ADVANCED THEORY AND SIMULATIONS
www.advtheorysimul.com

Micro/Nano Motor Navigation and Localization via Deep Reinforcement Learning

*Yuguang Yang, Michael A. Bevan, and Bo Li**

Efficient navigation and precise localization of Brownian micro/nano self-propelled motor particles within complex landscapes could enable future high-tech applications involving for example drug delivery, precision surgery, oil recovery, and environmental remediation. Here, a model-free deep reinforcement learning algorithm based on bio-inspired neural networks is employed to enable different types of micro/nano motors to be continuously controlled to carry out complex navigation and localization tasks. Micro/nano motors with either tunable self-propelling speeds or orientations or both, are found to exhibit strikingly different dynamics. In particular, distinct control strategies are required to achieve effective navigation in free space and obstacle environments, as well as under time constraints. The findings provide fundamental insights into active dynamics of Brownian particles controlled using artificial intelligence and could guide the design of motor and robot control systems to meet diverse application requirements.

1. Introduction

In the past decade, there has been growing interest in engineering active particles for a diverse range of applications.[1–9] Active particles are designed to harvest energy to power translational motion and are envisioned as potential micro-/nano motors to carry out tasks in complex, hard-to-reach environments (e.g., mazes, blood vessels, and porous media). The potential of such motors has been demonstrated in emerging applications like drug delivery, precision surgery, and environmental remediation.[1,3,4,10–16]

Dr. Y. Yang, Prof. B. Li
Institute of Biomechanics and Medical Engineering
Applied Mechanics Laboratory
Department of Engineering Mechanics
Tsinghua University
Beijing 100084, China
E-mail: libome@tsinghua.edu.cn
Dr. Y. Yang, Prof. M. A. Bevan
Chemical & Biomolecular Engineering
Johns Hopkins University
Baltimore, MD 21218, USA

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/adts.202000034

DOI: 10.1002/adts.202000034

The ability of efficient navigation (move from one position to another) and precise localization (maintaining a position) of micro-/nano motors in complex environments plays a crucial role in deploying these motors in applications.[17] Unlike macroscale robots, micro-/nano motors are often under actuated (i.e., not all degrees of freedom can be controlled), and common experimental realizations usually allow individual control on self-propulsion speed (via light, acoustics, etc.[18]) or propulsion direction (via magnetic fields[19]) or speeds and direction combined,[20] but rarely both speed and direction independently. Additional hurdles to reliable control include Brownian motion that can cause significant deviations from intended trajectories. Further considering the rich locomotion dynamics resulting from constituent materials (e.g., metal, polystyrene, etc.[5,18,21]), motor shapes (e.g., spheres,[22] rods,[23,24] and rationally tailored shapes[20]), and activation mechanisms (e.g., chemical catalysis[5] and external fields[25]), it is desirable to have a generic algorithm that addresses underactuation and stochastic disturbances and is broadly suited for different motor designs and control objectives.

Strategies to realize efficient navigation and precise localization include empirical and approximate methods in relatively simple navigation scenarios[26,27] and a more formal algorithmic optimization framework we developed recently that could accommodate complex[28] and even unknown obstacle environments.[29] Particularly, in light of recent fast developments of artificial intelligence and deep learning technologies,[30-32] we recently addressed the navigation challenge in large-scale, unknown obstacle environments via a data-driven visual-based deep reinforcement learning (DRL) algorithm.[29] The DRL algorithm employs a bio-inspired neural network architecture that enables visual navigation based on local neighborhood sensor information and equips the motors with intelligence to efficiently navigate unknown landscapes with random obstacle configurations. Despite its success for binary-activation self-propelled colloidal motors (on and off of self-propulsion), this DRL algorithm can only apply to motors with discrete control inputs, thus failing to meet the requirement of continuous control in applications like high precision localization.

In this work, we develop a flexible, generic DRL algorithm that allows continuous control of motors with different translational and rotational dynamics to carry out localization and

navigation tasks. Leveraging this DRL algorithm, we investigate and compare navigation and localization strategies employed in different scenarios, which ultimately provides guidance for designing future autonomous micro-/nano motor systems. By varying the input information and reward signal structure, we demonstrate its capabilities to navigate in free space and obstacle environments and under additional arrival timing constraints. Our results shed light on DRL-controlled motor dynamics and also provide a new route toward devising motor control systems able to cope with complicated and diverse tasks.

2. Models and Algorithms

In this work, we consider three types of motors, which have the basic locomotion elements among a wide range of motors. The first type of motor considered, which we refer to as a full-control motor hereafter, allows continuous control of its self-propulsion speed and direction. The second type of motor allows continuous control of its self-propulsion direction (e.g., via magnetic field[19]) but not its speed, which we refer to as a rotor motor. The third type of motor allows continuous control of its self-propulsion but not its orientation (e.g., via light[33]), which we refer to as a translator motor.

A full-control motor has the equation of motion given by

$$\begin{aligned}\partial_t \mathbf{r} &= \boldsymbol{\xi}_r(t) + \frac{D_t}{kT}\mathbf{F} + v\mathbf{n}, \quad v \in [0, v_{\max}]\\ \partial_t \theta &= \xi_\theta(t) + w, \quad w \in [-w_{\max}, w_{\max}]\end{aligned} \tag{1}$$

a rotor motor has equation of motion of

$$\begin{aligned}\partial_t \mathbf{r} &= \boldsymbol{\xi}_r(t) + \frac{D_t}{kT}\mathbf{F} + v_{\max}\mathbf{n}\\ \partial_t \theta &= \xi_\theta(t) + w, \quad w \in [-w_{\max}, w_{\max}]\end{aligned} \tag{2}$$

and a translator motor has the equation of motion of

$$\begin{aligned}\partial_t \mathbf{r} &= \boldsymbol{\xi}_r(t) + \frac{D_t}{kT}\mathbf{F} + v\mathbf{n}, \quad v \in [0, v_{\max}]\\ \partial_t \theta &= \xi_\theta(t)\end{aligned} \tag{3}$$

where $\boldsymbol{r} = (x, y)$ and θ denote the position and orientation, respectively, t is time, kT is thermal energy, $\boldsymbol{F}$ is the force due to motor–obstacle electrostatic interactions (see the Experimental Section), v and w are propulsion speed and rotation speed control inputs, respectively. Brownian translational and rotational displacement processes $\boldsymbol{\xi}_r$ and ξ_θ are zero-mean Gaussian noise process with variances $\langle \boldsymbol{\xi}_r(t) \cdot \boldsymbol{\xi}_r(t')^{\mathrm{T}} \rangle = 2D_t\delta(t-t')$ and $\langle \xi_\theta(t)\xi_\theta(t') \rangle = 2D_r\delta(t-t')$, respectively, where D_t is the translational diffusivity and D_r is the rotational diffusivity. All lengths are normalized by particle radius a_R and time is normalized by characteristic Brownian rotational time $\tau = 1 / D_r$. The control update time is $t_c = 0.1\tau$, the integration time step $\Delta t = 0.001\tau$, $v_{\max} = 2\, a_R / t_c$, $D_t = 1.33 a_{R^2} D_r$, and $w_{\max}$ takes different values that will be specified in the following sections.

We formulate the tasks of localization and navigation as sequential decision-making processes in which a motor agent will be rewarded when it is sufficiently close to the specified target location. Formally, we use $s_n = (\mathbf{r}_n, \theta_n)$ to denote the motor's state, where the subscript n is the indexed time step. The motor's observation at s_n, denoted by $\phi(s_n)$, is comprised of a binary image representation of the motor's square neighborhood and the target position ($\mathbf{r}^t$) in the motor's local frame, as shown in **Figure 1**. We represent the motor's decision-making by control policy π, which maps an observation $\phi(s_n)$ to its decisions on self-propulsion and rotation, denoted by a. An optimal control policy π^* that encourages the motor to localize itself around and navigate toward a specified target can be obtained by maximizing the expected reward accumulated during a navigation process, $\mathbb{E}\sum_{n=0}^{\infty}\gamma^n[r(s_{n+1})]$, where r is the one-step reward function and γ is the discount factor to reward rewards in future states. We set $\gamma = 0.99$ to encourage the learning of policies that value rewards coming from distant future. To minimize localization error and arrival time,[28,34] the reward r is set equal to 1 whenever the motors locate within a threshold distance to the target and 0 otherwise (see the Experimental Section for additional details on setting up reward functions).

