Extended Abstract

Motivation Laboratory automation is thoroughly impacting biological research, transforming how experiments are conducted and accelerating scientific discovery (Huang et al, 2023). Our focus is on the pipette: the most utilized tool in biological research. Yet, current laboratory automation systems rely on rigid, preprogrammed parameters, demanding significant time and technical expertise. We address this limitation by developing an PPO RL system that learns optimal pipetting behaviors through trial-and-error and adapts to various pipetting tasks without manual reprogramming. This is the first application of PPO to pipetting. PPO's clipped-objective update provides stable, high precision control, but on policy training was previously infeasible due to the difficult of and cost of fluid simulation. The result show a proof of concept that self calibrating, reinforcement learning pipelines can replace hand tuned scripts and lower the barrier to lab automation hardware.

Method We develop a custom Gymnasium environment modeling pipetting in a 10×10×3 workspace. Liquid droplets are represented as spheres with realistic physics constraints. The pipette aspirates liquid when within 0.5 units of a droplet, height below 1.0, and plunger depth above 0.3. Dispensing occurs at target wells when height is below 1.5 and plunger depth is below 0.7. The action space is 4D continuous with values in [-1,1]. Actions control X, Y, Z movement with 0.2 scaling plus plunger depth. The observation space is a 14D vector containing pipette location and direction (4D), droplet positions (9D), and task progress (1D). We implement PPO actor-critic algorithm with full batch training. We sweep five PPO parameters: learning rate, training iterations, steps per epoch, entropy coefficient, and clip ratio as well as two task specific variables: target position and radius to add environmental variations and complexity to increase task difficulty.

Implementation We use PyTorch and Gymnasium for implementation. Our main architecture consists of a shared feature extractor. This feeds separate actor and critic networks. Each network has two 256-unit hidden layers with ReLU activations. The actor outputs continuous actions via tanh activation. The critic estimates state values. Training employs full-batch PPO with experience collection over varying steps per epoch. We implement standard PPO loss functions including policy loss, value loss, and entropy regularization.

Results Our agent successfully completed its goal of aspirating and dispensing 3 droplets, and learns and adapts to changes in its environment and tighter environmental constraints. In our optimization, our learning rate sweep, a higher value of 1e-3 achieved the highest episode success rate at 94.1 percent versus our baseline 80.2 percent, while extreme values such as 1e-5 resulted in zero success. Increasing the number of PPO training iterations per epoch from 20 to 30 improved success from 80.2 percent to 99.8 percent. Batch size analysis revealed that 1,000 steps per epoch was optimal, achieving 93.1 percent success, while smaller and larger batch sizes significantly underperformed. An entropy coefficient of 0.005 led to 90.5 percent success, compared to 51.1 percent with no entropy and 80.9 percent with a high coefficient of 0.2. A clip ratio of 0.1 outperformed other values with 89.9 percent success, while both 0.05 and 0.5 resulted in failure. In the environment difficulty and variation experiments, agents generalized well to different target well positions with success rates ranging from 88 to 99 percent. Reducing the target radius from 1.0 to 0.25 led to a drop from 88 to 79 percent success. The combined optimal configuration achieved a 99 percent success rate, representing a 24 point improvement over the baseline.

Discussion These results confirm that PPO can reliably learn pipetting behavior when properly tuned. Learning rate and training iterations had the greatest impact on performance, followed by batch size. Moderate entropy encouraged effective exploration, while extreme values degraded performance. Likewise, balanced clip ratios stabilized policy updates, with both overly strict and overly loose settings resulting in failure. The agent also generalized well to different target locations. While our simplified droplet model enabled fast, stable training, it omits key fluid properties such as viscosity, surface tension, and evaporation. These limitations reflect constraints in existing simulation environments, which lack scalable support for complex liquid dynamics. Future work will incorporate more realistic fluid mechanics, scale up the number of droplets, and explore curriculum learning to handle multi-step laboratory protocols. This will expand the applicability of reinforcement learning in lab automation beyond single-step pipetting tasks.

Conclusion This work presents the first successful application of PPO reinforcement learning to automated pipetting tasks. Through systematic hyperparameter optimization, we achieved up to 99 percent success rates in simplified liquid handling simulation. Our key contributions include development of a computationally tractable pipetting simulation environment. While our approach represents significant simplification of real-world challenges, it establishes foundation for accessible learning based laboratory automation. The hope is that this eventually will replace rigid, preprogrammed systems.

PPO Reinforcement Learning for Pipetting Laboratory Automation

Gurmenjit Kaur Bahia
Department of Biology
Stanford University
gurmenb@stanford.edu

Ashley Chen
Department of Computer Science
Stanford University
ashleyrc@stanford.edu

Abstract

Laboratory automation is advancing biology but is still held back by pipetting systems that need manual setup and rigid control. We show that Proximal Policy Optimization (PPO), a type of reinforcement learning, can train a system to pipette accurately through trial and error. We built a custom simulation environment where the agent learns to handle droplets in a 3D workspace using simplified suction and dispensing physics, with 14 inputs and 4 continuous action outputs. We trained it with a PPO actor-critic model, tuning five key training settings and two environment variables. The agent reached up to 99% accuracy, 24 points better than the baseline. Learning rate and number of training steps were most important, while entropy and clipping helped with stability. Though our model simplifies real fluid behavior, it allows fast training shows that reinforcement learning can help make lab automation more flexible and adaptive.

