# Dynamic Adaptation Using Deep Reinforcement Learning for Digital Microfluidic Biochips

TUNG-CHE LIANG, NVIDIA Corporation, USA
YI-CHEN CHANG, Duke University, USA
ZHANWEI ZHONG, Marvell Technology Inc., USA
YAAS BIGDELI, Duke University, USA
TSUNG-YI HO, National Tsing Hua University, Taiwan
KRISHNENDU CHAKRABARTY and RICHARD FAIR, Duke University, USA

We describe an exciting new application domain for deep reinforcement learning (RL): droplet routing on digital microfluidic biochips (DMFBs). A DMFB consists of a two-dimensional electrode array, and it manipulates droplets of liquid to automatically execute biochemical protocols for clinical chemistry. However, a major problem with DMFBs is that electrodes can degrade over time. The transportation of droplet transportation over these degraded electrodes can fail, thereby adversely impacting the integrity of the bioassay outcome. We demonstrated that the formulation of droplet transportation as an RL problem enables the training of deep neural network policies that can adapt to the underlying health conditions of electrodes and ensure reliable fluidic operations. We describe an RL-based droplet routing solution that can be used for various sizes of DMFBs. We highlight the reliable execution of an epigenetic bioassay with the RL droplet router on a fabricated DMFB. We show that the use of the RL approach on a simple micro-computer (Raspberry Pi 4) leads to acceptable performance for time-critical bioassays. We present a simulation environment based on the OpenAI Gym Interface for RL-guided droplet routing problems on DMFBs. We present results on our study of electrode degradation using fabricated DMFBs. The study supports the degradation model used in the simulator.

CCS Concepts: • **Hardware → Analysis and design of emerging devices and systems**; • **Computer systems organization → Embedded systems;**

Additional Key Words and Phrases: Biochips, Biological system modeling, Real-time systems, Reinforcement learning

This research was supported in part by the National Science Foundation under grants CCF-1702596, ECCS-1914796, and ECCS-2313498.

A preliminary version of this article appeared in *Proceedings of the International Conference on Machine Learning* (ICML), 2020 [42].

The work was finished when T.-C. Liang and Z. Zhong were Ph.D. students at Duke Univesity.

Authors' addresses: T.-C. Liang, NVIDIA Corporation, Santa Clara, CA 95051; e-mail: tliang@nvidia.com; Y.-C. Chang, Y. Bigdeli, K. Chakrabarty, and R. Fair, Duke Universiy, Durham, NC 27708; e-mails: {yichen.chang, yaas.bigdeli, krish, rfair}@duke.edu; Z. Zhong, Marvell Technology Inc., Santa Clara, CA 95054; e-mail: zzhong@marvell.com; T.-Y. Ho, National Tsing Hua University, Hsinchu, Taiwan, 30013; e-mail: tyho@cs.nthu.edu.tw.



1084-4309/2024/01-ART24 $15.00
https://doi.org/10.1145/3633458

**ACM Reference format:**
Tung-Che Liang, Yi-Chen Chang, Zhanwei Zhong, Yaas Bigdeli, Tsung-Yi Ho, Krishnendu Chakrabarty, and Richard Fair. 2024. Dynamic Adaptation Using Deep Reinforcement Learning for Digital Microfluidic Biochips. *ACM Trans. Des. Autom. Electron. Syst.* 29, 2, Article 24 (January 2024), 24 pages.
https://doi.org/10.1145/3633458

## 1 INTRODUCTION

In recent years, we have seen progress on the use of deep **Reinforcement Learning (RL)** to assist sequential decision-making problems, such as games [2, 67, 77], robotics [18], autonomous driving [57, 61, 78], quantitative trading strategies [37], and healthcare systems [46]. The systems assisted by RL have shown tremendous promise in games [50, 67], robotics [18], and natural language processing [22, 51]. This can be attributed to the fact that RL systems in dynamic environments can learn from history and adapt better to the environment. In this article, we show that because the health of an electrode in a **Digital Microfluidic Biochip (DMFB)** dynamically changes over time, we can utilize innovations in RL to ensure more reliable droplet transportation in DMFBs.

### 1.1 Digital Microfluidic Biochips

The rapid worldwide spread and impact of the COVID-19 virus has created an urgent need for reliable, accurate, and affordable testing on a massive scale. For example, the National Institutes of Health (NIH) has launched the Rapid Acceleration of Diagnostics (RADx) initiative to develop and implement technologies for COVID-19 testing [54]. One of the most promising technologies for realizing this goal is digital microfluidics. A microfluidic biochip (DMFB) manipulates tiny amounts of fluids to automatically execute biochemical protocols for point-of-care clinical diagnosis with high efficiency and fast sample-to-result turnaround [16, 65, 74]. Because of these characteristics, the RADx initiative has awarded grants to several biomedical diagnostic companies to develop microfluidic technologies that could dramatically increase testing capacity and throughput [53, 56]. Other applications of DMFBs include screening of newborn infants [33, 69], drug discovery [38], and clinical diagnostics [8, 62].

A DMFB consists of an electrode array in two dimensions that controls the movement of discrete liquid droplets. Upon actuation by a sequence of control voltages, the electrode array can perform a variety of fluidic operations, such as dispensing, mixing, and splitting [7, 25]. Figure 1(a) shows a DMFB in which two droplets are present on a patterned electrode array. Nanoliter droplets on this platform are transported using the principle of **Electrowetting-on-Dielectric (EWOD)** [60]. This principle refers to the modulation of the interfacial tension between a conductive fluid and a solid electrode coated with a dielectric layer through the application of an electric field between them. See Figure 1(b).

Illumina commercialized digital microfluidics for sample preparation in 2015 through NeoPrep—a nearly $40K instrument that automates the preparation of up to 16 sequencing libraries at a time [31]. Genmark has also deployed the microfluidic technology for infectious disease testing [59], and Baebies uses this technology to detect lysosomal storage diseases in newborns [26].

However, reliability remains a major concern in DMFB systems. Illumina halted the sale of NeoPrep in February 2017. In its letter to customers, Illumina cited reliability issues in-house and far worse ones in the field. Even though biochips are tested after production, defects such as electrode degradation can occur during system lifetime [13, 71]. As the electrodes are actuated over time, two types of electrode degradation might happen: charge residual and charge trapping. Charge residual is caused by the accumulated charges, which can be mitigated by inserting grounding vectors [58]. Charge trapping is when the charges are trapped in the dielectric insulator, and this

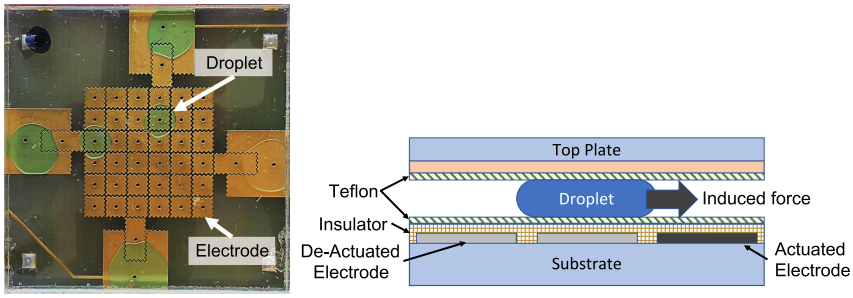



Fig. 1. (a) Top view of a DMFB. Two droplets are present on the biochip. (b) Illustration of the side view of a DMFB. The droplet is moved to the right using EWOD.

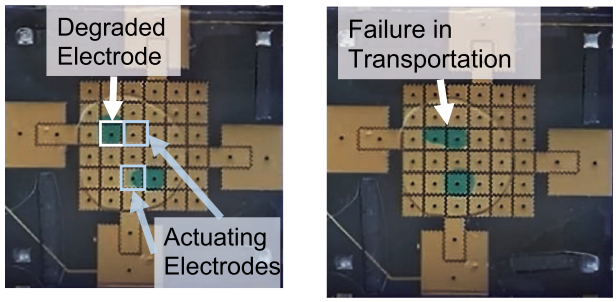



Fig. 2. Droplet transportation failure due to electrode degradation. (a) Two droplets on the electrode array. Two electrodes are actuated to move these droplets. (b) After electrode actuation, the upper droplet cannot be moved completely because it was present over a degraded electrode; the lower droplet is correctly moved to the desired electrode.

phenomenon is irreversible [5]. A consequence of electrode degradation is that droplet movement is impeded [72]. An example of electrode degradation is shown in Figure 2. The figure shows two droplets on the biochip—one located on a degraded electrode. Two electrodes are actuated to move these droplets. However, one of these operations fails because the degraded electrode exerts additional surface-tension force. Detailed analyses of the relationship between electrode defects and fluidic operations can be found in the work of Drygiannakis et al. [14].

### 1.2 Motivating RL-Guided Droplet Routing

In a typical use model for DMFBs [70], a bioassay protocol with fluidic operations is obtained from biologists. Next, a synthesis technique maps these operations to groups of electrodes, referred to as fluidic modules, of a biochip to perform the required operations [4]. A droplet has to be transported from one module to the next. The problem of determining droplet transportation paths between modules is referred to as *droplet routing*. A number of droplet routing techniques have been proposed in the literature for bioassay applications [73, 82, 86]. Su et. al [73] proposed the first systematic droplet routing approach, which adopted the Lee algorithm and minimized the number of electrodes used for droplet routing. Xu and Chakrabarty [82] proposed a droplet routing aware synthesis tool, which was based on parallel recombinative simulated annealing. Zhao and Chakrabarty [86] proposed an integer linear programming based method to co-optimize droplet routing and pin mapping.

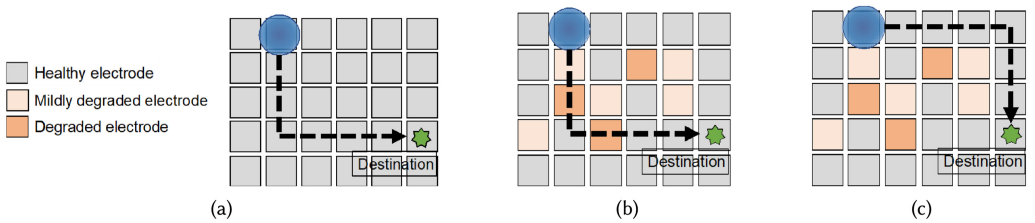



Fig. 3. Droplet routing paths from a start point to an end point (gray: healthy electrodes; brown: degraded electrodes). (a) A pre-computed path for a healthy DMFB. (b) The DMFB has aged, and some electrodes have degraded. Some degraded electrodes are involved in the pre-computed path. Droplet transportation may hence fail. (c) A more reliable path for the aged DMFB.

However, these methods overlook the fact that transportation of droplet may fail if the electrodes on the routing path degrade over time.

***Example.*** Figure 3(a) shows a pre-computed routing path. We can see that this route is the shortest path between the start and the destination points. Droplet transportation can be successful because the biochip is healthy (i.e., no electrode degradation has occurred). Conversely, Figure 3(b) shows that droplet transportation to the destination fails because degraded electrodes exist in the associated path. If an online droplet router knows the locations of the degraded electrodes, it can generate another route that involves only healthy electrodes. An alternative route is shown in Figure 3(c); note that this is a shortest path, and it avoids electrodes that are degraded.