We use a deep neural network, known as an actor network, to approximate the optimal control policy and another deep neural network, called a critic network, to approximate the optimal state-action value function [Figure 1], which is known as the Q^* function. Q^* function is given by

$$\begin{aligned}Q^*\left(\phi(s), a\right) = \mathbb{E}\big[&r\left(s_1\right) + \gamma^1 r\left(s_2\right) + \gamma^2 r\left(s_3\right)\\ &+ \cdots | \phi(s_0) = \phi(s), a_0 = a, \pi^*\big]\end{aligned} \tag{4}$$

which is the expected sum of rewards along the process by following the optimal policy π^*, after observing ϕ(s) and an initial action a. Both neural networks employ convolution neural layers to process sensory information about the motor's neighborhood, represented by a $W \times W$ binary image ($W = 30$), and a fully connected layer to process the target's position and actions.

We use a DRL algorithm known as deep deterministic gradient descent[35] plus additional enhancements[36,37] to simultaneously train the two networks to approximate their desired target functions. We train the neural network through extensive navigation data in different navigation scenarios with the goal to learn robust navigation strategies (see the Experimental Section for additional details on training neural networks).

3. Results and Discussions

3.1. Free Space Navigation and Localization Dynamics

We first examine the navigation and localization strategies obtained from our DRL algorithm for different types of motors in free space. Before we discuss the specific control policies for each type of motor, we first discuss the high-level mechanism that motors manage to get to targets located at different positions. For both navigation and localization, different motors control either propulsion speed or direction or both such that they can quickly move to specified targets. For targets that are lying in front of motors, the control strategies are relatively straight forward – simply self-propelling toward the targets. When targets are

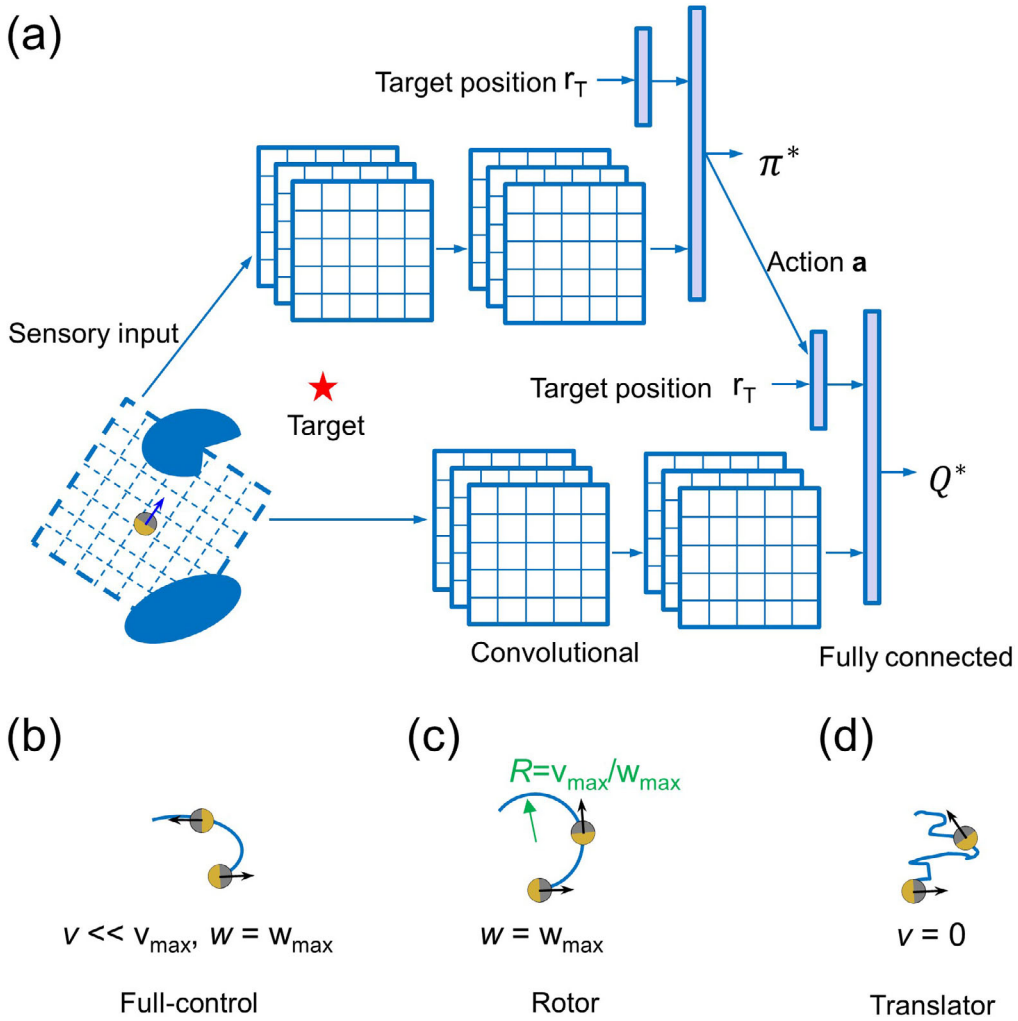

Figure 1. a) The neural network architecture used in our DRL algorithm. The details of the architecture are provided in the Experimental Section. The neural network contains two sub-networks: an actor network and a critic network, where the actor network takes observation as input and outputs actions to adjust self-propulsion speeds and directions, and the critic network takes observation and actions as input and outputs corresponding Q^* value. The observation consists of two streams of sensory inputs, including pixel image ($30a \times 30a$) of the motor's neighborhood fed into convolutional layers and the target's position and actions fed into a fully connected layer. b–d) Different types of motors we consider in this work and their re-orientation strategies.

lying elsewhere, an adjustment on self-propulsion orientation is necessary. Figure 1b–d schematically summarizes the strategies employed by different types of motors to achieve major reorientation. A full-control motor will employ a small propulsion speed and the maximum rotation to re-orient, analogous to steering an automobile [Figure 1b]. A rotor motor will also engage the maximum rotation. Because a rotor is constantly engaged in the maximum self-propulsion ($v = v_{max}$), it will trace out a circular arc of radius $R = v_{max}/w_{max}$ as it engages full rotation speed ($w = w_{max}$) for orientation adjustment [Figure 1c]. A translator motor, due to its inability to directly control orientation, simply turns off self-propulsion and will wait for Brownian motion to sample the desired orientation [Figure 1d]. The typical waiting time for a translator is thus on the scale of the characteristic Brownian rotational time τ.

Figure 2a,b shows the control decisions for a full-control motor on propulsion and rotation speed (normalized by v_{max} and w_{max}). The control decisions are parameterized by the different target locations while the motor is placed at the origin and orients along the x axis. Key aspects of the control strategy are summarized as following: i) If the target is located exactly in front of the motor, ≈zero rotation is applied and self-propulsion is employed, with the amount proportional to the distance up to v_{max}; ii) If the target is located behind the motor, ≈zero self-propulsion is applied but the maximum rotation speed (i.e, -1 and 1) is used to quickly reorient itself; (iii) When the target is located inside a wide vision cone with cone angle ≈± 120°, both rotation and propulsion are engaged, with the amount roughly in proportion to the distance and angle deviation; (iv) Even when the target is lying on the two side and slightly behind (vision cone angle 90–120°), nonzero propulsion is engaged to coordinate with the rotation to achieve the target as soon as possible.