1 Introduction

Laboratory automation has accelerated biological research. It has completely transformed how experiments are conducted and accelerated scientific discovery (Huang et al, 2023). Automation has increased throughput, reduced human error, and enabled more complex experiments. Our focus in this project is automating the pipette. The pipette is the most used tool in bioscience research. Every day, researchers perform millions of pipetting operations. These happen across pharmaceutical development, diagnostic testing, and basic biological research. This widespread use makes the pipette an important priority for advancing lab automation.

Current laboratory automation systems have fundamental limitations. They lack adaptability and accessibility, which restricts their widespread adoption. Existing commercial platforms like Opentrons, Hamilton, and Tecan provide well-developed frameworks. However, they require substantial programming expertise for implementation and customization. These systems work only with pre-programmed parameters. Users must manually specify these parameters for each experimental protocol. Even minor protocol changes create problems. Switching between different microplate formats or accommodating new labware requires time-consuming recalibration. This creates significant barriers to adoption. The problem is especially severe for smaller laboratories without dedicated bioinformatics or automation engineering expertise.

The critical gap in these systems is their reliance on static programming rather than adaptive learning. These platforms excel at executing predetermined sequences. However, they cannot learn or adjust their behavior in real-time based on environmental feedback. They cannot adjust operational parameters like tip positioning, aspiration speeds, or approach angles. They cannot adapt to environmental conditions or labware variations. Instead, they require extensive

reagent-specific programming. They need constant human oversight whenever conditions deviate from pre-programmed specifications.

We address this limitation by developing a comprehensive reinforcement learning framework and seeking to optimize it's parameters for maximum performance. Our system enables robotic systems to learn complex pipetting behaviors autonomously through environmental interaction. Rather than hand coding every motion, our agent learns through trial-and-error. The agent learns to optimize aspiration and dispensing sequences. It learns pipette positioning strategies and plunger control policies through interaction with realistic laboratory environments.

Our research questions consist of: Can PPO successfully learn aspiration and dispensing behaviors from trial-and-error interaction within a realistic pipetting simulation and be optimized with hyperparameters?

We hypothesize that an on-policy reinforcement learning framework using Proximal Policy Optimization (PPO) can be optimized and learn accurate and adaptable pipetting behaviors in a simulated environment, allowing for more flexible and efficient automation than traditional rule-based or off-policy RL methods.

Our project serves as an important proof of concept for applying reinforcement learning to laboratory protocol automation to establish the foundation for future self-improving robotic systems. These systems can complete entire protocols simply by being able to pipette. This technology has immediate applications across multiple domains. It applies to high-throughput screening, clinical diagnostic testing, and pharmaceutical research protocols. In these areas, pipetting precision significantly impacts experimental outcomes and scientific reproducibility.

2 Related Works

The field of robotics has developed significant and truly revolutionary capabilities for manipulation tasks through reinforcement learning. Researchers, including our very own Professor Finn, at UC Berkeley made substantial contributions on deep reinforcement learning for hand-eye coordination in grasping tasks (Levine et al, 2016). Their approach integrated visual perception with motor control, allowing robots to learn grasping behaviors through extensive trial-and-error interactions with objects. Building on this foundation, researchers at Google brain developed QT-Opt, a vision-based system designed to adapt to novel objects. Their approach utilized a distributed reinforcement learning framework with massive amounts of data and off-policy training to achieve grasping capabilities in robotic systems (Kalashnikov et al, 2018)

However, these approaches focus primarily on rigid object interaction nowhere near simulating the complex systems, transport, and coordination involved in pipetting. Successful pipetting requires several coordinated actions that differ fundamentally from object manipulation. First, the pipette tip must be positioned accurately above liquid droplets. The plunger must then be retracted to generate suction for aspiration. Once droplets are aspirated, they must be transported together without falling. At the target location, the plunger extends in a controlled sequence to dispense each sample. This process demands very precise spatial navigation between multiple liquid targets while continuously managing the plunger's mechanical state.

Advanced laboratory automation systems have emerged to streamline experimental workflows. For example, Burger's mobile robotic chemist and King's autonomous scientist demonstrate capabilities in experiment planning and execution in the lab automation space (Burger et al., 2020; King et al., 2009). Yet, their underlying motion behaviors, including pipetting remain hard coded, lack the flexibility to adapt in real time to novel liquid configurations or environments. More directly related to pipetting, researchers at Carnegie Mellon applied actor-critic reinforcement learning using the IMPALA algorithm to automate liquid handling across complex 20×20 well plates (Ferdosi et al., 2023). This marked the first RL-based approach to pipetting, but it relies on massive off-policy data collection and large-scale infrastructure, making it impractical

for real time implementation in typical lab settings where compute and hardware resources are limited.

In contrast, Proximal Policy Optimization (PPO) has emerged as a widely used, stable policy gradient method for continuous control tasks (Schulman et al., 2017). PPO's clipped objective stabilizes training and prevents destabilizing updates, making it well-suited for robotics domains that demand fine-grained precision and robustness. Its sample efficiency and smooth control capabilities make it an ideal fit for pipetting, which requires reliable sequencing of aspiration and dispensing across variable targets. Additionally, researchers at Yale developed "spherical hands" that maintain object size invariant manipulation by aligning joint axes to a common point, enabling consistent control across object geometries (Ma et al., 2016). We drew inspiration from this system in designing our ball fluid pipetting simulation environment.