In Figure 3, a different color is used to indicate the degraded electrodes. However, in reality, we cannot identify degraded electrodes by simple examination; this is because the degradation process results from charge trapped in the insulator. When routing errors occur, simply replacing the degraded DMFB with a new one will not only increase the cost but also lead to undesirable wastage of biosamples. Droplets that are in the middle of an unfinished operation, such as mixing or diluting, need to be abandoned. The wastage of droplets is particularly undesired in some applications, such as newborn screening [32] and forensic analysis [83], since the bio-examples are limited in volume and availability. For example, in the newborn screening test provided by Baebies Inc., the entire screening test contains 10 to 20 different assays and each assay needs 100 nl of dried blood spot extract [32]. Thus, a newborn screening test needs at least 1,000 nl of dried blood spot extract, which needs 200 to 300 $\mu$ L (4–6 drops) of whole blood [3]. Prior work has led to synthesis methods that prevent excessive usage of a few electrodes by evenly distributing fluidic operations to multiple electrodes [5, 88]. However, these methods can only postpone the occurrence of electrode degradation, which still happens as electrodes are actuated over time. If such electrode degradation happens during bioassay execution and a route is associated with degraded electrodes, bioassay execution will fail, and it will need to be re-executed on a new biochip [29]. Furthermore, the locations of degraded electrodes may vary from biochip to biochip because the electrode degradation process is affected by geometric variations and different electrode actuation times [24].

Several methods have been proposed to perform error recovery when routing tasks fail [1, 40, 66]. However, these methods are focused on recovery after routing failures, and they do not proactively alleviate the occurrence of erroneous behaviors caused by electrode degradation. Recently, an RL-based routing framework was developed to identify degradation-aware routing strategies for **Micro-Electrode-Dot-Array (MEDA)** biochips [15]. However, this method cannot be used for DMFBs due to the inherent difference between DMFBs and MEDA biochips: MEDA biochips provide the real-time degradation status of each electrode using built-in sensing circuits. This is not the case of non-MEDA DMFBs. In this work, we adopt RL techniques to respond to the dynamic degradation environments, which is not possible with existing offline routing methods.

Numerous papers have been published in recent years to advance applications that leverage RL theory [9, 11, 49]. Our work aims to introduce RL to a new application—that is, the droplet routing problem on DMFBs. We target an RL formulation for the droplet routing problem to address the dynamic degradation of electrodes. An RL-based droplet router addresses the electrode degradation problem and ensures reliable bioassay executions in three ways. First, it provides real-time decision for droplet routing. Second, it can "learn" from the prior experience associated with electrodes that start malfunctioning. Therefore, the droplet router can generate routing paths that include any healthy electrodes. Third, even though the degradation processes may differ for two DMFBs, the router can generate different, yet reliable, routing paths on distinct DMFBs for the same routing objective.

### 1.3 Article Contributions

This article represents one of the first attempts to map RL to clinical microfluidic systems. The main contributions of this work are as follows:

- We describe a new framework for RL-based droplet routing on DMFBs. We discuss the challenges inherent in formulating droplet routing as an RL task.
- We describe an experiment using fabricated PCB-based DMFBs to gain insights into electrode degradation. The insights derived in this manner support our degradation model in the simulator.
- We present an online droplet routing framework, which uses deep RL to generate a policy that uses real-time observations of a DMFB to choose droplet paths dynamically. Training is first carried out in a simulated DMFB. Next, the pre-trained policy is loaded on the controller for the DMFB, and the policy generates routing paths in a real-time manner.
- We consider a parallel droplet routing scenario where multiple droplets are transported concurrently on a DMFB. We formulate a **Multi-Agent Reinforcement Learning (MARL)** framework for parallel droplet routing on DMFBs. Experimental results show that the MARL framework outperforms the single-agent RL framework in parallel routing scenarios.
- We evaluate the proposed solution by executing an epigenetic bio-protocol on a fabricated DMFB. Our experiment shows that the online router can learn the degradation behavior of electrodes and generate reliable routes.
- We identify the timing constraints associated with the use of the RL approach on a simple, GPU-less micro-computer (Raspberry Pi 4). The results show that the timing constraints arising from the RL approach do not impede the fluidic operations in a bioassay.

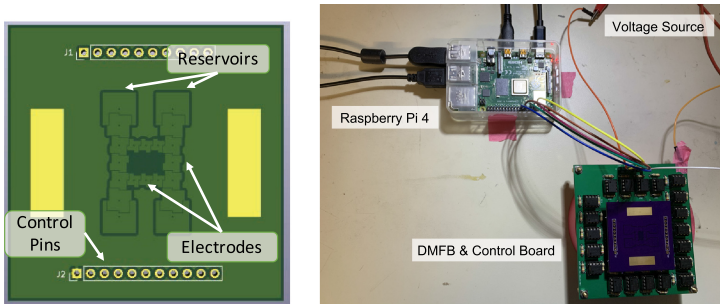
## 2 PROBLEM FORMULATION

The problem formulation for the droplet routing problem on DMFBs is as follows.

Given a DMFB consists of a two-dimensional array of electrodes with size $N \times M$, let $e_{i,j}$ represent the $i^{\text{th}}$ row and the $j^{\text{th}}$ column electrode of the DMFB, where $1 \leq i \leq N, 1 \leq j \leq M$. The main objective for the droplet routing problem is to minimize the time required to transport the droplet from the source $e_{x,y}$ to the destination $e_{k,m}$. In a single-droplet routing problem without electrode degradation, the problem can be simplified as finding the shortest path between $e_{x,y}$ and $e_{k,m}$.

### 2.1 Routing with Multiple Droplets

Multiple droplet routing tasks can be parallel executed on a DMFB at the same time. Assume that there are $n$ droplets in total, where $n \geq 2$. For any two of the droplets $d_i$ and $d_j$, where $1 \leq i, j \leq n$ and $i \neq j$, assume that their positions at timestep $t$ are $e_{x^t,y^t}$ and $e_{k^t,m^t}$, respectively. The following are the fluidic constraints that should be satisfied [6]:




(a) (b)

Fig. 4. (a) The fabricated DMFB. (b) The experimental setup.

(1) $|x^t - k^t| > 1$ or $|y^t - m^t| > 1$,
(2) $|x^{t+1} - k^t| > 1$ or $|y^{t+1} - m^t| > 1$ or $|x^t - k^{t+1}| > 1$ or $|y^t - m^{t+1}| > 1$.

With the preceding constraints, the objective of the multi-droplet routing problem is to minimize the maximal routing timestep among all the $n$ routing tasks.

### 2.2 Routing with Electrode Degradation

For the droplet routing problem that considers electrode degradation, we define function $d(e_{i,j})$ to describe the degradation status of an electrode, where $0 \leq d(e_{i,j}) \leq 1$. $d(e_{i,j})$ is 1 when the electrode $e_{i,j}$ is completely healthy. The modeling of the degradation status function will be explained in Section 5.3. Note that the value of the electrode status function cannot be obtained by the users during the execution since the degradation status is not measurable. As an electrode degrades, the success rate of droplet transition decreases. A failing transition causes the droplet to stay in the same position. For an electrode with the degradation status $d(e_{i,j})$, we assume that the success rate of transition is $d(e_{i,j})$ and the expected steps for a success transition is $1/d(e_{i,j})$. Therefore, the objective of a droplet routing problem with electrode degradation can be formulated as following: find a path $\{e_{x_1,y_1}, e_{x_2,y_2}, \ldots, e_{x_T,y_T}\}$ that minimizes $\sum_{i=1}^{T} 1/d(e_{x_i,y_i})$,where $e_{x_1,y_1}$ the is the source and $e_{x_T,y_T}$ is the destination.

## 3 ELECTRODE DEGRADATION IN DMFBS

Previous work has shown that charge trapping in a dielectric layer follows an exponential model [13, 44, 47, 84]. To independently validate this claim, we design an experiment where we monitor electrode degradation in the fabricated PCB-based DMFB.

The electrode size of the DMFB is 2×2 $mm^2$ (Figure 4(a)). Four reservoir modules are placed on two sides of the biochip; these modules are used to dispense droplets of reagents. Every electrode can be individually controlled; the control signals are provided by a control board placed below the DMFB. The activation/de-activation status of each electrode is controlled by a high-voltage relay (part no. Panasonic AQW212). A high-voltage relay in our setup is controlled by a configuration bit; the configuration bits are stored in a register (part no. Texas Instruments SN74AHC595). The details of the control hardware are shown in Figure 4(b). The Raspberry Pi 4 on the left generates control signals. We used a voltage source of 1.5 KHz and 200 Vpp for electrode actuation. To avoid introducing excessive current, a resistor $R = 1$ MΩ is placed in series between each electrode and the high-voltage source.

We developed an actuation sequence for the electrodes that leads to repeated fluidic operations on the biochip. When we execute the actuation sequence on the DMFB, each electrode is actuated

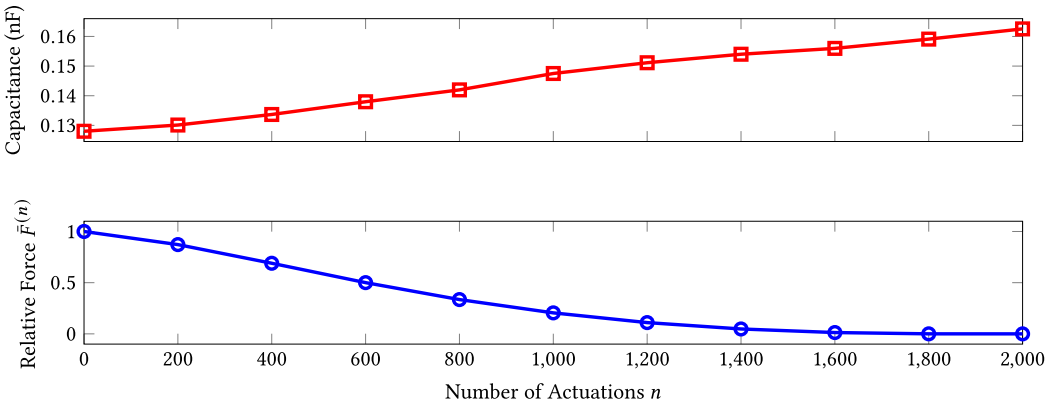



Fig. 5. Capacitance increase (top) and EWOD force degradation (bottom).

for 1 second for hundreds of times. After executing the actuation sequence, we actuated an electrode and measured the charging times needed using an oscilloscope. Because the electrode and the top plate form a capacitor, and a resistor is placed in series with the electrode, the charging path is a simple RC circuit. The effective capacitance of an electrode can be derived using the equation

$$V_C(t) = Vpp\left(1 - e^{-t/RC}\right),$$

where $C$ is the effective capacitance of the electrode, $V_C$ is the voltage of the electrode, and $t$ is time. The degradation results are shown in Figure 5. The results show that the capacitance of an electrode grows linearly as we repeatedly actuate the electrode.