The control strategies for a rotor motor [Figure 2c–e] display similar structures to the rotation decision of the full-control

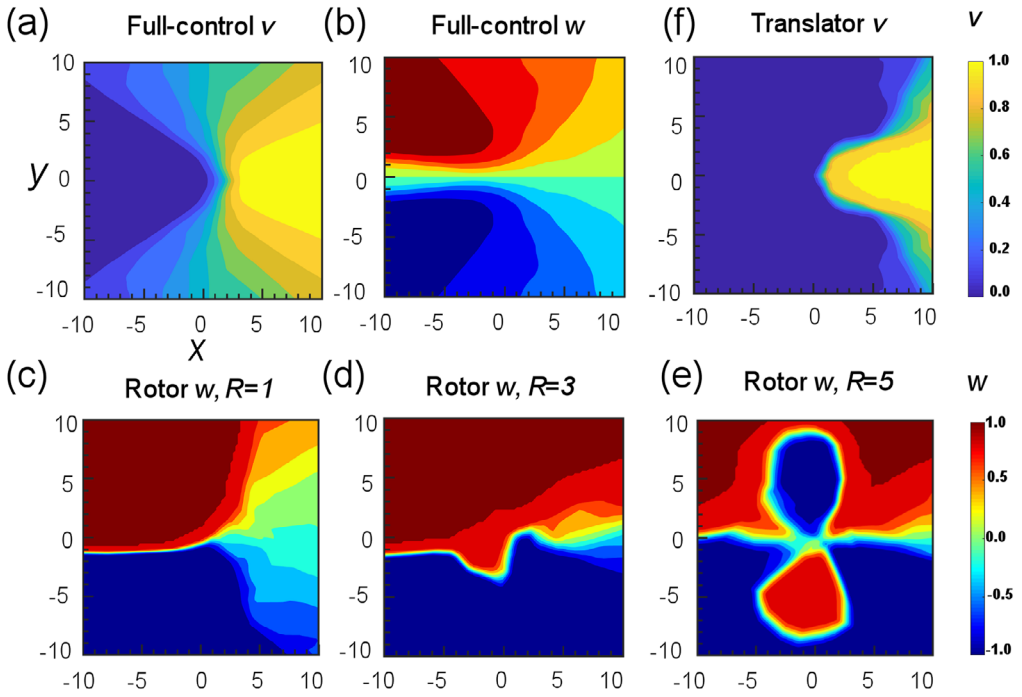

Figure 2. Learned control strategies for different types motors and representative controlled trajectories in free space navigation. In presenting the control policies, we place the motor at (0, 0) with orientation aligning with *x* axis and vary the target location. a,b) Normalized control strategies (normalized by v_{max} and w_{max}) of propulsion speed a) and rotation speed b) as a function of target locations for a full-control motor. c–e) Normalized control strategy of rotation speed for rotor motors with circular radius $R = 1$, $R = 3$, and $R = 5$, respectively, with $R = v_{max}/w_{max}$. f) Normalized control strategy of propulsion speed for a translator motor.

motor but with additional structures depending on the ratio of v_{max} over w_{max}. This ratio determines circular radius R of the trajectory when a rotor motor employs w_{max} for re-orientation [Figure 1c]. When a target is located in the right front, orientation adjustment is unnecessary and thus zero rotation is applied; when a target is located in the right back, the maximum rotation is applied for prompt re-orientation. When the target is lying front but with some angle off, the rotor applies rotation, increasing with the angel deviation, to re-orient itself, but has one critical difference compared to the full-control motor: The rotor usually applies larger rotations in order to quickly re-orient itself since the maximum propulsion speed is always engaged, whereas for the full-control motor, its rotation and propulsion are well coordinated to re-orient and move toward the target.

Additional structures emerge in the control policies [Figure 2c–e] when the target is located near the two sides of the motor with a large R. Because a rotor motor is constantly engaging the maximum self-propulsion, it cannot directly arrive at the target on their near side by simply changing orientations to the side where the target lies. Instead, the rotor will first re-orient to the other direction to temporarily move away from the target, which can be rationalized by the need to gain more room to re-orient. As we increase allowable maximum rotation speed w_{max} (i.e., decreases the circular radius), the control strategy converges to the full-control case.

The optimal control policy of translator motors can be coarsely summarized as orientation timing; that is, self-propulsion is on when the motor favorably orients to the target and off if their orientation is unfavorable. The strength of self-propulsion is approximately proportional to the distance between the target and the motor, up to v_{max}. Similar strategies have been revealed in a number of previous studies.[26,28,29,38,39]

Navigation trajectories of different motors under control steered toward targets at different locations are shown in **Figure 3**a–e. Full-control motors employ a combination of propulsion and rotation strategies, as shown in Figure 2a,b, to realize efficient navigation toward and localization around specified targets [Figure 3a]. Without Brownian motion, trajectories are initialized with re-orientation toward the target if needed and continue with subsequent straight-line movement; with Brownian motion, the rotation will be constantly employed to correct orientation deviations and leads to curved trajectories. After arrival, full-control motors localize themselves by simply applying ≈zero propulsion and rotation in absence of Brownian motion or employ the same strategies in Figure 2a,b to correct deviations from Brownian motion.

The navigation and localization trajectories of rotor motors [Figure 3b–d] display several interesting features compared to full-control motors. When a rotor motor navigates to targets in the back or on the side, its trajectory will trace out arcs with larger radius (i.e., it needs more room to re-orient) compared to full-control motors (see upper panel of Figure 3a–d) due to their constant-on maximum self-propulsion. Rotor motors also display interesting localization behaviors as a result of inability to control its propulsion. Because the propulsion is constantly engaged, after passing through the target, the rotor still needs to constantly adjust its orientation in order to get back to its target. As a result, their trajectories can form regular patterns surrounding the

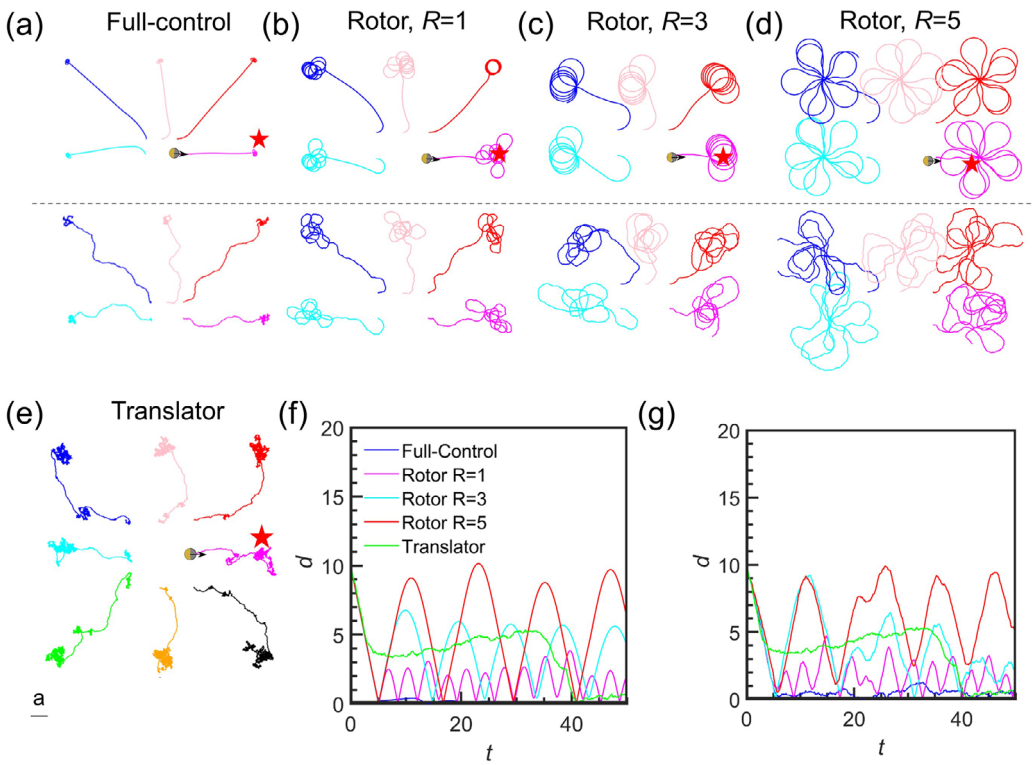

Figure 3. a–e) Navigation and localization trajectories of motors starting at different location but with the same horizontal orientation and toward different targets, denoted by stars (see the Experimental Section for more details on the setup). In (a–d), upper panels (above the dashed line) are trajectories without Brownian translation and rotation for the purpose of illustrating the control policy; lower panels (below the dashed line) are trajectories with Brownian translation and rotation. f,g) Motor-target distance versus time as motors navigate toward and then localize around a target in front of them and at a distance of $10a$. To more clearly illustrate the navigation and localization dynamics, Brownian translation and rotation are not added to full-control and rotor motors in (f), while they are added in (g).

target in the zero-noise limit or irregular ones when there is Brownian motion.

Compared to full-control and rotor motors that can directly control orientation, translator motors rely on Brownian motion to sample favorable directions. Controlled trajectories of translator motors demonstrate an intermittent, non-smooth features since they need to stop and wait for the favorable orientation from Brownian rotation [Figure 3e].