Our approach is completely novel. We present the first proof of concept for autonomous pipetting using on-policy reinforcement learning. As discussed above, previous methods have either depended on off-policy techniques like IMPALA or hard-coded motion sequences, both of which demand substantial computational resources and massive parallel data collection. We create a novel reinforcement learning framework that uses Proximal Policy Optimization (PPO) to learn pipetting behaviors through direct trial and error interaction, allowing for resource efficiency, flexiblity, and adaptive control. We hope our system will serve as a proof of concept of RL in lab automation and serve on a promising path for complex and complete biological lab robotic automation in the future.

3 Method

3.1 Environment and Simulation

We develop a custom Gymnasium environment to model pipetting in a $10 \times 10 \times 3$ unit workspace. Liquid droplets are represented as rigid spheres with simple collision and position tracking. The pipette is modeled as a cylinder whose end-effector position (x, y, z) and plunger depth p are fully controllable. Aspiration is triggered when the tip is within 0.5 units of a droplet, $z < 1.0$, and $p > 0.3$. Dispensing occurs when the tip is above the target well, $z < 1.5$, and $p < 0.7$. These thresholds approximate realistic laboratory tolerances based on empirical observations.

We used this design because representing droplets as spheres and using simple geometric thresholds greatly reduces computational overhead while preserving the core challenges of precise alignment, suction timing, and fluid transport. This balance enables rapid iteration on policy and hyperparameters without sacrificing task fidelity.

3.2 Action and Observation Spaces

Action space: A 4-dimensional continuous vector in $[-1, 1]^4$. The first three dimensions command incremental movements in x, y, z (scaled by 0.2 units/step). The fourth dimension controls plunger depth.

Observation space: A 14-dimensional vector

$$s = [\, x,\, y,\, z,\, p,\, x_1,\, y_1,\, z_1,\, x_2,\, y_2,\, z_2,\, x_3,\, y_3,\, z_3,\, f\,],$$

where (x, y, z, p) is the pipette location and plunger depth, (x_i, y_i, z_i) are the three droplet coordinates, and $f \in \{0, 1\}$ flags whether all droplets have been aspirated.

Our action space gives continuous control in $[-1, 1]$ allows fine-grained, smooth adjustments in both spatial movement and plunger actuation. Discrete actions would force coarse steps, impeding the precision required to avoid air bubbles or spillage.

Our observation space including full pipette positioning, droplet positions, and a task-progress flag provides complete situational awareness to our agent. The flag simplifies mode switching between aspirating and dispensing, ensuring the policy can segment behavior without inferring task stage from raw coordinates alone.

3.3 Reward Function

We define the step reward as:

$R_t = w_1\,\Delta\text{aspiration}_t + w_2\,\Delta\text{dispensing}_t + w_3\,\mathbf{1}_{\text{complete}} + w_4\,\text{guidance}_t - w_5\,\text{collision}_t - w_6\,\text{jerk}_t$

with weights: $w_1 = 10$, $w_2 = 20$, $w_3 = 50$, $w_4 = 1$, $w_5 = 5$, $w_6 = 0.5$.

Our Rationale for each Term in the Reward Function:

- Incremental aspiration and dispensing terms (w_1, w_2) provide dense learning signals, helping the agent make progress without relying on sparse rewards.
- Completion bonus (w_3) strongly encourages the agent to finish all required aspiration and dispensing actions.
- Guidance rewards (w_4) offer small directional incentives toward targets or sources, improving learning speed in early episodes.
- Collision penalties (w_5) discourage the agent from hitting well boundaries, protecting both physical integrity and task success.
- Jerk penalties (w_6) promote smooth pipette motion, which helps prevent liquid turbulence or spillage.

3.4 Learning Algorithm: PPO

We implement Proximal Policy Optimization (PPO) in an actor-critic framework. The advantage is estimated via Generalized Advantage Estimation (GAE):

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \big[r_{t+l} + \gamma V(s_{t+l+1}) - V(s_{t+l})\big],$$

with $\gamma = 0.99$, $\lambda = 0.95$. The clipped surrogate objective is

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\Big[\min\big(r_t(\theta)\,\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\,\hat{A}_t\big)\Big],$$

where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ and $\epsilon = 0.2$. The critic minimizes

$$L^{\text{VF}}(\phi) = \mathbb{E}_t\big[(V_\phi(s_t) - \hat{R}_t)^2\big],$$

with $\hat{R}_t = \hat{A}_t + V(s_t)$.

We chose to utilize PPO because its clipped updates constrain policy changes to a trusted region, preventing abrupt shifts that could cause the pipette to miss droplets or spill liquid. Compared to TRPO (which requires expensive second-order computations) or off-policy methods like DDPG/SAC (which can suffer from value bias), PPO offers stable, sample-efficient learning with minimal algorithmic complexity—ideal for precision laboratory tasks.

Both actor and critic are 2-layer MLPs with 256 units each and ReLU activations. We chose 256×256 as a trade-off: large enough to represent complex pipetting strategies, but small enough to train efficiently on 200 k frames. We set the clipping parameter $\epsilon = 0.2$, entropy coefficient 0.01, and GAE $\lambda = 0.95$ based on standard continuous-control defaults shown to work well in robotics domains.