The EWOD force of a droplet is given by Zhong et al. [87]:

$$F_{EWOD} = \frac{C_{unit}(V_C - V_T)^2}{2} L_{eff}, \tag{1}$$

where $V_T$ is the threshold voltage, $C_{unit}$ is the structural capacitance per unit area in the dielectric layer, and $L_{eff}$ is the length of the contact line. Therefore, the EWOD force exerted by an electrode (relative to the same EWOD force at full health) can be estimated as

$$\bar{F}^{(n)} \approx (V^{(n)}/V_a)^2, \tag{2}$$

where $n$ is the number of actuations of the electrode, $V^{(n)}$ is the actuation voltage on (potentially affected by electrode degradation), and $V_a$ is the nominal actuation voltage. By plugging our experimental results to (2), the impact of the electrode number of actuations and the relative EWOD force is shown in Figure 5. As the EWOD force correlates to the exponential actuation times, we derive an exponential model that has the least-squared error to fit the measured data. The model fitting results show that the relationship between the number of actuation $n$ and the relative EWOD force $\bar{F}^{(n)}$ can be modeled as

$$\bar{F}^{(n)} \approx \tau^{2n/c}, \tag{3}$$

where $\tau \in [0, 1]$ and $c \in \mathbb{R}$ are constants capturing the degradation rate. The degradation parameters are estimated as $\tau \in [0.5, 0.7]$ and $c \in [500, 800]$.

For realistic bioassays, the value of $n$ varies between different applications. For instance, the total number of operations range from 18 (PCR) to 1,920 (ProteinSplit7) in the work of Grissom and Brisk [17], where each operation needs several steps to be completed. Besides, additional operations might be performed when error recovery is required, increasing the total number of actuations. Thus, the number of total actuations ranges from hundreds to thousands.

## 4 BACKGROUND ON RL

### 4.1 Deep RL

An agent in an RL formulation is placed within an environment. The agent's goal is to accomplish a using task with the best performance given a set of small actions. At each step, the agent takes one of these actions, and the agent receives an observation and reward from the environment [75].

RL problems can be formally stated using Markov decision processes. A Markov decision process contains two sets ($S$ and $A$), a probability function $f$, a reward model $R$, and a variable $\gamma$. The observations made by the agent are included in a set $S$, and an observation is also referred to as a state. An element $s_t \in S$ is an observation made by the agent at time $t$. We use $A$ to denote a set of actions taken by the agent. An action $a_t \in A$ denotes the action made by the agent at time $t$. Note that $P(s_{t+1}|a_t, s_t)$ refers to the transition model; it describes what the next state $s_{t+1}$ will be after the agent takes action $a_t$ while in state $s_t$. The reward model is denoted by $R(s_t)$; it describes the agent's reward when it enters the state $s_t$. The parameter $\gamma$ is a discount factor, where $0 \leq \gamma \leq 1$ and $\gamma \in \mathbb{R}$. It represents the relative importance of immediate and future rewards. The agent's goal is to select the best policy $\pi$ that maximizes the total reward received from the environment from the start state to an end state. The expected cumulative discounted reward is expressed as $U(t) = \mathbb{E}[\sum_t \gamma^t \cdot R(s_t)]$.

### 4.2 RL Algorithms

We briefly describe three deep RL algorithms that we use to evaluate our RL framework; these algorithms are **Temporal-Difference (TD)**, on-policy gradient descent, and off-policy actor-critic approaches.

*4.2.1 Double Deep Q-Network.* The **Deep Q-Network (DQN)** algorithm [50] is a TD method that uses a neural network to approximate the state-action value function

$$Q(s,a) = \max_{\pi} \mathbb{E}\left[\sum_{0}^{\infty} \gamma^i r_{t+i} | s_t = s, a_t = a, \pi\right].$$

DQN relies on an experience replay dataset $\mathcal{D}_t = \{m_1, \ldots, m_t\}$, which stores the agent's experiences $m_t = (s_t; a_t; r_t; s_{t+1})$ to reduce correlations between observations. The experience consists of the current state $s_t$, the action the agent took $a_t$, the reward it received $r_t$, and the next state after transition $s_{t+1}$. The learning update at each iteration $j$ uses a loss function based on the TD update:

$$L_j(\theta_j) = \mathbb{E}_{m_k \sim \mathcal{D}}[(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta_j))^2],$$

where $\theta_j$ and $\theta^-$ are the parameters of the online Q-networks and the target network, respectively, and the experiences $m_k$ are sampled uniformly from $\mathcal{D}$. The parameters of the target network are fixed for a number of iterations while the online network $Q(s, a; \theta_j)$ is updated by gradient descent. In partially observable environments, an agent can only observe $o_t$ instead of the entire state $s_t$. The experience replay is therefore updated as $m_t = (o_t; a_t; r_t; o_{t+1})$.

In DQN, the *max* operator uses the same values to select an action and evaluate an action, which can lead to overoptimistic value estimation [21]. An improved method named *double DQN* was proposed to mitigate this problem [76]. In double DQN, the loss function at iteration $j$ is updated as

$$L_j(\theta_j) = \mathbb{E}_{m_k \sim \mathcal{D}}[(r + \gamma Q(s', \text{argmax}_{a'} Q(s', a'; \theta_j); \theta^-) - Q(s, a; \theta_j))^2].$$

*4.2.2 Proximal Policy Optimization Algorithm.* **Proximal Policy Optimization (PPO)** is an on-policy method that improves gradient descent stability without performance collapse [64]. It

updates policies using the following equation:

$$\theta_{k+1} = \underset{\theta}{\operatorname{argmax}} \underset{s,a\sim\pi_{\theta_k}}{\mathbf{E}} [L(s, a, \theta_k, \theta)].$$

The update usually takes several steps of stochastic gradient descent (SGD) to maximize the objective. Here, the loss function $L$ is defined as

$$L(s, a, \theta_k, \theta) = \min(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s, a), g(\epsilon, A^{\pi_{\theta_k}}(s, a))),$$

where $A$ is an estimator of the advantage function, $\epsilon$ is a hyperparameter, and

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & \text{if} A \geq 0 \\ (1-\epsilon)A & \text{if} A < 0. \end{cases}$$

*4.2.3 Actor-Critic with Experience Replay.* **Actor-Critic with Experience Replay (ACER)** is an off-policy actor-critic model that increases the sample efficiency and reduces the data correlation [80]. Similar to asynchronous advantage actor-critic (A3C) [48], ACER learns the value function by training multiple actors in parallel. To obtain stability of the off-policy estimator, ACER adopts a retrace Q-value estimation:

$$\Delta Q^{ret}(S_t, A_t) = \gamma^t \prod_{1 \leq \gamma \leq t} \min\left(c, \frac{\pi(A_\tau|S_\tau)}{\beta(A_\tau|S_\tau)}\right) \delta_t,$$

where $(\pi, \beta)$ is the target and behavior policy pair, $\delta_t$ is the TD error, and $c$ is a constant. In addition to a retrace Q-value estimation, ACER uses importance sampling and a trust region policy optimization [63].

### 4.3 MARL Training Schemes

We consider three widely used training schemes for our MARL framework: centralized, concurrent, and parameter sharing [20]. We briefly describe how each approach can be used with MARL.

***Centralized.*** The centralized learning approach assumes a joint model that receives all the observations and generates the joint actions for all the agents. A drawback of this approach is that it leads to exponential growth in the observation and actions spaces with the number of agents.

***Concurrent.*** In concurrent learning, each agent learns its own individual policy. Each independent policy maps an agent's private observation to an action. In the policy gradient approach, this means optimizing multiple policies simultaneously from the joint reward signal.

***Parameter Sharing.*** Similar to concurrent learning, each agent is assigned with a neural network policy. However, in the parameter sharing approach, all the agents share the parameters of a single policy. This allows the policy to be trained with the experiences of all agents simultaneously. However, each agent is still able to act differently based on the observation it receives.

## 5 RL APPROACH TO DROPLET ROUTER ON DMFBS

We consider a bioassay that is executed on a cyberphysical DMFB. The droplet location is determined in real time using a CCD camera [45, 81]. A controller, connected to the DMFB, is loaded with all the droplet routing tasks needed to complete the bioassay [72]. Figure 6 illustrates the overall system.

### 5.1 Droplet Routing as an RL Problem

We formulate droplet routing as a sequence of decision-making problems within the RL framework. We utilize a droplet routing agent that can make real-time observations of the DMFB, it can move

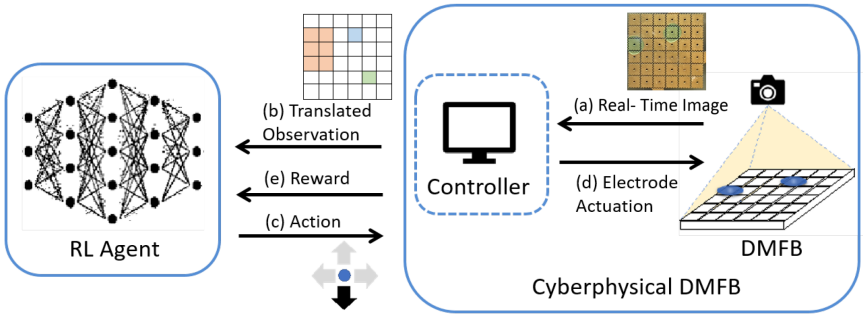



Fig. 6. The RL framework for droplet routing on DMFBs. (a) Real-time images are captured using the CCD camera. (b) The droplet locations are computed by the controller. The information is mapped to an array as input for the RL agent. (c) An action is chosen by the RL agent. (d) Electrodes are actuated by the controller based on the action. (e) The RL agent receives a reward.

a droplet to an adjacent electrode at a timestep, and the agent's goal is to transport the droplet from a given start electrode to a given destination electrode. The agent is rewarded or punished based on the state transition result after it takes an action.

***Actions.*** At any timestep, a droplet can be transported to one of the four directions: north, south, east, and west. Therefore, we define the action set as $A = \{a_n, a_s, a_e, a_w\}$; each element denotes a direction along which the droplet can be moved.

***States.*** A state $s_t$ consists of the location of the transported droplet, the droplet destination, and electrodes that are concurrently utilized by other fluidic operations. During a bioassay, multiple operations may be carried out concurrently to achieve high throughput. If a droplet is moved while a mixing operation is also being carried out, the set of electrodes used for the mixing operation cannot be used for droplet transportation in order to prevent undesirable contamination.

At any given timestep, observation made on the DMFB is processed as an RGB image. Control software is used to determine the locations of on-chip droplets [45]. The resolution of the RGB image is given by the number of electrodes on the DMFB. An electrode with a droplet on it is interpreted as a blue pixel. The destination electrode is interpreted as a green pixel. The electrodes occupied by all the other concurrent operations are interpreted as red pixels (see Figure 6(b)).