We further compare the navigation and localization dynamics of different motors by examining their distance versus time as they navigate toward and localize around a target in front of them and at a distance of $10a$ [Figure 3f,g]. Full-control and rotor motors can first arrive at the target around 0.5τ (the minimum time possible) as they directly head toward the target at the maximum speed. The translator motors first quickly propel $\approx 5a$ toward the target as their initial orientations are favorably oriented. Then they stay around with no propulsion and wait for favorable orientation to be sampled by Brownian rotation and finally arrive at the target at around 4τ. After arrival, full-control motors can closely localize around the target, with the motor-target distance vanishing in absence of Brownian motion and $\approx 1a$ in presence of Brownian motion. Rotor motors will periodically circulate around the target, with the maximum distance $\approx 2R$. Although translator motors arrive at the target substantially slower than rotor motors, they can stay around the target with a distance of $\approx 1a$ by turning down propulsion strength.

3.2. Free Space Navigation and Localization Performance

We now quantify the navigation performance by comparing the mean traveled distance of motors within given time when they are controlled to transport along a fixed direction. **Figure 4**a,b show their traveled distance versus time within a fixed period of 50τ as they are navigating along the horizon direction. Representative trajectories in Figure 4a show that Brownian motion deviates motors' navigation trajectories toward their horizontal remote target, their propulsion and rotation decision largely maintain themselves near the ideal horizontal transport path. Full-control motors have transport speed $\approx 0.85 v_{\max}$, 15% lower than the ideal navigation speed $v_{\max}$ owing to Brownian motion disturbance. Rotor motors are slightly slower than full-control motors, particularly for $R = 5$ rotor motors that are unable to promptly correct deviations from Brownian motion that slows down navigation.

Translator motors have the worst navigation speed $\approx 0.23 v_{\max}$, which agrees with a theoretical approximate $v_{\max} \int_{-\theta_c}^{\theta_c} \cos(\theta) p^{eq}(\theta) d\theta \approx 0.225 v_{\max}$ where θ is the orientational

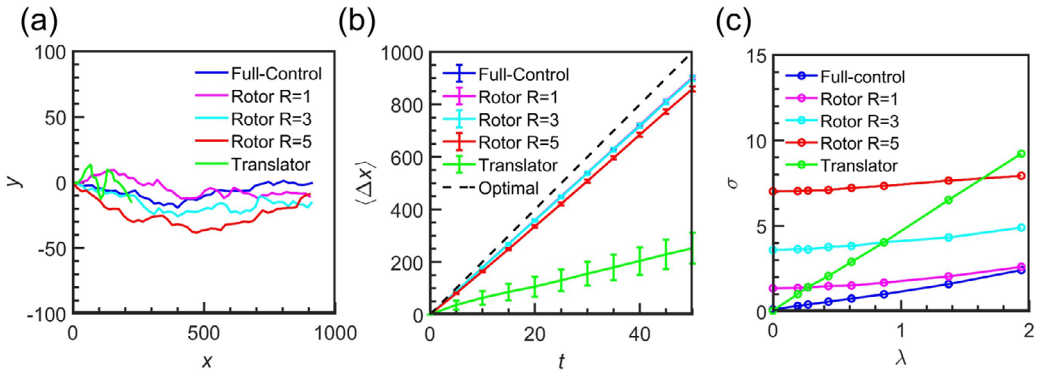

Figure 4. a) Navigation trajectories lasting 50τ of different motors starting from initial state (0, 0, 0) toward a target located at (1000, 0). b) Navigation performance of different types of motors characterized by mean travelled distance versus time along the horizontal distance. Dashed lines are optimal navigation at speed v_{max}. c) Location performance of different types of motors characterized by the steady state deviation σ from target as a function of position perturbation strength λ.

angle, $p^{eq}(\theta) = 1/2\pi$ is the equilibrium distribution of orientational angle, $\theta_c = \pi/4$ is the activation angle estimated from control policy in Figure 2f. Moreover, translator rotors have a substantially larger standard deviation in its travelled distance (see error bars in Figure 4b), indicating its lack of reliability to arrive in time compared to the other two types of motors. The substantial navigation inefficiency of the translator motor is attributed to its reliance on Brownian motion to adjust its orientation to favorable regions. Results in Figure 4b demonstrate that the controllability on self-propulsion direction plays a much more critical role in long-distance navigation than the controllability on self-propulsion speeds.

We further examine the localization performance of motors under various strengths of external disturbance imposed on motors' positions. The localization performance is characterized by the steady-state motor-target distance

$$\sigma = \left\langle \left\| \mathbf{r} - \mathbf{r}_T \right\| \right\rangle \tag{5}$$

where the bracket indicates evaluation using samples drawn from steady state (see the Experimental Section). Increasing the strength of external disturbance on motors' positions is realized by increasing the translation diffusivity D_t in Equation (1)–(3). We characterize the disturbance strength by a non-dimensional parameter λ

$$\lambda = \frac{\sqrt{D_t \Delta t_C}}{v_{max} \Delta t_C} \tag{6}$$

where λ is the ratio of random displacement over self-propulsion distance within one control time step Δt_C. Notably, in estimating σ at various levels of λ, we only increase λ to ≈ 1 as further increasing position disturbance will simply lead to predominantly random walk and the steady state will be unattainable.

As shown in Figure 4c, full-control motors display the best localization performance over the whole range of λ. Although rotor motors have similar navigation performance to full-control motors, their localization performance is the worst among all types of motors at small λ, particularly for rotor motors with large circular radius R. Because of non-controllability on self-propulsion, rotor motors rely on the hovering strategies for localization. Hovering with large circular radius R can cause proportional large deviation to the target [Figure 3f,g]. Finally, a translator motor has an intermediate localization performance at small λ, thanks to its ability to turn off propulsion when not needed.

As we increase λ, localization errors for all types of motors increase, but at different speeds. Particularly, localization errors of translator motors increase linearly and at a much faster speed compared to that of full-control motors. Localization errors of rotor motors increase at relatively slow speeds because they initially have relatively large errors already. At larger $\lambda \approx 1$, the rotor motor starts to outperform translator motor and the performance gap between full-control motor and rotor motor narrows. This is because as the random displacement at one-control step is comparable to the propulsion distance, the localization reduces to a free space navigation and thereby the importance of direction control outweighs the propulsion control, as we concluded from Figure 4b.

In short, results in Figure 4c demonstrate that: i) At $\lambda << 1$, the localization performance is primarily impacted by the controllability on self-propulsion speed; ii) At larger λ, the localization performance is impacted by the controllability on self-propulsion speed and direction, with the latter playing an increasingly predominant role.

3.3. Obstacle Environment Navigation

After understanding the navigation and localization in the free space, we now consider navigation strategies of different types of motors in environments with obstacles. We consider long narrow channel environments where obstacles are placed in the middle and on the side to block the motors [**Figure 5**a–d]. Efficient navigation in the channel requires the motor to circumvent the obstacles in the middle lane and quickly get out of concave traps formed by the obstacles on the side and the walls. The obstacle channels are spacious enough for rotor motor $R = 1$ to gracefully turn around but not enough for rotor motor $R = 5$. We consider the navigation environments with both convex squares and concave crosses obstacles to perform finer examination of navigation capability under different circumstances.