For baseline, we used a learning rate of 3×10^{-4}, batch size 64, and perform 10 PPO epochs per data collection. These settings balance stability and sample efficiency, matching widely adopted PPO baselines.

We define an episode as successful if the agent transfers all three droplets to their target wells before reaching the 200-step limit. The `episode_success` metric is computed as the fraction of evaluation episodes in which this full transfer occurs.

4 Experimental Setup

To thoroughly evaluate the performance of our PPO-based reinforcement learning approach for automated pipetting, we designed an experimental framework that systematically tests various hyperparameters and environment configurations. Our goal was to understand how different algorithmic

choices affect the agent's ability to learn precise liquid handling skills while ensuring our findings would generalize beyond the specific training conditions.

Our experiment occured in our simulated pipetting environment where the objective of the agent was to learn to aspirate 3 droplets from a source locations and dispense them into target wells. Each episode allows a maximum of 200 steps, with the agent receiving rewards based on successful aspiration and dispensing actions. The environment uses a 10x10 unit workspace with wells having a radius of 1.0 unit and depth of 2.0 units at baseline.

Baselines and Metrics We compare our optimized agent against two baselines:

1. Default PPO: our initial PPO configuration before hyperparameter tuning (learning rate 3×10^{-4}, 20 update iterations, 2000 steps/epoch, entropy coefficient = 0.01, clip ratio = 0.2) with source at (2,2) and target at (8,8).
2. Negative control: the target well placed outside the workspace, which yields a 0% success rate.

Our primary metric is success rate, defined as the percentage of episodes in which all three droplets are transferred to their target well within 200 steps. Secondary metrics include the average episode reward and average episode length.

Success Criterion An episode is considered successful if all three droplets reach their designated target wells before the 200-step limit. We report success rate over 5 held-out evaluation episodes using fixed random seeds.

We organized our experiments into six main categories, each targeting a specific aspect of the PPO algorithm or environment configuration:

4.1 Learning Rate Testing:

We tested six different learning rates ranging from very conservative $(1{*}10^{-5})$ to aggressive $(3{*}10^{-3})$ values. Our baseline used $3 * 10^{-4}$, which is commonly recommended for PPO implementations. The extreme values $(1 * 10^{-5})$ and $(3 * 10^{-3})$ were included to observe potential underfitting and overfitting behaviors respectively.

4.2 Training Iteration Experiments:

These experiments varied the number of policy update iterations performed after each data collection phase. We tested values from 5 iterations up to 50 iterations, with our baseline at 20 iterations. Lower values reduce training time but may lead to insufficient learning, while higher values risk overfitting to collected data.

4.3 Steps Per Epoch Experiments:

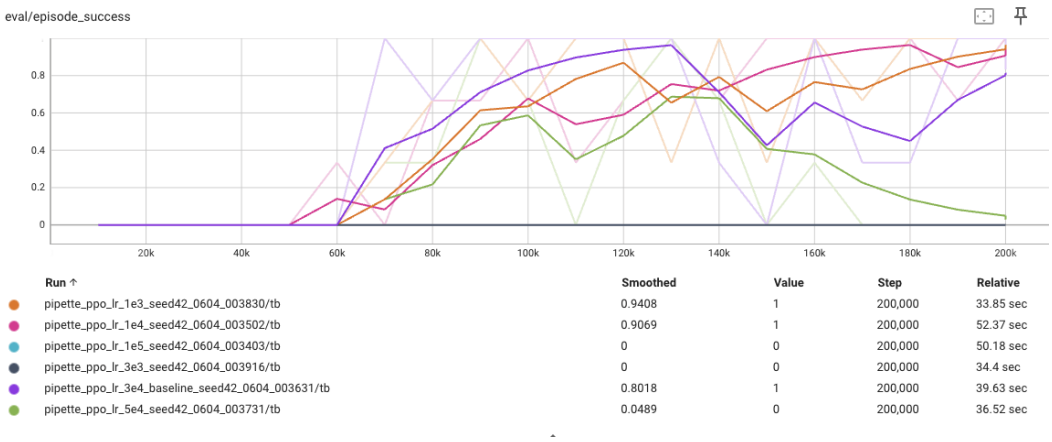
We examined how the amount of experience collected before each training update affects learning stability and sample efficiency. Our tests ranged from 500 to 8,000 steps per epoch, with 2,000 as our baseline. This parameter directly impacts the trade-off between learning stability and computational efficiency.

4.4 Entropy Coefficient Experiments:

To control the exploration-exploitation balance, we tested three entropy coefficient values: 0.0 (no exploration bonus), 0.005 (low exploration), and 0.020 (high exploration). This helps determine the optimal level of randomness needed for effective policy learning in our pipetting task.

4.5 Clipping Ratio Experiments:

The PPO clipping mechanism prevents destructively large policy updates. We tested conservative clipping (0.05, 0.10), standard clipping (0.20 baseline), and aggressive clipping (0.30, 0.50) to understand the sensitivity of our pipetting task to this crucial PPO hyperparameter.