***Rewards.*** The agent is rewarded if the droplet is transported to its destination. Let $e_{i,j}$ be the $i^{\text{th}}$ row and the $j^{\text{th}}$ column electrode of the DMFB. Suppose that in state $s_t$, a droplet is present at $e_{i,j}$, and its destination is $e_{k,m}$. We define $D(s_t)$ as the Manhattan distance of the droplet from the destination at state $s_t$; $D(s_t) = |i-k| + |j-m|$. After an action $a_t$ is taken, if $D(s_{t+1}) = 0$, the agent receives a positive reward of +1.0. Otherwise, the reward is computed as follows:

$$R_t = \begin{cases} +0.5 & \text{if} D(s_{t+1}) < D(s_t) \\ -0.3 & \text{if} D(s_{t+1}) = D(s_t) \\ -0.8 & \text{if} D(s_{t+1}) > D(s_t). \end{cases}$$

In the first case, the action leads to a state in which the droplet is closer to the destination. Therefore, the reward is positive. Any positive value can facilitate agent convergence because the total reward is maximized. In the second case, the agent is punished because the action does not result in a better state. In the third case, the agent is punished with a negative value of larger magnitude because it leads to a worse state.

### 5.2 Formulation of Parallel Droplet Routing as MARL

We formulate parallel droplet in the MARL framework where agents are fully cooperative. The *action* space and *state* for each agent are similar to that of the single-agent formulation.

***Rewards.*** We consider the cooperative setting for the MARL framework [12, 19, 39, 85] because the agents should not compete with each other to transport droplets. We first compute an assessment value $r^i$ of an agent $i$ after state transition. Similar to prior definition, let $D^i(t)$ be defined as the Manhattan distance of the droplet $d_i$ from the destination at timestep $t$. After an action $a_t^i$ is taken, if $D^i(t+1) = 0$, the assessment value $r^i$ is assigned a positive value of +1.0 because the droplet has reached the destination. Otherwise, the assessment value is computed as follows:

$$r^i = \begin{cases} -0.05 & \text{if} D^i(t+1) < D^i(t) \\ -0.1 & \text{if} D^i(t+1) \geq D^i(t). \end{cases}$$

In the first case, the action leads to a state in which the droplet is closer to the destination. In the second case, the action results in the same state or even a worse state. Therefore, we use a smaller value as the assessment value. In this reward setting, to gain the maximum value in a game, the agent is encouraged to take as few steps as possible to reach the destination.

As all the agents take a combination of actions, a possible resultant state is that droplets may get too close to each other, which can lead to unintended merging and sample/reagent contamination. To prevent this scenario, we also adjust the assessment values for droplets that are too close to each other. Assume that, after a joint set of actions is taken, the resultant locations of two droplets $d^i$ and $d^j$ are $e_{a,b}^{d^i}$ and $e_{c,d}^{d^j}$, respectively. The distance of the two droplets is computed as $D(d^i, d^j) = |a-c| + |b-d|$. If $D(d^i, d^j) \leq 2$, the assessment values are adjusted as $r^i = r^i - 0.8$ and $r^j = r^j - 0.8$. In decentralized learning, each agent $i$ is rewarded by its own assessment value $r^i$; in centralized learning, we give each agent a team-average reward $R_{avg} = \frac{\sum_{i=1}^{N} r^i}{N}$.

### 5.3 DMFB Simulator: Training of RL Agents

We next describe an online droplet router, incorporated as an RL agent, that can execute all the droplet routing tasks. To train the agent, we developed an OpenAI-Gym environment named *DMFB-Env*. The DMFB matrix consists of $N \times M$ electrodes, where $N$ and $M$ are inputs to DMFB-Env.

***Transition Model.*** DMFB-Env operates in two modes: healthy and degrading. Recall that $e_{i,j}$ denotes an electrode at the $i^{\text{th}}$ row and the $j^{\text{th}}$ column of the DMFB. The transition function is defined as

$$T(e_{i,j}, a_t) = \begin{cases} e_{i-1,j} & \text{if} a_t = a_N \\ e_{i+1,j} & \text{if} a_t = a_S \\ e_{i,j+1} & \text{if} a_t = a_E \\ e_{i,j-1} & \text{if} a_t = a_W, \end{cases}$$

where $1 < i < N$ and $1 < j < M$. If the droplet is present at the boundary of the electrode array and the action is toward the outside of the biochip, the droplet will remain at the same location. For example, if the droplet is present at $e_{0,0}$ and the action is either $a_N$ or $a_W$, the droplet remains at $e_{0,0}$. Similarly, if the next location of the droplet is in the electrodes that are used for the other concurrent fluidic operations, the droplet stays at the same electrode.

For the degrading mode, we introduce a function $d(e_{i,j})$ that describes the degradation status of an electrode, where $0 \leq d(e_{i,j}) \leq 1$. If the electrode $e_{i,j}$ is healthy, $d(e_{i,j}) = 1$; otherwise, $d(e_{i,j}) = 0$. The study in the work of Dong et al. [13] showed that an electrode can only be actuated up to 200 times before it is completely degraded. Therefore, we define a degradation factor $\tau$, where

Table 1. CNN Configuration

| Layer | Type | Depth | Activation | Stride | Padding |
|---|---|---|---|---|---|
| 1 | Convolution | 32 | ReLU | 3 | 1 |
| 2 | Convolution | 32 | ReLU | 3 | 1 |
| 3 | Max Pool | N/A | N/A | 2 | 1 |
| 4 | Convolution | 64 | ReLU | 3 | 1 |
| 5 | Convolution | 64 | ReLU | 3 | 1 |
| 6 | Max Pool | N/A | N/A | 2 | 1 |
| 7 | Convolution | 128 | ReLU | 3 | 1 |
| 8 | Convolution | 128 | ReLU | 3 | 1 |
| 9 | Max Pool | N/A | N/A | 2 | 1 |
| 10 | Fully Connected | 8 | ReLU | N/A | N/A |

$0.5 \leq \tau \leq 0.7$, and the degradation function $d(e_{i,j})$ is defined as

$$d(e_{i,j}) = \tau^{\lfloor n/250 \rfloor},$$

where $n$ is the number of actuations. Each electrode is randomly assigned a different value of $\tau$ to simulate the geometric variance of the electrode array.

A Bernoulli random variable $X_{i,j}$ is defined as the transition outcome when the droplet is present at $e_{i,j}$: when $X_{i,j} = 1$, the transition is successful as $T(e_{i,j}, a_t)$; when $X_{i,j} = 0$, the transition fails, and the droplet remains at the same electrode. The probability mass function of $X_{i,j}$ is defined as

$$\begin{cases} P(X_{i,j} = 1) = d(e_{i,j}) \\ P(X_{i,j} = 0) = 1 - d(e_{i,j}). \end{cases}$$

***RL Agent.*** The RL agent is a deep neural network (see Figure 6). It observes images and chooses an action $a_t \in A$. It receives a reward value based on the outcome of the previous action.

Over the past few years, many neural network architectures have been proposed [27, 36, 68]. Because DMFBs commercially available today typically include a few hundred electrodes [86], we evaluate the effectiveness of RL-based adaptation using DMFBs of size $N \times M$, where $25 \leq N \times M \leq 1,225$. While fully connected neural networks are adequate for small DMFB instances (less than 100 electrodes), we found that they do not converge for large DMFBs. Our evaluation showed that **Convolutional Neural Networks (CNNs)** are effective for the preceding DMFB instances. However, because the network needs to be loaded on a DMFB, the computational resources on the associated controller may be limited compared to a server. For example, in the work of Willsey et al. [81], the DMFB includes only a quad-core 1.2-GHz ARMv7 processor with 1 GB of RAM, and it does not contain a GPU; therefore, large CNNs are not feasible in this application scenario. We tested several options for the number of hidden layers and the number of neurons per layer. Our results show that a simple CNN, as described in Table 1, can solve the droplet- routing problem for large DMFBs with more than 1,000 electrodes.

### 5.4 RL Training

We consider fabricated DMFBs as test cases and evaluate the effectiveness of RL-based adaptation using arrays of size $N \times N$. $N$ is set as $10 \leq N \leq 35$ since the number of total electrodes for recent commercial microfluidics biochips is around 500 on a chip [32]. For each training game of DMFB-Env, a random routing task is generated. In addition, DMFB-Env generates some random concurrent modules to simulate high-throughput bioassay execution during droplet routing. We

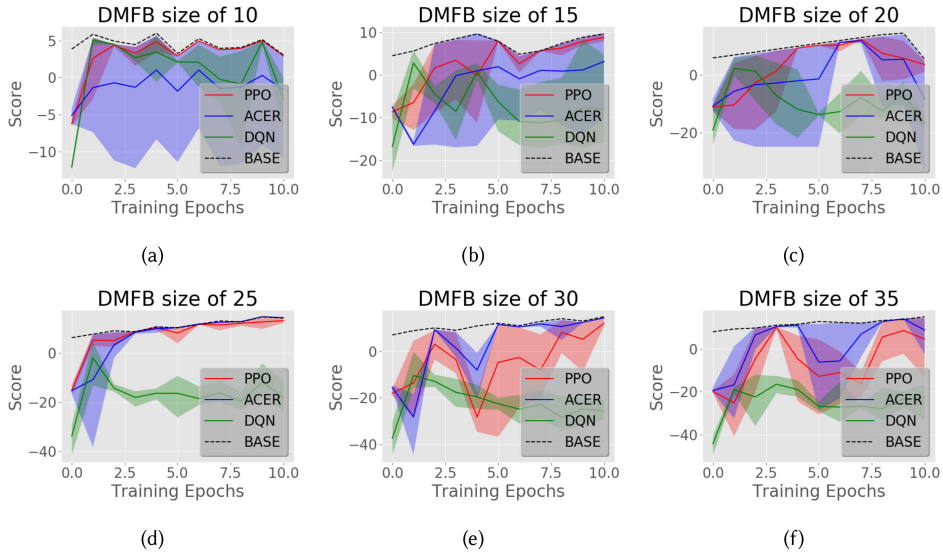



Fig. 7. Training process corresponding to different RL algorithms. Score is the total reward that the RL agents receive in a game. The performance is compared with an offline optimization method [86]. (a) $10 \times 10$ DMFB. (b) $15 \times 15$ DMFB. (c) $20 \times 20$ DMFB. (d) $25 \times 25$ DMFB. (e) $30 \times 30$ DMFB. (f) $35 \times 35$ DMFB.

evaluated three RL algorithms (i.e., double DQN, PPO, and ACER) described in Section 4 in DMFB-Env in the healthy mode. We used default parameter settings in the work of Hill et al. [23] for the three algorithms. The training was executed on a Linux platform integrated with an 11-GB-memory GPU (NVIDIA GeForce RTX 2080 Ti). The training processes using PPO take nearly 2 hours to converge, which is the fastest among the other algorithms. Although it takes several hours to train a model to perform as well as the offline method, training needs to be carried out only once, and the trained model can subsequently be used for all fabricated DMFBs. We compare the RL approaches with an offline optimization method [86].