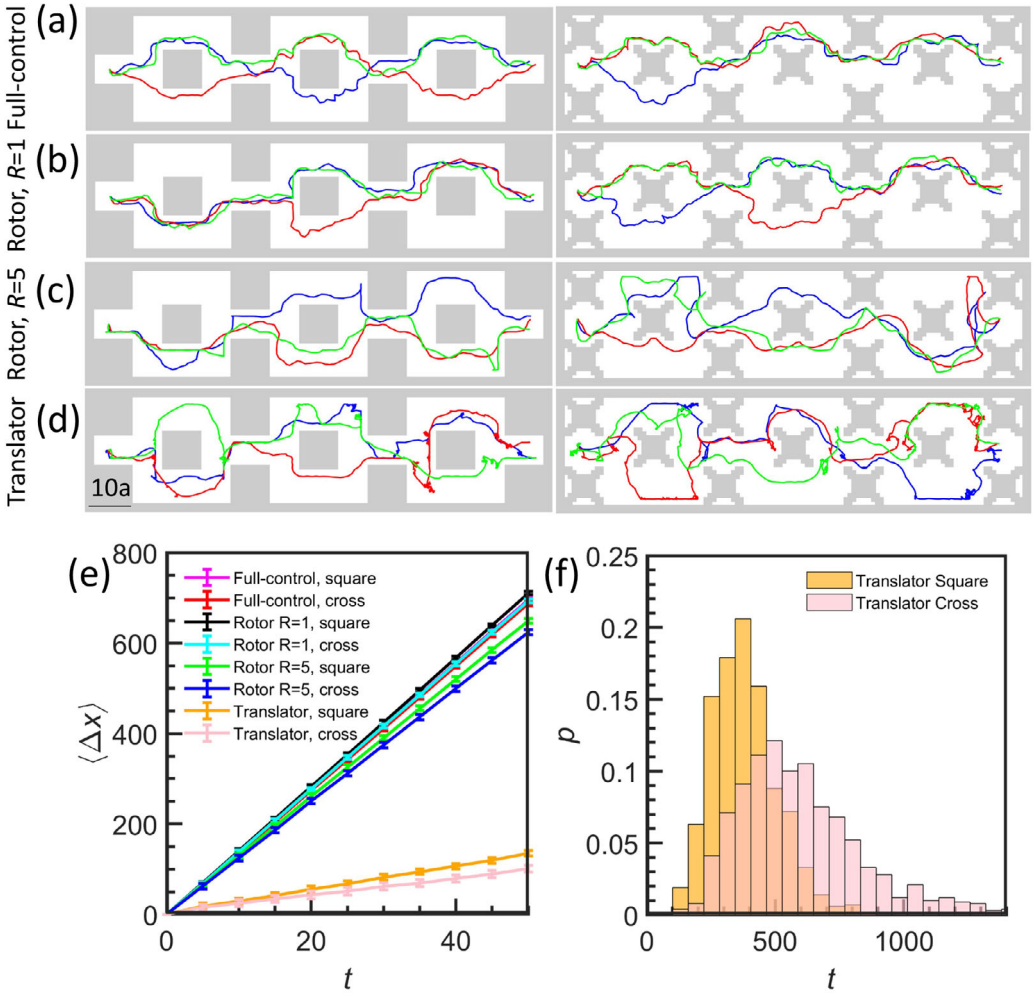

Figure 5. a–d) Controlled trajectories of different motors navigating through channels filled with square (left) and cross obstacles (right). Motors are starting at the left end and navigating toward the right end of the channel. e) Navigation performance in the obstacle channel of different types of motors characterized by the mean travelled distance versus time along the horizontal distance. f) First passage time distribution of a translator motor in the square obstacle channel versus the cross obstacle channel.

Representative controlled navigation trajectories for different motors inside obstacle environments are shown in Figure 5a–d. In both square and cross obstacle channels, full-control motors [Figure 5a] can swiftly control their self-propulsion direction to successfully get around obstacles and avoid getting trapped by concave geometries. The resulting trajectories usually closely follow the boundaries of the obstacles in the middle of the channels to shorten travel path distance for faster arrival. Rotor motors with a small circular radius $R = 1$ [Figure 5b] display similar navigation behavior to full-control motors since they both have the capability of fast adjusting orientation to avoid obstacles and traps. On the other hand, rotor motors with a large circular radius $R = 5$ [Figure 5c] can only adjust direction slowly, and thereby their navigation process usually involves accidentally hitting on the obstacle wall while adjusting the directions. Another downside of the slow direction adjustment is the resulting larger trajectory excursions away from the boundaries of middle lane obstacles, causing delayed arrivals. In addition, cross obstacles can often temporarily trap rotor motors with $R = 5$ since they cannot move away from traps in an agile manner.

Because translator motors have no direct control on propulsion direction, they have to wait (at ≈zero propulsion) for desired orientations sampled from Brownian rotation and then self-propel to circumvent obstacles when favorable directions are sampled. The trajectories of a translator motor [Figure 5d] have similar features to that of a rotor motor with $R = 5$, namely, large deviations from middle lane obstacle boundaries and occasional trapping by cross obstacles.

We perform more quantitative comparison [Figure 5e] on the navigation performance by comparing the horizontal distance travelled versus time in infinitely extended patterned channels like Figure 5a. For all motors, the mean distances traveled versus time are linear, with an average speed $\approx 0.65 v_{max}$, dropping by ≈25% from $\approx 0.85 v_{max}$ in free space navigation. The linearity in travelled distances versus time, instead of leveling off, indicates that controlled motors can all successfully pass through obstacle channels.

Full-control motors and rotor motors with $R = 1$ have similar performances, irrespective of obstacle shapes. Rotor motors with large circular radius $R = 5$ cannot make prompt turns, causing them to get into traps and travel shorter distance (in terms of horizontal distance) within given time. Translator motors display an average navigation speed of $\approx 0.13 v_{max}$, a drop of ≈40% with respect to its free space navigation performance. Compared with the relatively small drop of ≈25% of full-control and rotor motors in presence of obstacles, translator motors' navigation performance is more markedly affected by the presence of obstacles. Another noticeable result in Figure 5e is that the standard deviations of traveled distance for translator motors are much smaller than that in free space navigation due to the confinement effects of obstacles.

We also find that cross obstacles can lead to slower navigation speed for all types of motors. In general, increased proportions of concave features will tend to trap all types of motors, but it only marginally impacts full-control motors and rotor motors with $R = 1$. More remarkable impacts from concave geometry are found for rotor motors with $R = 5$ and translator motors, where the former cannot turn around quickly due to rotation speed limit and the latter cannot directly control direction. Translator motors are affected the most because concave cross obstacles require re-orientation to a larger extent than square obstacles do, and thus require more waiting time for Brownian rotation sampling. We further characterize the effect of concave geometry on navigation of translator motors via first passage time distribution comparison in Figure 5f. Clearly, increasing convex features of obstacles not only can increase the mean first passage time, but also lead to substantial heavier tails in the distribution resulting from the trapping effects.

3.4. Temporal Control

In previous examples, we have investigated the control of motors in space, where motors are navigating toward or localizing around specified spatial targets. The flexibility of our DRL algorithm also allows other control objectives, such as control in the temporal dimension, with minimal modifications on the input and reward function in the algorithm. Here we consider an example of arrival time control objective where we require the motors to arrive at specified locations within a specified time window, neither sooner nor later. Such arrival time control capability could be of potential use for motor applications with timing constraints. For example, in automatically scheduled drug release, drugs are required to be delivered within a restricted time window. More broadly, additional temporal control could enable solutions to problems involving collective dynamics where individuals are precisely controlled to synchronise and coordinate in time (e.g., ants, colloidal swarms).[40-42]

We achieve arrival time control by including time as an observation variable (together with the target location) and provide a time-dependent reward signal that encourages arrivals within the time window but discourages arrivals at other times. For demonstration purpose, we consider motors that navigate to a specified location in free space but requires earliest arrival after $T_c = 5\tau$. We set reward of $r = 1$ for arrivals after T_c and $r = -1$ if arrivals are earlier than T_c, aiming to penalize early arrivals. Because the motor receives discounted rewards (i.e., via $\gamma < 1$), the control policy will be optimized to steer motors to specified target as early as possible after T_c. Similarly, we can also set $r = 1$ when the motor arrives within time window $[T_L, T_R]$ and $r = -1$ elsewhere to enable more precise temporal control objective.

Figure 6a–c shows representative navigation trajectories of different motors from the origin (0, 0) to a target at (20, 20), with arrival time constraints applied. We select such short-ranged target (distance ≈ 28*a*) that motors can mostly arrive earlier than the allowed time T_c. Different motors have learned different navigation strategies that accommodates an arrival time constraint. The full-control motor employs a slow-down strategy that it reduces its self-propulsion speed and slowly arrives at the target at the required time windows [Figure 6a]. The rotor motor cannot control its self-propulsion speed; instead, it will first steer toward the vicinity of the target and then hover around the target as part of postponing its arrival until T_c [Figure 6b]. Translator motors will first engage their full power to get to the vicinity of the target and then wait till T_c, after which they self-propel right away to the target [Figure 6c]. On a higher level, the rotor and translator motors are taking a similar early-arrive-and-wait strategy but implement it according to their specific dynamics. Notably, navigation strategies that satisfy the arrival time constraint are not unique. For example, instead of taking the slow-down strategy, the full-control motor can also first wait somewhere and then employ full power. Our algorithm usually tends to find local optimal solutions that give relatively smooth strategies (in terms of variations of self-propulsion speeds and directions).