Run	Smoothed	Value	Step	Relative
pipette_ppo_lr_1e3_seed42_0604_003830/tb	0.9408	1	200,000	33.85 sec
pipette_ppo_lr_1e4_seed42_0604_003502/tb	0.9069	1	200,000	52.37 sec
pipette_ppo_lr_1e5_seed42_0604_003403/tb	0	0	200,000	50.18 sec
pipette_ppo_lr_3e3_seed42_0604_003916/tb	0	0	200,000	34.4 sec
pipette_ppo_lr_3e4_baseline_seed42_0604_003631/tb	0.8018	1	200,000	39.63 sec
pipette_ppo_lr_5e4_seed42_0604_003731/tb	0.0489	0	200,000	36.52 sec

Figure 1: Learning Rate Sweep

4.6 Environment Testing:

Beyond hyperparameter tuning, we conducted additional experiments to ensure our algorithm was actually learning pipetting behaviors rather than exploiting environment-specific quirks. We modified the well radius in some experiments, testing smaller targets (0.25 and 0.5 units) to increase task difficulty and verify that the agent could adapt to more precise requirements. To verify that our agent learns genuine pipetting skills rather than memorizing specific locations, we trained models with target wells positioned at different coordinates: (8,12), (10,8), and (5,10). This tests the spatial generalization capabilities of our learned policies.

Each experiment was run with consistent random seeds to ensure reproducibility, and all training sessions used identical evaluation protocols. Every model was evaluated using the same set of test scenarios to enable fair comparison across different hyperparameter configurations. Training progress was monitored through success rates, episode rewards, and convergence stability metrics, with results logged using TensorBoard.

5 Results

Our hyperparameter study revealed significant insights into the optimal configuration for PPO-based pipetting automation. The results demonstrate that careful tuning of algorithmic parameters can dramatically improve task performance. These findings confirm that PPO can reliably learn pipetting behavior when appropriately tuned, with learning rate and training iterations having the strongest influence on final performance.

5.1 Learning Rate Analysis

The learning rate experiments showed a clear U-relationship between learning speed and final performance, as summarized in Table 1 and in Figure 1. Very small learning rates (1×10^{-5}) prevented any meaningful learning, achieving only 0% success. Moderate learning rates performed well, with 1×10^{-4} achieving 90.69% success and our baseline 3×10^{-4} reaching 80.18%. Surprisingly, the aggressive learning rate of 1×10^{-3} achieved the highest success rate at **94.08%**, suggesting that faster learning can be beneficial for this task when balanced properly. However, the most aggressive rate (3×10^{-3}) completely failed, likely due to destructive policy updates that prevented stable learning.

These results indicate that pipetting tasks benefit from moderately fast learning, possibly because the precise motor control required needs rapid adaptation to environmental feedback. The narrow window between optimal and destructive learning rates emphasizes the importance of careful tuning in robotic control applications and highlights the sensitivity of reinforcement learning algorithms to this fundamental hyperparameter.

Table 1: Learning Rate Experimental Results

Learning Rate	Success Rate (%)
1×10^{-5}	0
1×10^{-4}	90.69
3×10^{-4}	80.18
5×10^{-4}	4.89
1×10^{-3}	**94.08**
3×10^{-3}	0

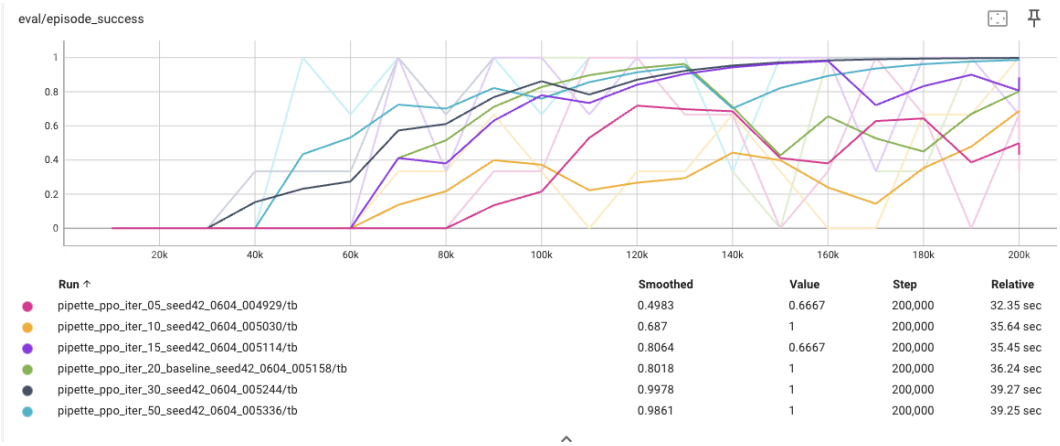


Figure 2: Training Iterations Sweep

5.2 Training Iterations Impact

The training iterations sweep revealed that more intensive training generally leads to better performance, as shown in Table 2 and Figure 2. Starting from 49.83% success with only 5 iterations, performance steadily improved as we increased training intensity. The peak performance of **99.78%** was achieved with 30 iterations, representing nearly perfect task execution. Interestingly, 50 iterations showed slightly lower performance (98.61%), suggesting potential overfitting when training becomes too intensive.

Table 2: Training Iterations Experimental Results

Training Iterations	Success Rate (%)
5	49.83
10	68.73
15	80.64
20	80.18
30	**99.78**
50	98.61

This pattern suggests that pipetting skills require substantial practice to develop but can be overtrained. The optimal range of 20-30 iterations provides sufficient learning without the risk of overfitting to specific training examples, which is deeply important for real-world deployment where conditions may vary. This finding aligns with general principles of machine learning where excessive training can lead to reduced generalization capability and overfitting.