The training processes for different sizes of DMFBs are shown in Figure 7. For each RL algorithm, we ran 18 simulations with random seeds; the average performance of each algorithm is plotted as a solid line, and the similar color region shows the interval between its best performance and its worst performance. A training epoch contains 20,000 timesteps. We observe that double DQN does not converge in all training settings. In some cases, double DQN learned sub-optimal policies first, and then the policy learned lower-reward experiences, which results in converging to more passive policies. The results are similar to RL training in other environments [34, 43]. We observe that PPO performs well in all training settings, but it sometimes takes more training epochs to converge. This is because PPO is sensitive to initialization [28, 35, 79]. In addition, the update rule of PPO encourages the policy to exploit rewards that it has already found over the training course. Therefore, if an initial network policy is far from global optima, the policy can be easily trapped in local minima. We also observe that ACER does not perform well in some training settings. As the action space and observation space grow exponentially, the experiences stored in the limited replay buffer become important for ACER training.

Our training results show that in all training settings, PPO can outperform the other two RL algorithms. To fine-tune the RL approach using PPO, we tested two significant parameters in PPO to find the best performance of our RL agent for different sizes of DMFBs, the number of concurrent environments, and the number of steps for each update.

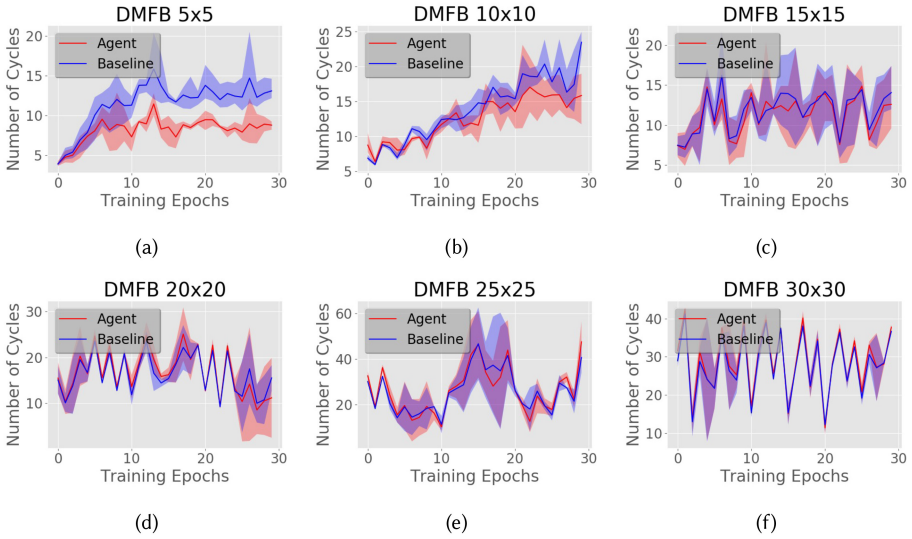



Fig. 8. Evaluation of the trained models in degrading mode of DMFB-Env. The performance, expressed as the required number of actuation (clock) cycles, is compared with the static routing method from Zhao and Chakrabarty [86]. (a) $5 \times 5$ DMFB. (b) $10 \times 10$ DMFB. (c) $15 \times 15$ DMFB. (d) $20 \times 20$ DMFB. (e) $25 \times 25$ DMFB. (f) $30 \times 30$ DMFB.

Figure 9 shows the training rewards for agents with varying the number of concurrent environments and the number of steps for each update. Here, we show the training rewards for a $10 \times 10$ DMFB, a $20 \times 20$ DMFB, and a $30 \times 30$ DMFB. The training is not stable when there are only a few concurrent environments. For example, when there are four environments, we found out that the performance of the training model (updated every 16 steps) drops significantly after a few training epochs. We also observed that for eight environments, irrespective of the update step interval, the model's performance is consistently better. Similar trends are observed in training for other sizes of DMFBs. Therefore, we chose eight concurrent environments as the PPO setting for model training.

We produced a video recording of droplet routing for a $5 \times 8$ DMFB during training (see [41]). From the video, we see that, at first, the agent moved the droplet randomly without knowing the policy needed to reach the destination. After 200K timesteps, the agent started to "learn" from past experience; early on, after 400K timesteps, it could transport the droplet to the destination using the shortest path for only a few of the routing tasks. However, after 800K timesteps, the agent was able to complete all the routing tasks using the shortest paths.

### 5.5 MARL Training

To train the agents, we developed a PettingZoo-Gym environment to simulate the parallel droplet scenarios. For each training game, $n_{rt}$ random routing tasks are generated, where $n_{rt} = \{2, 3\}$. Each routing task is performed concurrently by one of the agents. The size of the DMFB is $N \times N$, where $10 \leq N \leq 30$. We first performed the agent training using three RL algorithms (PPO, double DQN, and ACER). We also used three different MARL training schemes, including centralized, concurrent, and parameter sharing.

Figure 10 shows the training processes for two and three concurrent routing tasks in the healthy mode. A training epoch contains 20,000 timesteps. The performance of different algorithms is compared with the offline optimization method (Baseline) and the RL agents that are trained under

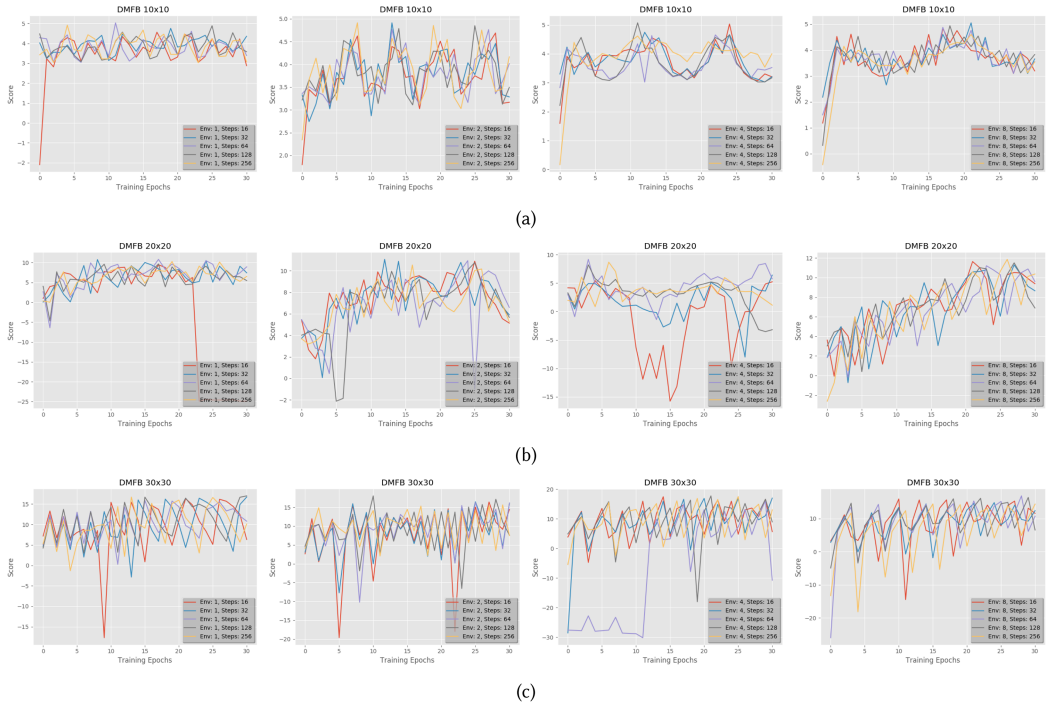



Fig. 9. Training rewards for agents with different hyper-parameter settings. Score is the total reward that the RL agent receives in a game. (a) Training rewards for DMFBs of size $10 \times 10$ electrodes. (b) Training rewards for DMFBs of size $20 \times 20$ electrodes. (c) Training rewards for DMFBs of size $30 \times 30$ electrodes.

single routing task environments (Single). The results show that the concurrent scheme is the most effective and efficient scheme to train the MARL routing models for DMFBs. We observed that PPO and ACER have similar performance, whereas DQN fails to converge in all the training settings. In some of the settings, such as the concurrent training with DMFBs of size $20 \times 20$, the ACER algorithm converges faster than the PPO algorithm.

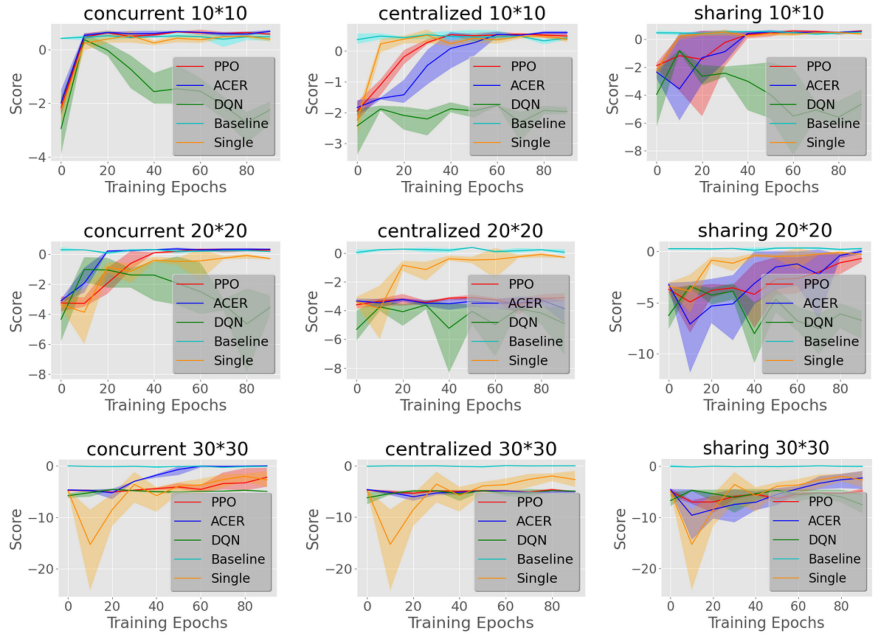
However, the figure shows that single agents can achieve comparable performance as PPO and ACER when the size of DMFB is small. However, as the size of DMFB grows and the number of concurrent routing task increases, the performance of single-agent models rapidly decrease since the single-agent models did not learn the coordination between droplets. The results illustrate the importance of MARL models for concurrent routing scenarios.

## 6 EVALUATION

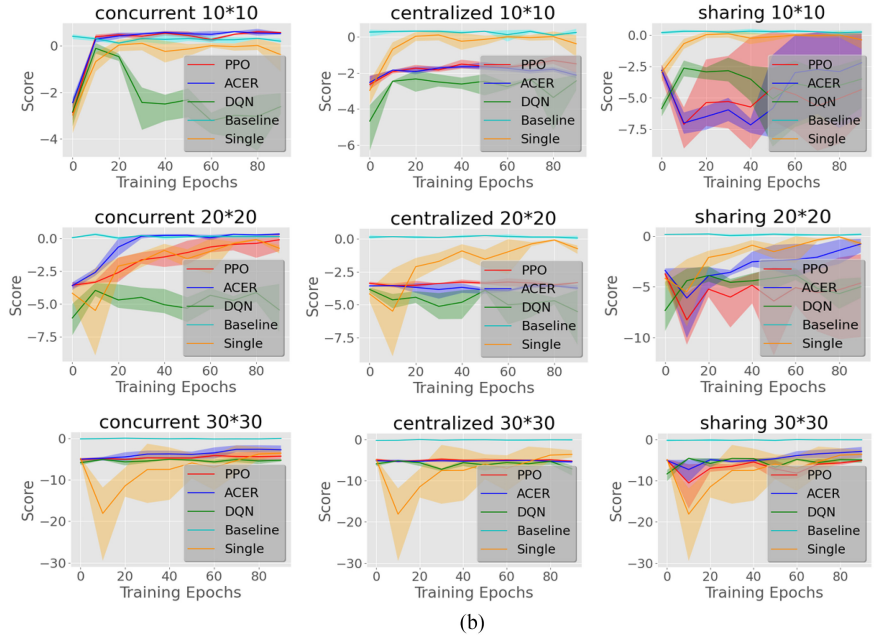
To evaluate our RL framework, we considered DMFBs with the number of electrodes ranging from 25 to 900. For each DMFB, we first trained three models with the same network architecture (as described in Table 1) using DMFB-Env, and the models were trained in the healthy mode to achieve the same performance as that of the baseline [86]. After training, we evaluated the performance of the models in the degrading mode of DMFB-Env. We also evaluated the RL framework by executing an epigenetic bioassay on a fabricated biochip.