To understand the underlying rationale for these adapted decisions, we further quantify the first arrival time statistics [Figure 6d] for motors with constrained versus unconstrained arrival times. Without arrival time constraints, the full-control and the rotor motors arrive around 1.5 τ; with arrival time constraints, full-control and rotor motors arrive around the allowed time T_c. Without arrival time control, the translator motor has a wide heavy-tailed arrival time distribution, with its mode around 3τ. The wide distribution arises from sampling of favorable orientations via Brownian rotation. Statistically, they have a significant chance of arriving very late if not enough favorable orientations are sampled. After adding the constraint, the translator motor's arrival time has a sharp peak at T_c, with a similar tail to the unconstrained case. The formation of peaks for translator motors is the result of the early-arrival-and-wait strategy where large portion of motors take immediate action to arrive at the target near T_c. Notably, the addition of arrival time constraint does not cause a heavy tail after $> 10\tau$, indicating that the constraints only push back early arrivals but do not affect late arrivals in the original unconstrained setting. An interesting aspect on the strategy of translator motor is that the translator motor does not adopt the simple slow-down strategy like the full-control motor. This is because such a slow-down strategy will push back all trajectories and is suboptimal as it delays late arrivals further.

In short, adding arrival time control objectives regulates the learned strategies. This control strategy is the compromised result of the arrival time requirement and the inherent uncontrollable elements of the motor dynamics. In terms of application guidelines, the full control motor and rotator can achieve a good

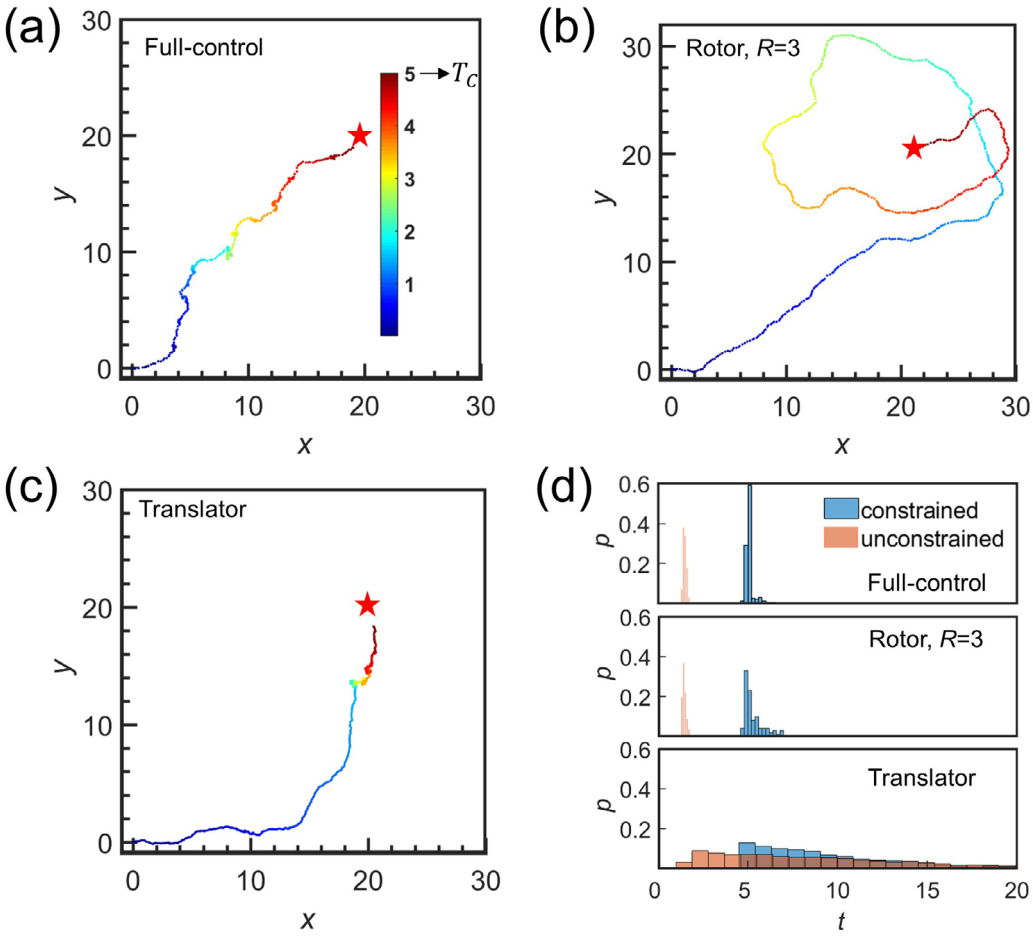

Figure 6. Representative navigation trajectories of motors with arrival time constraint for a full-control motor a), rotor motor with $R = 5$ b), and a translator motor c). Motors are controlled to arrive at the earliest time after $T_c = 5\tau$. d) The histogram of arrival times of controlled navigations with and without arrival time constraint (motors are controlled to arrive as early as possible).

arrival time control, although the rotor will require additional hovering space, which will probably become an issue for applications involving strong confinement.

4. Conclusions and Outlook

We developed a general DRL algorithm that enables continuous control of a broad class of Brownian micro/nano motors in a number of navigation scenarios including free space, obstacle environments, and arrival time constraints. Our DRL can learn competitive strategies solely through navigation data without knowledge of the underlying model. Different motor locomotion dynamics and control objectives have led to different control strategies in free space and obstacle environments. Although it seems a lack of control degree of freedom might significantly impair its functions and performance, our DRL is able to alleviate negative impacts by employing different control strategies in navigation and localization in free space, obstacle environment, and navigation timing control.

Our DRL algorithm is model-free so that it can be used to realize continuous control on micro/nano motors with other dynamic models[1,22,43–47] (e.g., other actuation mechanisms, hydrodynamics, etc.) beyond what has been considered here. Although we have only considered two specific obstacle environments in this work, this type of architecture has been demonstrated to enable motors to learn generalizable navigation strategy in general obstacle environments.[29] Further extension of our DRL includes feeding other visual cues in the neural network such that motors can learn navigation and localization in environments like flow fields[20,48] and 3D porous media. Our DRL algorithm can also be conveniently integrated with experimental systems because of its capability to directly process raw sensor inputs from experimental imaging systems (e.g., microscopes) and other state estimation filter systems to cope with possible experimental measurement and control noises beyond the level of Brownian motion noise considered here.[49] The efficient navigation and localization based on local visual sensor information also suggests that engineering micro/nano motors and robots with local sensing ability[7] is a potential route to intelligent robot systems like macroscale robots. More broadly, our algorithm can also serve as a general-purpose algorithm for diverse continuous control tasks on the microscopic scale, such as controlling colloidal assembly on energy landscapes,[50–53] that challenges the classical Markov

decision process controller[54] because of the curse of dimensionality.

Various comparison studies in this work also provide useful directions to the design of navigation and motor systems for micro/nano robots. The full-control motors have the best performance in all the navigation and localization tests, suggesting that the hardware design of two control degrees of freedom is a direction worth pursuing, despite its considerable challenge. The continuously controllable rotor motors and translator motors are the most accessible experimental designs since only one control degree of freedom is needed. Particularly, if we allow enough maximum rotation speeds and sufficiently fast control frequency, a rotor motor can function comparable to a full-control motor. To realize both advantages of translator and rotor motors with only one control degree of freedom, one can use motors[39] with the switching ability between translator motors and rotor motors. A translator motor is considerably disadvantageous in long-distance navigation and obstacle environment navigation, although they have reasonable localization performance. A potential route for improving the performance of translator motors will be exploiting the swarm intelligence, which can be achieved using multi-agent stochastic feedback control.[42]

5. Experimental Section

Obstacle Representation and Collision Dynamics: Environment maps were directly converted to pixel images (pixel size 1a) using image processing software. Obstacle regions have value 1 whereas free space regions have value 0. The local neighborhood sensory input was obtained by first constructing a squared window of width $W = 30a$ centering on the motor and aligned with its orientation and then extracting a 30 by 30 binary matrix from the environment maps. Same to the previous work,[29] distant targets (target with distance larger than 30a) were projected to a proxy one located on a circle of radius 30a centering on the motor. Target positions were represented in local coordinate system of the motor.