5.3 Data Collection Efficiency

The steps per epoch experiments revealed an unexpected relationship between data collection and performance, as detailed in Table 3 and Figure 3. Rather than more data being better, we found that **1,000 steps per epoch achieved the highest success rate at 93.1%**. Performance actually decreased as we collected more data per training cycle, with 8,000 steps yielding only 43.37% success.

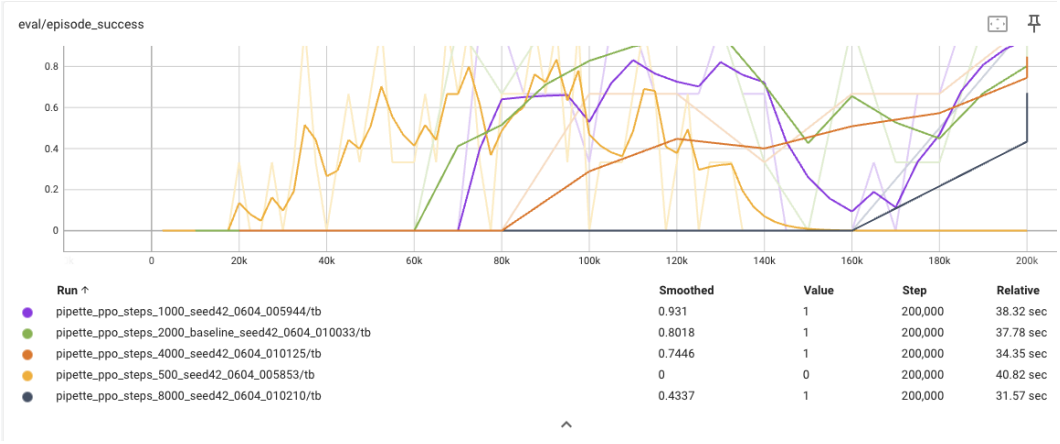


Figure 3: Steps Per Epoch Sweep

Table 3: Steps Per Epoch Experimental Results

Steps Per Epoch	Success Rate (%)
500	0
1000	**93.1**
2000	80.18
4000	74.46
8000	43.37

This result suggests that in more detailed motor control tasks like pipetting, fresher data may be more valuable than larger datasets. Smaller batch sizes might allow the algorithm to adapt more quickly to its improving policy, while larger batches could contain outdated experiences that interfere with learning.

5.4 Exploration Strategy Optimization

The entropy coefficient experiments examined how much randomness helps learning, with results presented in Table 4 and Fig 4. Complete elimination of exploration (entropy = 0.0) resulted in poor performance (51.09%), showing that some exploration is necessary. Low exploration (entropy =