### 6.1 Single-Agent Simulation Results

We compared the performance of the agent with the work of Zhao and Chakrabarty [86]. We set 50% of the degrading electrodes for DMFBs, and the results are shown in Figure 8. Here, we show




(a)




(b)

Fig. 10. Training process corresponding to different RL algorithms and training schemes with two concurrent routing tasks (a) and three concurrent routing tasks (b).

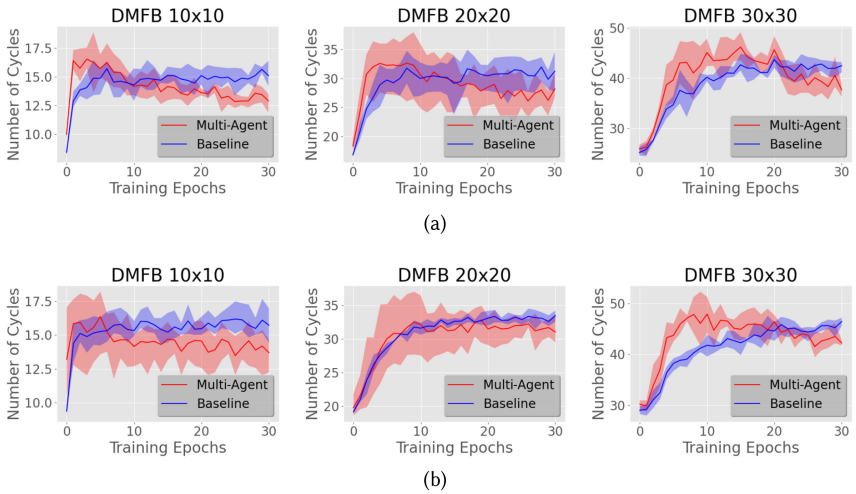



Fig. 11. Training results for MARL agents under degrading mode with two concurrent routing tasks (a) and three concurrent routing tasks (b).

the number of actuation cycles required in a game as the performance. The fewer actuation cycles required in a game, the better the performance is. We observe that the agent performs similar to the static (offline) method when the DMFBs start to degrade. This is because the RL agent has been trained to perform as well as the baseline in the healthy mode of DMFB-Env. After a small number of training games, the RL agent sometimes performs slightly worse because the agent may explore other alternative routes to avoid the degraded electrodes, and the alternative solutions may be worse than the original route. However, as DMFBs degrade further, the agent can outperform the baseline. We also observe that the proposed solution is more effective for smaller DMFBs. This is because, in our experimental setting, the DMFB with 25 electrodes is the most dynamic environment. The performance of the baseline method decreases if electrode degradation occurs in a DMFB. We see that the performance of the baseline method significantly decreases in the $5 \times 5$ DMFB. The experimental results show that the agent can adapt to all sizes of DMFBs, including the most dynamic environment (i.e., the $5 \times 5$ DMFB).

We recorded a video of droplet transportation in a simulated degraded environment; the video, called *Simulation.mp4*, can be found in the work of Liang et al. [41]. As some electrodes started to degrade, the agent can still use them to transport the droplet. In the simulated environments, sets of faults with different sizes have been injected. However, the agent is able to learn the changing health conditions of these electrodes. For subsequent tasks, the agent transports the droplet without using these degraded electrodes.

### 6.2 MARL Simulation Results

In the degrading mode of MARL, we set 10% of the degrading electrodes, and the degrading level of these electrodes increases as these electrodes are used over time. We compared the performance of the MARL models with the baseline method. The results are shown in Figure 11. We used the concurrent method to train the MARL models since concurrent is the most effective training method as discussed in Section 5. For DMFBs with the size of $10 \times 10$ and $20 \times 20$, we used PPO as the training algorithm since PPO and ACER achieve similar performance while the training processes of PPO are faster. For DMFBs with the size of $30 \times 30$, we used ACER as the training algorithm since ACER achieves the best performance among the three algorithms.

Table 2. Runtime (s) for RL Training and Referencing on a Micro-Computer

| Biochip Size | $5 \times 5$ | $10 \times 10$ | $15 \times 15$ | $20 \times 20$ | $25 \times 25$ | $30 \times 30$ | $35 \times 35$ |
|---|---|---|---|---|---|---|---|
| Training | 0.01 | 0.02 | 0.03 | 0.04 | 0.06 | 0.07 | 0.1 |
| Referencing | 0.02 | 0.01 | 0.01 | 0.15 | 0.01 | 0.14 | 0.02 |

The degradation processes are shown in Figure 11, where the performance is evaluated using the number of cycles needed to transport all the droplets to the destinations. Figure 11 shows that as the electrodes start to degrade, the MARL agent performs slightly worse than the baseline method since the agents are learning to avoid degraded electrodes and exploring alternative routes, which are longer than the routes taken by the baseline method. After several training epochs, the MARL agents outperform the baseline method as the DMFBs degrade further and the MARL agents have learned from the previous training games. As shown in Figure 11, for DMFBs with sizes of $10 \times 10$, $20 \times 20$, and $30 \times 30$, the number of training epochs that the model needs to adapt to the degrading environments are around 5 to 10, 10 to 15, and 20, respectively. The results show that the MARL agents can adapt to dynamically degraded environments under different sizes of DMFBs and provide more reliable routing strategies than the baseline method.

### 6.3 RL Runtime on a Micro-Computer

As the RL router learns to adapt to a degrading biochip, the RL agent needs to be repeatedly trained and referenced on the micro-computer of the DMFB system during the biochip execution. We profiled the runtime of the PPO training and referencing for each timestep on a micro-computer (Raspberry Pi 4) for various sizes of DMFBs (Table 2). Although the micro-computer includes a modest 1.5-GHz quad-core processor and only 4 GB of memory, one training timestep takes only about 0.04 seconds, and one referencing timestep takes only about 0.06 seconds. In our DMFB design, the actuation time required to move one droplet from an electrode to an adjacent electrode is 1 second. Therefore, the training step can be carried out concurrently while the fluidic operation occurs. The additional referencing time for the RL agent to determine the next fluidic operation is 0.06 seconds. The timing overhead of using the RL framework is therefore 6% when compared with the original DMFB system, which is negligible in practice.

### 6.4 Bioassay Execution on a Fabricated Biochip

In this section, we show the feasibility of deploying our RL model on a fabricated chip. The model deployment is general regardless of the sizes of biochips. In addition, the proposed RL framework can be used for any bioassay. As a specific case study, We designed and executed an epigenetic bioassay on a fabricated DMFB because benchtop epigenetic bioassays require large sample volumes and long execution time, and are labor intensive. Previous work has shown the effectiveness of epigenetic bioassays on DMFBs [30]. This epigenetic bioassay includes 19 routing tasks. We used the trained RL droplet router to transport droplets.

*6.4.1 Epigenetic Bioassay.* Even though all cells in the human body have the same DNA, or genotype, considerable differences in cell type and function, or phenotype, arise from the selective expression and suppression of certain genes. This phenotypic control can be attributed to various epigenetic mechanisms. These are processes and environmental factors that alter genomic behavior and its subsequent expression without any changes to the actual DNA. Epigenetics is the study of these factors and mechanisms of control in healthy and diseased populations. **Chromatin Immunoprecipitation (ChIP)** is used to study the epigenetic relationship between DNA and its supporting proteins [10]. Running a full ChIP protocol on a single sample requires

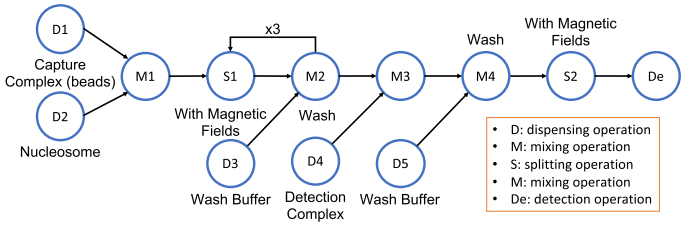
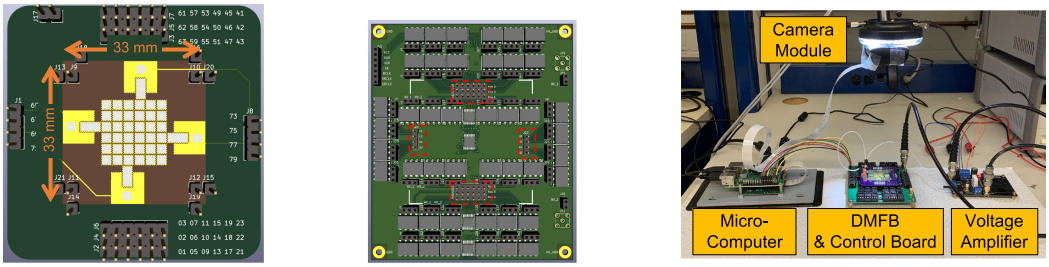



Fig. 12. The steps involved in a NuIP assay.




(a) (b) (c)

Fig. 13. (a) The fabricated DMFB. (b) The control board for the DMFB. (c) The experimental setup.

a large starting volume of cells (which are not always available) and several days to run the assay, and is highly labor intensive. We consider **Nucleosome Immunoprecipitation (NuIP)** on magnetic beads in order to translate ChIP from the benchtop to automated DMFBs to reduce sample sizes, decrease runtimes, and increase throughput.

The NuIP protocol modifies the traditional ChIP assay [10, 52] by first functionalizing a magnetic bead off-chip with an antibody that targets one of the histone proteins in the nucleosome of interest. This is the capture complex as shown in Figure 12. The nucleosome-containing sample is then mixed and incubated with the capture complex followed by magnetic splitting and washing steps. In the meantime, off-chip, an antibody specific to a different histone protein in the nucleosome is incubated with a fluorescent secondary antibody. This forms the detection complex reagent. Next, the beads are incubated with the detection complex. If there are nucleosomes attached to the beads, these will bind with the detection complex. After the excess detection complex is washed away, ensuring that there are no false positives, the beads are resuspended in a droplet and routed to the detection region. An LED tuned to the excitation wavelength of the fluorescent antibody shines on the beads which are imaged using a CCD camera outfitted with the appropriate emission wavelength filter. A fluorescing sample confirms the presence of the nucleosome of interest.