Obstacles on each pixel were represented by repulsive spheres to capture the interaction between the motor and the obstacles, whose interaction force (used in **Equation** (1–3)) were modeled by the electrostatic repulsion, given as,[46,55]

$$\mathbf{F} = \frac{\mathbf{r}_{RO}}{r_{RO}} B^{pp} \kappa \exp\left[-\kappa\left(r_{RO} - 2a\right)\right] \tag{7}$$

where $\boldsymbol{F}$ is the force on the motor, $\mathbf{r}_{RO} = \mathbf{r}_O - \mathbf{r}$ with $\mathbf{r}_O$ being the position of the obstacle, and $r_{RO} = ||\mathbf{r}_{RO}||$. B^{pp} is the pre-factor for electrostatic interactions and κ^{-1} is the Debye length. $B^{pp} = 2.2974a/kT$ and $\kappa^{-1} = 30$ nm were used as parameters.

DRL Algorithm and Training—Actor Network: The neighborhood sensory input first enters a convolutional layer consisting of 32 filters with kernel size 2 × 2, stride 1, and padding of 1, following a batch normalization layer, a rectifier nonlinearity and a 2 × 2 of maximum pooling layer. The output then enters a second convolutional layer consisting of 64 filters and the same kernel, stride and padding as the previous layer, following similarly by a batch normalization layer, a rectifier nonlinearly and a maximum pooling layer. The local target coordinate first enters a fully connected layer consisting of 32 units following by rectifier nonlinearity. Then the output from the target coordinate input and the sensory input will merge and enter a fully connected layer of 64 unit followed by rectifier nonlinearity. The output layer was a fully-connected linear layer with two output of normalized w' and v'. Note that tanh nonlinearity was applied to the output constrain the w' between [-1, 1] and sigmoid nonlinearity was applied to constrain v' between [0, 1]. w' and v' were then multiplied by v_{max} and w_{max} to get the final action output.

Table 1. Training parameters.

Parameter	value
Training episode N_E	≈5000 (full-control, rotor), ≈20 000 (translator)
Minibatch size, B	64
Replay memory size, N_M	500 000
Target network update frequency C	100
Discount factor, γ	0.99
Learning rate, α	0.00025
Soft update parameter, β	0.01
OU process mean level	0
OU process volatility	0.5
OU process mean reversion speed	0.15
Initial target threshold, T_s	0.1
Final target threshold, T_e	1
Target threshold decay, T_d	≈5000 (full-control, rotor), ≈20 000 (translator)
Max step in an episode, *maxStep*	100 (full-control, rotor), 500 (translator)
Sensor window size, W	30

DRL Algorithm and Training—Critic Network: Besides the target and neighborhood sensory input, action outputs from the actor network will also be fed into the critic network. The neighborhood sensory input will pass through the same convolutional layers as in the actor network. The target input will first concatenate with the action output from the actor network. The concatenated vector then will enter a fully connected layer consisting of 32 units followed by rectifier nonlinearity. Then the output from the target coordinate input and the sensory input will merge and enter a fully connected layer of 64 unit following by rectifier nonlinearity. The output layer was a fully-connected linear layer with one output as the Q value given the input of observation and action.

DRL Algorithm and Training—Training Algorithm: The algorithm used to the train the agent was the deep deterministic policy gradient algorithm[35] plus the hindsight experience replay enhancements[29,36] and scheduled multi-stage learning following the idea of curriculum learning.[56] At the beginning of each episode, the initial motor state and the target position were randomly generated in such a way that their distance gradually increases from a small value. More formally, let $D(k)$ denote the maximum distance between the generated initial state position and target position at training episode k, which is given by

$$D(k) = S_m \times (T_e + (T_e - T_s)exp(-k/T_d) \tag{8}$$

where S_m is the maximum of width and height of the training environment (at free space set S_m was set as= 100a), T_s is initial threshold, T_e is the final threshold, and T_d is the threshold decay parameter. Then during the training process, the motor gradually acquires control strategies of increasing difficulties (in terms of initial distance to the target).

During the training process, noises were added to the actions from actor network to enhance the exploration in the policy space. The noise was sampled from an Ornstein–Uhlenbeck (OU) process (on each dimension) given by

$$d\eta = -\alpha(m - \eta)dt + \sigma_{OU}dB_t \tag{9}$$

where α is the reversion parameter, m is the mean level parameter, σ_{OU} is the volatility parameter, and B_t is the standard Brownian motion process.

The complete algorithm is given below. The algorithm parameter is in **Table 1**.

Algorithm: deep deterministic policy gradient with hindsight experience replay

Initialize replay memory M to capacity N_M
Initialize actor network μ with random weight θ^{μ} and critic network Q with random weights θ^{Q}
Initialize target actor network μ' and critic network Q' with random weights $\theta^{\mu'}$ and $\theta^{Q'}$
For episode 1, N_E **do**
Initialize particle state s_0 and target position
Obtain initial observation $\phi(s_1)$
For n = 1, maxStep **do**
Select an action a_n from actor network plus additional perturbation sample from an OU process.
Execute action a_n using simulation and observe new state s_{n+1} and reward $r(s_{n+1})$
Generate observation state $\phi(s_{n+1})$ at state s_{n+1}
Store transition $(\phi(s_n), a_n, r(s_{n+1}), \phi(s_{n+1}))$ in M
Store extra hindsight experience in M every H step
Sample random mini-batch transitions $(\phi(s_j), a_j, r(s_{j+1}), \phi(s_{j+1}))$ of size B from M
Set target value

$$y_j = \begin{cases} r(s_j), \text{ if } s_{j+1} \text{ arrives at the target;} \\ r(s_j) + \gamma Q'(\phi(s_{j+1}), \arg\max_{\nu} Q(\phi(s_{j+1}), \nu)) \text{ otherwise} \end{cases}$$

Perform a gradient descent step on $(y_j - Q(\phi(s_j), a_j))^2$ to update the critic network parameters θ^{Q}
Update the actor network using the sampled policy gradient:

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{B} \sum_i \nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s_i}$$

Update the target networks:

$$\theta^{Q'} = \beta\theta^{Q} + (1-\beta)\theta^{Q'}$$
$$\theta^{\mu'} = \beta\theta^{\mu} + (1-\beta)\theta^{\mu'}$$

End For
End For

Simulation Setup and Performance Evaluation—Free Space Navigation and Localization: For full-control and rotor motors, motors were controlled to navigate to targets with relative coordinates of (10, 0), (10, 10), (0, 10), (-10, 10), and (-10, 0), with and without Brownian motion applied. For translator motors, motors were controlled to navigate to targets with relative coordinates of (10, 0), (10, 10), (0, 10), (-10, 10), (-10, 0), (-10, 10), (0, -10), and (10, -10), with Brownian motion applied. The mean traveled distance versus time for different motors were measured from 100 navigation trajectories starting from an initial state (0, 0, 0) to a target located at (1000, 0). The localization error versus disturbance strength λ was conducted at $\lambda = 0, 0.194, 0.274, 0.434, 0.613, 0.867, 1.171$, and 1.939. The steady state simulations last for 3000τ to collect sufficient data samples. For full-control and translator motors, the control policies were symmetrized such that propulsion speed or rotation decisions for navigating to the left and right were symmetrical or asymmetrical to each other.

Simulation Setup and Performance Evaluation—Navigation in Obstacle Environment: In the square obstacle channel, motors were controlled to navigate from state (14, 5, 0) to a target located at (14, 105). In the cross obstacle channel, motors were controlled to navigate from an initial state (14, 5, 0) to a target located at (14, 115). The mean traveled distance versus time for different motors were measured from 100 navigation trajectories starting from an initial state (14, 5, 0) to a target located at (1000, 0). The first passage time distribution was constructed from 1000 navigation trajectories starting an initial state (14, 5, 0) toward a target located at (14, 105).

Simulation Setup and Performance Evaluation—Navigation with Arrival time Constraint: The first passage time distribution was constructed from 1000 navigation trajectories starting from an initial state (0, 0, 0) toward a target located at (20, 20).

Conflict of Interest

The authors declare no conflict of interest.

Acknowledgements

Financial support from the National Natural Science Foundation of China (11961131005, 11922207) is acknowledged.