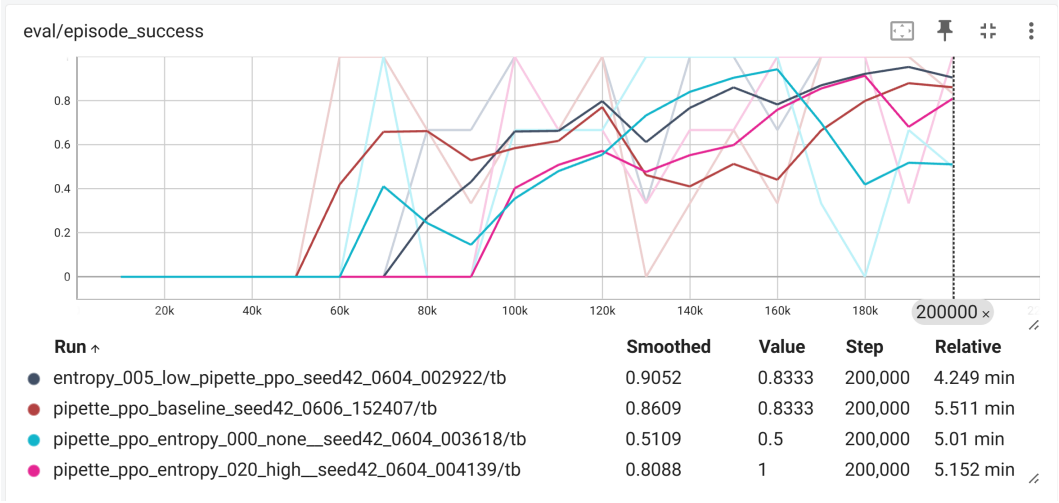


Figure 4: Entropy Coefficient Sweep

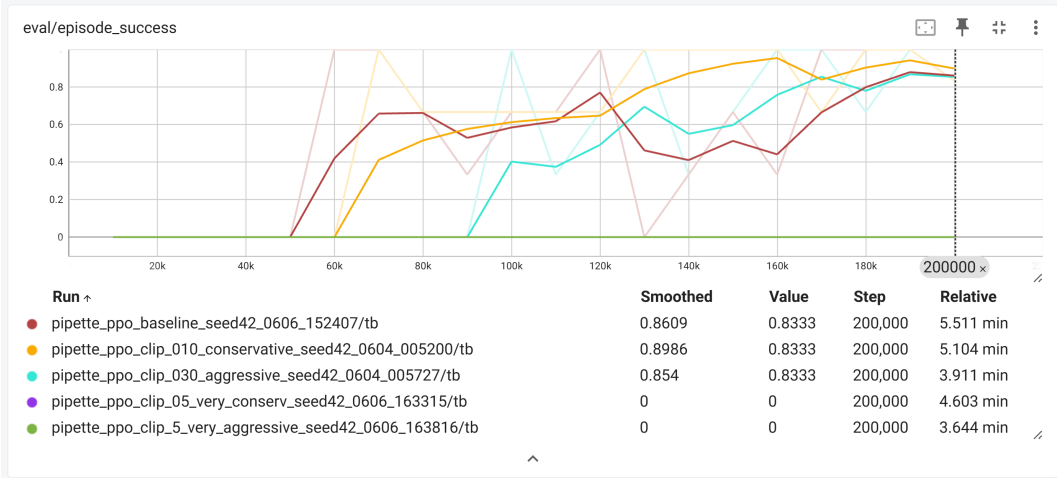


Figure 5: Clip Ratio Sweep

0.005) achieved the best results at **90.52%**, while our baseline moderate exploration (0.01) reached 86.09%. High exploration (0.02) decreased performance to 80.88%.

Table 4: Entropy Coefficient Experimental Results

Entropy Coefficient	Success Rate (%)
0.0	51.09
0.005	**90.52**
0.01 (baseline)	86.09
0.02	80.88

These results suggest that pipetting requires balancing between exploration and exploitation. Too little exploration prevents discovery of effective strategies, while too much exploration can interfere with late-stage control. Moderate entropy levels improved success by encouraging sufficient exploration, while extreme values led to under- or over-randomized policies. The optimal low-exploration strategy allows the agent to focus on precise movements while still discovering better techniques.

5.5 Clip Ratio Stability

The clipping ratio experiments showed that moderate constraints on policy updates works best for pipetting tasks, as summarized in Table5 and Figure 5. Very conservative clipping (0.05) failed completely, while slightly relaxed clipping (0.10) achieved **89.86%** success. Our baseline moderate clipping (0.20) performed well at 86.40%, and more aggressive clipping showed declining performance.

Table 5: Clip Ratio Experimental Results

Clip Ratio	Success Rate (%)
0.05	0
0.10	**89.86**
0.20 (baseline)	86.40
0.30	85.45
0.50	0

This pattern indicates that pipetting skills require steady but not overly cautious policy improvements. Appropriate clip ratios helped maintain stable updates, with both overly conservative and overly permissive values resulting in failure. Too much constraint prevents necessary learning, while too little constraint may lead to instability in the precise control required for liquid handling. This reinforces our choice of PPO as our algorithm- which values steady and stable progress.

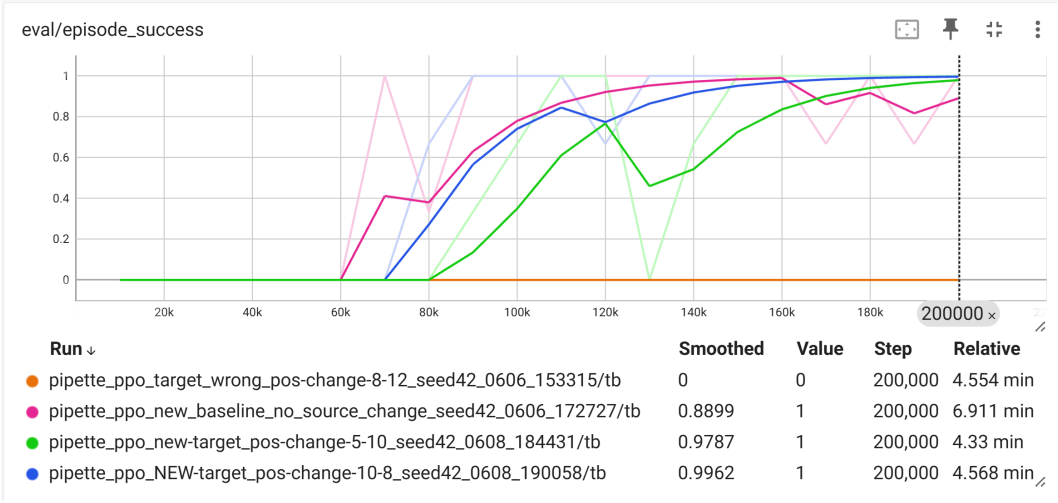


Figure 6: Target Position Changing Results

5.6 Spatial Generalization Capabilities

The target position experiments provided crucial validation that our agent learns genuine pipetting skills rather than memorizing specific locations, with results shown in Table 6 and Figure 6. Performance remained high across different target positions: 88% for (8,8), **99.48%** for (10,8), and 97% for (5,10). The one failure case (8,12) achieved 0% success because this position was outside the environment boundaries, confirming that our agent appropriately fails when given impossible tasks.

Table 6: Target Position Experimental Results

Target Position	Success Rate (%)
8, 8 (new baseline)	88
5, 10	97
10, 8	**99.48**
8, 12 (out of bounds)	0

These results demonstrate spatial generalization, indicating that the learned policies capture fundamental pipetting behaviors rather than location-specific movements. This is essential for real-world applications where target locations will vary and adaptability is crucial for practical deployment.