*6.4.2 Experimental Setup.* ***Fabricated DMFB.*** For our experiment, we designed a PCB-based DMFB and fabricated it using OSH Park [55]. The DMFB contains a $6 \times 6$ electrode-array (Figure 13(a)). A reservoir module is placed on each side of the array, and the modules can dispense different reagent droplets. Each electrode can be controlled individually. The control signals come from the pin heads that are soldered on the board boundary.

***Control Board.*** For the fabricated DMFB, the activation/de-activation status of each electrode is controlled by a high voltage relay (part no. Panasonic AQW212). A total of 44 relay ICs are soldered on the control board (36 for electrode array and 8 for reservoir modules) (see Figure 13(b)). Each high-voltage relay IC is controlled by a configuration bit, and these configuration bits are stored in the register ICs (part no. Texas Instruments SN74AHC595). In addition to these ICs, four

pin-header modules (shown within the red rectangles) are used as the DMFB socket, which allows DMFB replacement on the control board.

***Overall System.*** Figure 13(c) shows the hardware setup used to operate the DMFB. The DMFB is installed above the control board using the pin-header socket. A micro-computer (Raspberry Pi 4) on the left is used to generate control signals, and the RL agent is installed in the micro-computer. An amplifier board as well as the functional generator are used to generate a voltage source of 1 KHz and 200 Vpp, which provides actuation signals for the electrodes. A camera placed on top of the DMFB captures the droplet locations. The images are then utilized by the micro-computer for making real-time decisions.

*6.4.3 Experimental Results.* We performed the droplet routing tasks of the bioassay using our fabricated DMFB, where we simulated the degradation on an electrode at the location (3, 4). The degradation is simulated by applying a lower voltage of 150 Vpp on the electrode. During the third routing task, the degraded electrode is involved in the droplet transportation path and thus a failure occurred. Then, in the following routing task, the RL agent successfully learned from the experience and adopted an alternative path to avoid the degraded electrode. Examples of routing tasks on fabricated DMFB can be seen in previous work [41]. In the recorded video *DMFBExperiment.mp4* [41], intuitive routing cases are presented to show the effectiveness of our RL routing model.

## 7 CONCLUSION

We presented a novel framework for RL-based droplet routing on DMFBs. We also developed an OpenAI-Gym environment that can be used to train the RL droplet router for various DMFB sizes. The simulation is based on a study of electrode degradation using fabricated DMFBs. The experimental results showed that even though electrodes on a DMFB degrade over time, the RL droplet router can learn the degradation behavior and transport droplets using only healthy electrodes.

We also formulated a MARL framework for parallel droplet routing on DMFBs. We introduced a PettingZoo-Gym environment for DMFBs to perform the training of MARL agents. Experimental results showed that the MARL framework can learn from degradation environments and provide superior routing strategies, which results in fewer re-routes for failures, and thus the completion time of bioassay can be faster and a smaller volume of biosamples is needed.

We identified the timing constraint associated with the use of the RL approach on a micro-computer that does not contain a GPU. The results showed that the proposed RL approach does not impede the fluidic operations in time-critical bioassays. A failure of the DMFB results in costly sample and reagent loss. However, the proposed RL framework minimizes the need to discard biochips with degraded electrodes and abort bioassay protocols. This increases the lifespan of a biochip's utility and allows for the adaptation of a plethora of immunoprecipitation assays onto the DMFB platform.

## REFERENCES

[1] Mirela Alistar, Paul Pop, and Jan Madsen. 2016. Synthesis of application-specific fault-tolerant digital microfluidic biochip architectures. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 5 (2016), 764–777. https://doi.org/10.1109/TCAD.2016.2528498

[2] Noam Brown and Tuomas Sandholm. 2019. Superhuman AI for multiplayer poker. *Science* 365, 6456 (2019), 885–890.

[3] Donald H. Chace, Victor R. De Jesús, and Alan R. Spitzer. 2014. Clinical chemistry and dried blood spots: Increasing laboratory utilization by improved understanding of quantitative challenges. *Bioanalysis* 6, 21 (2014), 2791–2794. https://doi.org/10.4155/bio.14.237

[4] Krishnendu Chakrabarty, Richard B. Fair, and Jun Zeng. 2010. Design tools for digital microfluidic biochips: Toward functional diversification and more than Moore. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 29, 7 (2010), 1001–1017.

[5] Ying-Han Chen, Chung-Lun Hsu, Li-Chen Tsai, Tsung-Wei Huang, and Tsung-Yi Ho. 2013. A reliability-oriented placement algorithm for reconfigurable digital microfluidic biochips using 3-D deferred decision making technique. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32, 8 (2013), 1151–1162.

[6] Minsik Cho and David Z. Pan. 2008. A high-performance droplet routing algorithm for digital microfluidic biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27, 10 (2008), 1714–1724. https://doi.org/10.1109/TCAD.2008.2003282

[7] Kihwan Choi, Alphonsus H. C. Ng, Ryan Fobel, and Aaron R. Wheeler. 2012. Digital microfluidics. *Annual Review of Analytical Chemistry* 5 (2012), 413–440.

[8] Wei-Lung Chou, Pee-Yew Lee, Cing-Long Yang, Wen-Ying Huang, and Yung-Sheng Lin. 2015. Recent advances in applications of droplet microfluidics. *Micromachines* 6, 9 (2015), 1249–1271.

[9] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. 2019. Quantifying generalization in reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, Kamalika Chaudhuri and Ruslan Slakhutdinov (Eds.). Proceedings of Machine Learning Research, Vol. 97. PMLR, 1282–1289. https://proceedings.mlr.press/v97/cobbe19a.html

[10] Philippe Collas. 2010. The current state of chromatin immunoprecipitation. *Molecular Biotechnology* 45, 1 (2010), 87–100.

[11] Peter Dayan and Geoffrey E. Hinton. 1992. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems*, S. Hanson, J. Cowan, and C. Giles (Eds.), Vol. 5. Morgan-Kaufmann, 1–8.

[12] Thinh T. Doan, Siva Theja Maguluri, and Justin Romberg. 2019. Finite-time analysis of distributed TD(0) with linear function approximation for multi-agent reinforcement learning. *arXiv preprint arXiv:1902.07393* (2019).

[13] Cheng Dong, Tianlan Chen, Jie Gao, Yanwei Jia, Pui-In Mak, Mang-I. Vai, and Rui P. Martins. 2015. On the droplet velocity and electrode lifetime of digital microfluidics: Voltage actuation techniques and comparison. *Microfluidics and Nanofluidics* 18, 4 (2015), 673–683.

[14] Antonis I. Drygiannakis, Athanasios G. Papathanasiou, and Andreas G. Boudouvis. 2008. On the connection between dielectric breakdown strength, trapping of charge, and contact angle saturation in electrowetting. *Langmuir* 25, 1 (2008), 147–152.

[15] Mahmoud Elfar, Yi-Chen Chang, Harrison Hao-Yu Ku, Tung-Che Liang, Krishnendu Chakrabarty, and Miroslav Pajic. 2023. Deep reinforcement learning-based approach for efficient and reliable droplet routing on MEDA biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 42, 4 (2023), 1212–1222. https://doi.org/10.1109/TCAD.2022.3194808

[16] Anurup Ganguli, Ariana Mostafa, Jacob Berger, Mehmet Y. Aydin, Fu Sun, Sarah A. Stewart de Ramirez, Enrique Valera, Brian T. Cunningham, William P. King, and Rashid Bashir. 2020. Rapid isothermal amplification and portable detection system for SARS-CoV-2. *Proceedings of the National Academy of Sciences* 117, 37 (2020), 22727–22735.

[17] Daniel T. Grissom and Philip Brisk. 2014. Fast online synthesis of digital microfluidic biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33, 3 (2014), 356–369. https://doi.org/10.1109/TCAD.2013.2290582

[18] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *Proceedings of the International Conference on Robotics and Automation (ICRA'17)*. IEEE, Los Alamitos, CA, 3389–3396.

[19] Maxime Guériau, Romain Billot, Nour-Eddin El Faouzi, Salima Hassas, and Frédéric Armetta. 2015. Multi-agent dynamic coupling for cooperative vehicles modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[20] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative multi-agent control using deep reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*. 66–83.

[21] Hado Hasselt. 2010. Double Q-learning. *Proceedings of the 23rd International Conference on Neural Information Processing Systems*. 2613–2621.

[22] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 820–828.

[23] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. 2018. Stable baselines. *GitHub*. Retrieved November 27, 2023 from https://github.com/hill-a/stable-baselines

[24] Tsung-Yi Ho, Krishnendu Chakrabarty, and Paul Pop. 2011. Digital microfluidic biochips: Recent research and emerging challenges. In *Proceedings of the IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. 335–344.

[25] Tsung-Yi Ho, Jun Zeng, and Krishnendu Chakrabarty. 2010. Digital microfluidic biochips: A vision for functional diversity and more than Moore. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'10)*. IEEE, Los Alamitos, CA, 578–585.

[26] Patrick V. Hopkins, Carlene Campbell, Tracy Klug, Sharmini Rogers, Julie Raburn-Miller, and Jami Kiesling. 2015. Lysosomal storage disorder screening implementation: Findings from the first six months of full population pilot testing in Missouri. *Journal of Pediatrics* 166, 1 (2015), 172–177.

[27] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[28] Chloe Ching-Yun Hsu, Celestine Mendler-Dünner, and Moritz Hardt. 2020. Revisiting design choices in proximal policy optimization. *arXiv preprint arXiv:2009.10897* (2020).

[29] Tsung-Wei Huang, Tsung-Yi Ho, and Krishnendu Chakrabarty. 2011. Reliability-oriented broadcast electrode-addressing for pin-constrained digital microfluidic biochips. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'11)*. IEEE, Los Alamitos, CA, 448–455.

[30] Mohamed Ibrahim, Craig Boswell, Krishnendu Chakrabarty, Kristin Scott, and Miroslav Pajic. 2016. A real-time digital-microfluidic platform for epigenetics. In *Proceedings of the International Conference on Compilers, Architectures, and Synthesis for Embedded Systems*. 1–10.

[31] Illumina. 2015. Illumina NeoPrep Library Prep System. Retrieved January 14, 2020 from https://emea.illumina.com/company/news-center/press-releases/2015/2018793.html

[32] Baebies Inc. 2020. Versatility of Digital Microfluidics for Screening and Clinical Testing in Newborns. Retrieved December 20, 2022 from https://baebies.com/versatility-of-digital-microfluidics-for-screening-and-clinical-testing-in-newborns/

[33] Baebies Inc. 2021. Baebies Official Website. Retrieved December 20, 2022 from https://baebies.com

[34] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. 2020. Graph convolutional reinforcement learning. In *Proceedings of the International Conference on Learning Representations*.

[35] Aristotelis Lazaridis, Anestis Fachantidis, and Ioannis Vlahavas. 2020. Deep reinforcement learning: A state-of-the-art walkthrough. *Journal of Artificial Intelligence Research* 69 (2020), 1421–1471.