Keywords

artificial intelligence, deep reinforcement learning, localization, micromotors, nanomotors

Received: February 19, 2020
Revised: March 29, 2020
Published online:

[1] T. E. Mallouk, A. Sen, *Sci. Am.* **2009**, *300*, 72.
[2] J. Wang, W. Gao, *ACS Nano* **2012**, *6*, 5745.
[3] J. Li, B. E.-F. de Ávila, W. Gao, L. Zhang, J. Wang, *Sci. Robot.* **2017**, *2*, eaam6431.
[4] J. Li, I. Rozen, J. Wang, *ACS Nano* **2016**, *10*, 5619.
[5] S. Sánchez, L. Soler, J. Katuri, *Angew. Chem., Int. Ed.* **2015**, *54*, 1414.
[6] S. J. Ebbens, D. A. Gregory, *Acc. Chem. Res.* **2018**, *51*, 1931.
[7] V. B. Koman, P. Liu, D. Kozawa, A. T. Liu, A. L. Cottrill, Y. Son, J. A. Lebron, M. S. Strano, *Nat. Nanotechnol.* **2018**, *13*, 819.
[8] Z. Wu, L. Li, Y. Yang, P. Hu, Y. Li, S.-Y. Yang, L. V. Wang, W. Gao, *Sci. Rob.* **2019**, *4*, eaax0613.
[9] W. F. Paxton, K. C. Kistler, C. C. Olmeda, A. Sen, S. K. St. Angelo, Y. Cao, T. E. Mallouk, P. E. Lammert, V. H. Crespi, *J. Am. Chem. Soc.* **2004**, *126*, 13424.
[10] L. Soler, V. Magdanz, V. M. Fomin, S. Sanchez, O. G. Schmidt, *ACS Nano* **2013**, *7*, 9611.
[11] C. P. Goodrich, M. P. Brenner, *Proc. Natl. Acad. Sci. USA.* **2017**, *114*, 257.
[12] M. Xiao, X. Guo, M. Cheng, G. Ju, Y. Zhang, F. Shi, *Small* **2014**, *10*, 859.
[13] P. L. Venugopalan, R. Sai, Y. Chandorkar, B. Basu, S. Shivashankar, A. Ghosh, *Nano Lett.* **2014**, *14*, 1968.
[14] K. K. Dey, X. Zhao, B. M. Tansi, W. J. Méndez-Ortiz, U. M. Córdova-Figueroa, R. Golestanian, A. Sen, *Nano Lett.* **2015**, *15*, 8311.
[15] M. Medina-Sánchez, L. Schwarz, A. K. Meyer, F. Hebenstreit, O. G. Schmidt, *Nano Lett.* **2015**, *16*, 555.
[16] M. Yu, L. Xu, F. Tian, Q. Su, N. Zheng, Y. Yang, J. Wang, A. Wang, C. Zhu, S. Guo, X. Zhang, Y. Gan, X. Shi, H. Gao, *Nat. Commun.* **2018**, *9*, 2607.

[17] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield, *Sci. Rob.* **2018**, *3*, eaar7650.
[18] M. Guix, C. C. Mayorga-Martinez, A. Merkoçi, *Chem. Rev.* **2014**, *114*, 6285.
[19] T. R. Kline, W. F. Paxton, T. E. Mallouk, A. Sen, *Angew. Chem., Int. Ed.* **2005**, *44*, 744.
[20] A. M. Brooks, S. Sabrina, K. J. Bishop, *Proc. Natl. Acad. Sci. USA.* **2018**, *115*, E1090.
[21] S. Palagi, P. Fischer, *Nat. Rev. Mater.* **2018**, *3*, 113.
[22] J. R. Howse, R. A. Jones, A. J. Ryan, T. Gough, R. Vafabakhsh, R. Golestanian, *Phys. Rev. Lett.* **2007**, *99*, 048102.
[23] C. Bechinger, R. Di Leonardo, H. Löwen, C. Reichhardt, G. Volpe, G. Volpe, *Rev. Mod. Phys.* **2016**, *88*, 045006.
[24] J. Wang, Y. Yang, M. Yu, G. Hu, Y. Gan, H. Gao, X. Shi, *J. Mech. Phys. Solids* **2018**, *112*, 431.
[25] A. M. Maier, C. Weig, P. Oswald, E. Frey, P. Fischer, T. Liedl, *Nano Lett.* **2016**, *16*, 906.
[26] M. Selmke, U. Khadka, A. P. Bregulla, F. Cichos, H. Yang, *Phys. Chem. Chem. Phys.* **2018**, *20*, 10502.
[27] M. Selmke, U. Khadka, A. P. Bregulla, F. Cichos, H. Yang, *Phys. Chem. Chem. Phys.* **2018**, *20*, 10521.
[28] Y. Yang, M. A. Bevan, *ACS Nano* **2018**, *12*, 10712.
[29] Y. Yang, M. A. Bevan, B. Li, *Adv. Intell. Syst.* **2019**, *0*, 1900106.
[30] Y. LeCun, Y. Bengio, G. Hinton, *Nature* **2015**, *521*, 436.
[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *Nature* **2015**, *518*, 529.
[32] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *Nature* **2016**, *529*, 484.
[33] J. Palacci, S. Sacanna, A. P. Steinberg, D. J. Pine, P. M. Chaikin, *Science* **2013**, *339*, 936.
[34] R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, USA **1998**.
[35] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, *arXiv preprint* **2015**, arXiv:1509.02971.
[36] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, W. Zaremba, *Adv. Neural Info. Process. Syst.* **2017**, *31*, 5048.
[37] H. Van Hasselt, A. Guez, D. Silver, in *Thirtieth AAAI Conf. on Artificial Intelligence*, Phoenix, Arizona **2016**.
[38] D. F. B. Haeufle, T. Bäuerle, J. Steiner, L. Bremicker, S. Schmitt, C. Bechinger, *Phys. Rev. E* **2016**, *94*, 012617.
[39] T. Mano, J.-B. Delfau, J. Iwasawa, M. Sano, *Proc. Natl. Acad. Sci. USA* **2017**, *114*, E2580.
[40] O. Feinerman, I. Pinkoviezky, A. Gelblum, E. Fonio, N. S. Gov, *Nat. Phys.* **2018**, *14*, 683.
[41] J. E. Ron, I. Pinkoviezky, E. Fonio, O. Feinerman, N. S. Gov, *PLoS Comput. Biol.* **2018**, *14*, e1006068.
[42] Y. Yang, M. A. Bevan, *Sci. Adv.* **2020**, *6*, eaay7679.
[43] J. L. Bitter, Y. Yang, G. Duncan, H. Fairbrother, M. A. Bevan, *Langmuir* **2017**, *33*, 9034.
[44] Y. Liu, Y. Yang, B. Li, X. Q. Feng, *Soft Matter* **2019**, *15*, 2999.
[45] L. van der Maaten, G. Hinton, *J. Mach. Learn. Res.* **2008**, *9*, 2579.
[46] Y. Yang, M. A. Bevan, *J. Chem. Phys.* **2017**, *147*, 054902.
[47] X. Shi, F. Tian, *Adv. Theory Simul.* **2019**, *2*, 1800105.
[48] L. Biferale, F. Bonaccorso, M. Buzzicotti, P. Clark Di Leoni, K. Gustavsson, *Chaos* **2019**, *29*, 103138.
[49] Y. Yang, Ph.D., Johns Hopkins University (Baltimore, MD) **2017**.
[50] M. A. Bevan, D. M. Ford, M. A. Grover, B. Shapiro, D. Maroudas, Y. Yang, R. Thyagarajan, X. Tang, R. M. Sehgal, *J. Process Control* **2015**, *27*, 64.
[51] T. D. Edwards, Y. Yang, D. J. Beltran-Villegas, M. A. Bevan, *Sci. Rep.* **2014**, *4*, 6132.
[52] X. Tang, B. Rupp, Y. Yang, T. D. Edwards, M. A. Grover, M. A. Bevan, *ACS Nano* **2016**, *10*, 6791.
[53] Y. Yang, R. Thyagarajan, D. M. Ford, M. A. Bevan, *J. Chem. Phys.* **2016**, *144*, 204904.
[54] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, Wiley, Hoboken, NJ **2014**.
[55] Y. Yang, B. Li, *J. Chem. Phys.* **2019**, *151*, 164901.
[56] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, P. Abbeel, *arXiv preprint* **2017**, arXiv:1707.05300.