(NH_4)_2MoO_4$, and 1.5 g of K_2S are mixed and ground. The furnace and crucible are preheated to 800 °C, and then the ground powder is placed in the crucible. After 5 minutes, critical manipulations of the extracting crucible from the furnace and pouring the molten salt into liquid nitrogen are programmatically encoded following robotic training. The product is collected after the volatilization of liquid nitrogen.

Synthesis of MoS_2 by the molten-salt method

6.25 g of LiCl, 7.65 g of KCl, and 0.5 g of iron acetylacetonate are mixed and ground. The furnace and crucible are pre-heated to 800 °C before the grounded powder is put into the crucible. After 5 minutes, critical manipulations of extracting the crucible from the furnace and pouring the molten salt into liquid nitrogen are programmatically encoded following robotic training. The product is collected after the volatilization of liquid nitrogen.

Supplementary Table 1: Experimental parameters, peak positions, FWHMs, and PL intensities.

Peak position (nm)	FWHM (nm)	PL Intensity	Injection position	Injection speed	Cooling route	Reaction temperature (°C)	Cooling temperature (°C)	Cs-OL temperature (°C)
661.7	46.4	41.2	p1	s3	c3	140	0	80
657.8	47.8	40	p2	s3	c3	140	0	80
654.8	47.1	31.6	p3	s3	c3	140	0	80
681.8	37.3	80.7	p1	s3	c3	160	0	80
678.8	43.5	63.9	p2	s3	c3	160	0	80
661.1	47.4	36.3	p3	s3	c3	160	0	80
699.1	30.7	126.3	p1	s3	c3	180	0	80
695.8	31.2	123.7	p2	s3	c3	180	0	80
693.8	31.9	111.7	p3	s3	c3	180	0	80
668.9	38.9	37.3	p1	s1	c3	140	0	80
666.8	41	31	p1	s2	c3	140	0	80
661.7	41.2	33.8	p1	s3	c3	140	0	80
685.9	35.5	68.6	p1	s1	c3	160	0	80
682.9	37.8	61.8	p1	s2	c3	160	0	80
682	39.9	54.1	p1	s3	c3	160	0	80
716.2	29.4	118.2	p1	s1	c3	180	0	80
710.2	29.8	78.5	p1	s2	c3	180	0	80
702.8	30	52.4	p1	s3	c3	180	0	80
660.8	46	46.2	p1	s3	c1	140	0	80
661.7	46.4	43.3	p1	s3	c2	140	0	80
662.9	41.9	47.7	p1	s3	c3	140	0	80
687.1	36.9	77.6	p1	s3	c1	160	0	80
682.9	40.3	73.4	p1	s3	c2	160	0	80
683.8	38.7	79.3	p1	s3	c3	160	0	80
697.9	30.3	115.6	p1	s3	c1	180	0	80
694.9	30.9	113.1	p1	s3	c2	180	0	80
696.8	30.5	121.6	p1	s3	c3	180	0	80
653.9	42.9	46.6	p1	s3	c3	120	0	80
666.8	41.2	60.7	p1	s3	c3	140	0	80
687.8	36.4	81.6	p1	s3	c3	160	0	80
700.9	30	123.8	p1	s3	c3	180	0	80
707.9	30.1	97	p1	s3	c3	200	0	80
712.9	29.8	70.5	p1	s3	c3	220	0	80
699.1	30.3	138.1	p1	s3	c3	180	0	60
701.8	30	133.9	p1	s3	c3	180	0	70

700.9	30	125.9	p1	s3	c3	180	0	80
702.8	29.8	118.2	p1	s3	c3	180	0	90
703.5	29.8	112.7	p1	s3	c3	180	0	100
704.9	29.8	11.2	p1	s3	c3	180	0	110
705.8	29.6	109.3	p1	s3	c3	180	0	120
698.8	30	113.5	p1	s3	c3	180	-40	80
701.8	30.3	112	p1	s3	c3	180	-30	80
701.8	30.3	108.1	p1	s3	c3	180	-20	80
700.9	30.3	107.2	p1	s3	c3	180	-10	80
701.8	29.8	108	p1	s3	c3	180	0	80
701.8	30.3	118	p1	s3	c3	180	10	80
701.8	30.5	104.2	p1	s3	c3	180	20	80
705.8	29.6	99.3	p1	s3	c3	180	RT	80

Supplementary Table 2: LLMs recommended experimental parameters and chosen experimental parameters for target $CsPbI_3$ QDs (peak at 700 nm, FWHM ≤ 30 nm, PL intensity ≥ 130).

LLM	Injection position	Injection speed	Cooling route	Reaction temperature (°C)	Cooling temperature (°C)	Cs-OL temperature (°C)
claude3.7	p1	s1	c3	175	0	60
claude3.7-thinking	p1	s1	c3	185	10	55
deepseek-r1	p1	s3	c3	185	-20	60
deepseek-v3	p1	s3	c3	178	0	65
gemini-2.5-pro-exp	p1	s3	c3	182	0	60
gpt4.1	p1	s3	c3	180	0	65
o3	p1	s3	c3	182	0	65
o4-mini	p1	s3	c3	180	0	65
qwen-max	p1	s3	c3	180	0	70
Chosen parameters	**p1**	**s3**	**c3**	**180**	**0**	**65**

Supplementary Table 3: FHWMs of reported $CsPbI_3$ QDs.

Ref. N.	Peak position (nm)	FWHM (nm)	Refs.
1	639	~42	***Nat. Commun.,*** 2024, 15(1): 5696
2	633	36	***Angew. Chem. Int. Ed.*** 2024, 63, e2023187
3	630~680	34~40	***Adv. Funct. Mater.*** 2024, 34, 2405679
4	680~690	~40	***J. Am. Chem. Soc.*** 2019, 141, 2069−2079
5	680	~35	***Angew. Chem. Int. Ed.*** 2020, 59, 22230–22237
6	680	43	***Adv. Funct. Mater.*** 2019, 29, 1902446
7	661	38	***ACS Appl. Mater. Interfaces*** 2022, 14, 17691−1
8	680	~33	***Chem. Mater.*** 2019, 31, 881−889
9	640	40	***J. Phys. Chem. Lett.*** 2018, 9, 4166−4173
10	625~680	~40	***Nano letters***, 2015, 15(6): 3692–3696
11	620~690	~50	***Science***, 2016, 354(6308): 92–95
12	**700.2**	**29.6**	**This work**

References

[1] Carl D. B. On calculating with B-splines. J. Approx. theory **6**, 50-62 (1972).

[2] Xu S., Liu J., Yang C., Wu X. & Xu, T. A learning-based stable servo control strategy using broad learning system applied for microrobotic control. IEEE T. Cybern. **52**, 13727-13737 (2021).

[3] Xu S., Xu T., Li D., Yang C., Huang C. & Wu X. A robot motion learning method using broad learning system verified by small-scale fish-like robot. IEEE T. Cybern. **53**, 6053-6065 (2023).

[4] Xu S., Ou Y., Wang Z., Duan J. & Li H. Learning-based kinematic control using position and velocity errors for robot trajectory tracking. IEEE Trans. Syst. Man Cybern. -Syst. 52, 1100-111 (2020).

[5] Chen C. L. P. & Liu Z. Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. IEEE Trans. Neural Netw. Learn. Syst., 29, 10-24 (2018).