[36] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86, 11 (1998), 2278–2324.

[37] Jinho Lee, Raehyun Kim, Seok Won Yi, and Jaewoo Kang. 2020. MAPS: Multi-agent reinforcement learning-based portfolio management system. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 4520–4526.

[38] Jia Li and Chang-Jin "CJ" Kim. 2020. Current commercialization status of electrowetting-on-dielectric (EWOD) digital microfluidics. *Lab on a Chip* 20, 10 (2020), 1705–1712.

[39] Yuqian Li and Vincent Conitzer. 2015. Cooperative game solution concepts that maximize stability under noise. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 979–985.

[40] Zipeng Li, Kelvin Yi-Tse Lai, Po-Hsien Yu, Krishnendu Chakrabarty, Miroslav Pajic, Tsung-Yi Ho, and Chen-Yi Lee. 2016. Error recovery in a micro-electrode-dot-array digital microfluidic biochip. In *Proceedings of the 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD'16)*. IEEE, Los Alamitos, CA, 1–8. https://doi.org/10.1145/2966986.2967035

[41] Tung-Che Liang, Yi-Chen Chang, Zhanwei Zhong, Yaas Bigdeli, Tsung-Yi Ho, Krishnendu Chakrabarty, and Richard Fair. 2022. Recorded Videos during Training and Evaluation. Retrieved December 15, 2022 from https://duke.is/vc86a

[42] Tung-Che Liang, Zhanwei Zhong, Yaas Bigdeli, Tsung-Yi Ho, Krishnendu Chakrabarty, and Richard Fair. 2020. Adaptive droplet routing in digital microfluidic biochips using deep reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML'20)*.

[43] Tung-Che Liang, Jin Zhou, Yun-Sheng Chan, Tsung-Yi Ho, Krishnendu Chakrabarty, and Chen-Yi Lee. 2021. Multi-agent reinforcement learning, microfluidics, MEDA biochips. In *Proceedings of the International Conference on Machine Learning (ICML'21)*.

[44] Yang Liu, Ajit Shanware, Luigi Colombo, and Robert Dutton. 2006. Modeling of charge trapping induced threshold-voltage instability in high-$\kappa$ gate dielectric FETs. *IEEE Electron Device Letters* 27, 6 (2006), 489–491.

[45] Yan Luo, Krishnendu Chakrabarty, and Tsung-Yi Ho. 2012. Error recovery in cyberphysical digital microfluidic biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32, 1 (2012), 59–72.

[46] Michael S. Manak, Jonathan S. Varsanik, Brad J. Hogan, Matt J. Whitfield, Wendell R. Su, Nikhil Joshi, Nicolai Steinke, Andrew Min, Delaney Berger, Robert J. Saphirstein, Gauri Dixit, Thiagarajan Meyyappan, Hui-May Chu, Kevin B. Knopf, David M. Albala, Grannum R. Sant, and Ashok C. Chander. 2018. Live-cell phenotypic-biomarker microfluidic assay for the risk stratification of cancer patients via machine learning. *Nature Biomedical Engineering* 2, 10 (2018), 761–772.

[47] William K. Meyer and Dwight L. Crook. 1983. Model for oxide wearout due to charge trapping. In *Proceedings of the 21st International Reliability Physics Symposium (IRPS'83)*. IEEE, Los Alamitos, CA, 242–247.

[48] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the International Conference on Machine Learning*. 1928–1937.

[49] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. *CoRR abs/1602.01783* (2016).

[50] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[51] Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. *arXiv preprint arXiv:1506.08941* (2015).

[52] Joel D. Nelson, Oleg Denisenko, and Karol Bomsztyk. 2006. Protocol for the fast chromatin immunoprecipitation (ChIP) method. *Nature Protocols* 1, 1 (2006), 179.

[53] NIH. 2021. NIH Delivering New COVID-19 Testing Technologies to Meet U.S. Demand. Retrieved January 30, 2021 from https://www.nih.gov/news-events/news-releases/nih-delivering-new-covid-19-testing-technologies-meet-us-demand

[54] NIH. 2021. Rapid Acceleration of Diagnostics (RADX). Retrieved January 30, 2021 from https://www.nih.gov/research-training/medical-research-initiatives/radx

[55] OSH Park. 2020. PCB Fabrication Company. Retrieved January 31, 2020 from https://oshpark.com

[56] Yabo Ouyang, Jiming Yin, Wenjing Wang, Hongbo Shi, Ying Shi, Bin Xu, Luxin Qiao, Yingmei Feng, Lijun Pang, Feili Wei, Xhianghua Guo, Ronghua Jin, and Dexi Chen. 2020. Down-regulated gene expression spectrum and immune responses changed during the disease progression in COVID-19 patients. *Clinical Infectious Diseases* 71, 16 (2020), 2052–2060.

[57] P. Palanisamy. 2020. Multi-agent connected autonomous driving using deep reinforcement learning. In *Proceedings of the International Joint Conference on Neural Networks*. 1–7. https://doi.org/10.1109/IJCNN48605.2020.9207663

[58] Jun Park, Seung Lee, and Hyoung Kang. 2010. Fast and reliable droplet transport on single-plate electrowetting on dielectrics using nonfloating switching method. *Biomicrofluidics* 4 (2010), 024102. https://doi.org/10.1063/1.3398258

[59] Virginia M. Pierce and Richard L. Hodinka. 2012. Comparison of the GenMark diagnostics eSensor respiratory viral panel to real-time PCR for detection of respiratory viruses in children. *Journal of Clinical Microbiology* 50, 11 (2012), 3458–3465.

[60] Michael G. Pollack, Richard B. Fair, and Alexander D Shenderov. 2000. Electrowetting-based actuation of liquid droplets for microfluidic applications. *Applied Physics Letters* 77, 11 (2000), 1725–1726.

[61] Ahmad E. L. Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2017. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging* 2017, 19 (2017), 70–76.

[62] Jonathan E. Schmitz and Yi-Wei Tang. 2018. The GenMark ePlex®: Another weapon in the syndromic arsenal for infection diagnosis. *Future Microbiology* 13, 16 (2018), 1697–1708.

[63] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*. 1889–1897.

[64] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[65] Cormac Sheridan. 2020. COVID-19 spurs wave of innovative diagnostics. *Nature Biotechnology* 38, 7 (2020), 769–772.

[66] Vineeta Shukla, Fawnizu Azmadi Hussin, Nor Hisham Hamid, and Noohul Basheer Zain Ali. 2017. Advances in testing techniques for digital microfluidic biochips. *Sensors* 17, 8 (2017), 1719. https://doi.org/10.3390/s17081719

[67] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. 2017. Mastering the game of GO without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[68] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[69] Rama S. Sista, Rainer Ng, Miriam Nuffer, Michael Basmajian, Jacob Coyne, Jennifer Elderbroom, Daniel Hull, Kathryn Kay, Maithri Krishnamurthy, Christopher Roberts, Daniel Wu, Adam D. Kennedy, Rajendra Singh, Vijay Srinivasan, and Vamsee K. Pamula. 2020. Digital microfluidic platform to maximize diagnostic tests with low sample volumes from newborns and pediatric patients. *Diagnostics* 10, 1 (2020), 21.

[70] Fei Su and Krishnendu Chakrabarty. 2004. Architectural-level synthesis of digital microfluidics-based biochips. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD'04)*. 223–228.

[71] Fei Su and Krishnendu Chakrabarty. 2006. Yield enhancement of reconfigurable microfluidics-based biochips using interstitial redundancy. *ACM Journal on Emerging Technologies in Computing Systems* 2, 2 (2006), 104–128.

[72] Fei Su, Krishnendu Chakrabarty, and Richard B. Fair. 2006. Microfluidics-based biochips: Technology issues, implementation platforms, and design-automation challenges. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25, 2 (2006), 211–223.

[73] Fei Su, William Hwang, and Krishnendu Chakrabarty. 2006. Droplet routing in the synthesis of digital microfluidic biochips. In *Proceedings of the Design, Automation, and Test in Europe Conference*, Vol. 1. 1–6.

[74] Fu Sun, Anurup Ganguli, Judy Nguyen, Ryan Brisbin, Krithika Shanmugam, David L. Hirschberg, Matthew B. Wheeler, Rashid Bashir, David M. Nash, and Brian T. Cunningham. 2020. Smartphone-based multiplex 30-minute nucleic acid test of live virus from nasal swab extract. *Lab on a Chip* 20, 9 (2020), 1621–1627.

[75] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

[76] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[77] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Remi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Plaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wunsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575, 7782 (2019), 350–354.

[78] Akifumi Wachi. 2019. Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

[79] Yuhui Wang, Hao He, Xiaoyang Tan, and Yaozhong Gan. 2019. Trust region-guided proximal policy optimization. In *Proceedings of the 33rd Conference on Neural Information Processing Systems*. 1–11.

[80] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. 2017. Sample efficient actor-critic with experience replay. In *Proceedings of the International Conference on Learning Representations*.

[81] Max Willsey, Ashley P. Stephenson, Chris Takahashi, Pranav Vaid, Bichlien H. Nguyen, Michal Piszczek, Christine Betts, Sharon Newman, Sarang Joshi, Karin Strauss, and Luis Ceze. 2019. Puddle: A dynamic, error-correcting, full-stack microfluidics platform. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'19)*. 183–197.

[82] Tao Xu and Krishnendu Chakrabarty. 2007. Integrated droplet routing in the synthesis of microfluidic biochips. In *Proceedings of the Design Automation Conference (DAC'07)*. 948–953.

[83] Paloma Yáñez-Sedeño, María Chicharro, Reynaldo Villalonga, and José Pingarrón. 2014. Biosensors in forensic analysis. A review. *Analytica Chimica Acta* 823C (2014), 1–19. https://doi.org/10.1016/j.aca.2014.03.011

[84] Sufi Zafar, Alessandro Callegari, Evgeni Gusev, and Massimo V. Fischetti. 2003. Charge trapping related threshold voltage instabilities in high permittivity gate dielectric stacks. *Journal of Applied Physics* 93, 11 (2003), 9298–9303.

[85] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Başar. 2018. Fully decentralized multi-agent reinforcement learning with networked agents. *arXiv preprint arXiv:1802.08757* (2018).

[86] Yang Zhao and Krishnendu Chakrabarty. 2012. Simultaneous optimization of droplet routing and control-pin mapping to electrodes in digital microfluidic biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 31, 2 (2012), 242–254.

[87] Zhanwei Zhong, Zipeng Li, Krishnendu Chakrabarty, Tsung-Yi Ho, and Chen-Yi Lee. 2018. Micro-electrode-dot-array digital microfluidic biochips: Technology, design automation, and test techniques. *IEEE Transactions on Biomedical Circuits and Systems* 13, 2 (2018), 292–313.

[88] Zhanwei Zhong, Tung-Che Liang, and Krishnendu Chakrabarty. 2020. Reliability-oriented IEEE Std. 1687 network design and block-aware high-level synthesis for MEDA biochips. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'20)*. IEEE, Los Alamitos, CA, 544–549.

Received 27 July 2022; revised 27 March 2023; accepted 4 November 2023