5.7 Precision Requirements

The radius experiments tested the agent's ability to handle different precision requirements, as detailed in Table 7 and Figure 7. Performance decreased as targets became smaller and more demanding: 82% success with 0.5-unit radius wells, 79.3% with 0.25-unit radius, compared to 88% with standard 1.0-unit wells.

Table 7: Target Radius Experimental Results

Radius	Success Rate (%)
0.25	79.3
0.5	**82**
1.0 (new baseline)	88

While performance declined with increased precision demands, the agent maintained reasonable success rates even with very small targets. This suggests that the learned policies can adapt to different precision requirements, though additional training might be needed for extremely demanding applications.

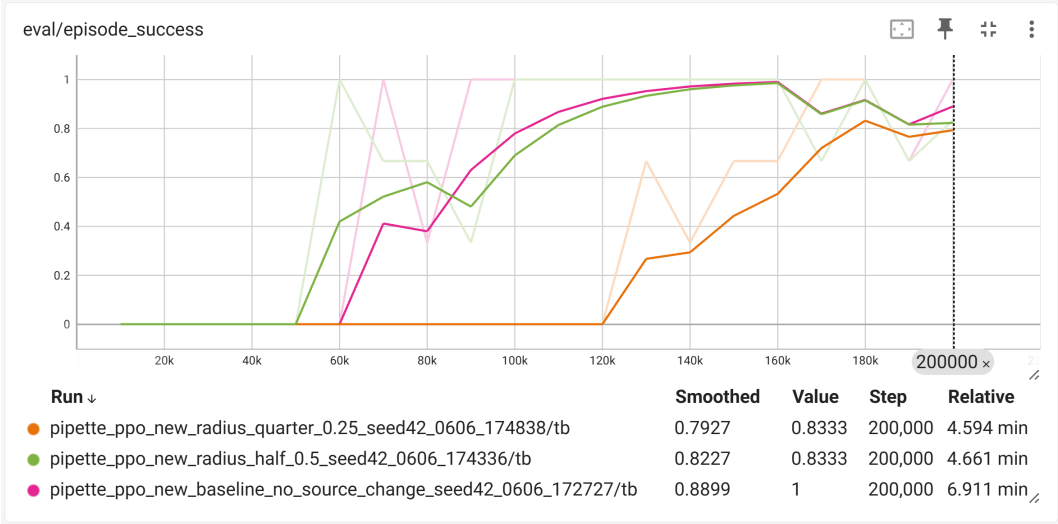


Figure 7: Target Radius Decrement Experiment Results

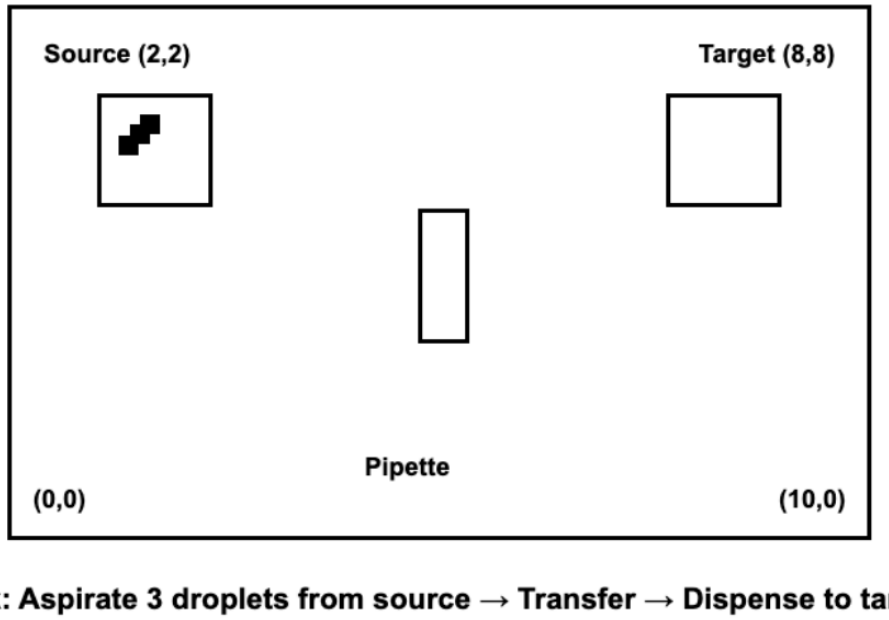


Figure 8: Pipette environment visual

5.8 Qualitative Analysis

We analyzed the agent's learned behavior through detailed examination of state trajectories and action patterns. The agent's internal state was tracked using two boolean arrays: `aspirated_droplets` and `droplets_in_target`, which provided insight into task progression and failure modes. We also modeled our simulated environment, which is shown in Figure 8.

The trained agent exhibited sophisticated movement patterns that emerged naturally from the reward structure. During the approach phase, the agent developed a characteristic spiral descent, maintaining altitude at approximately 2.5 units until positioned directly above the source well. This conservative strategy minimized collision risk while ensuring accurate positioning.

Most notably, the agent learned to aspirate droplets sequentially rather than simultaneously. A typical successful episode showed the following progression in the `aspirated_droplets` array:

```
Step 45: [False, False, False]
Step 52: [True, False, False]
Step 58: [True, True, False]
Step 64: [True, True, True]
```

This sequential pattern emerged despite equal reward weighting for all aspirations, suggesting the agent discovered that individual targeting improved reliability. The agent consistently followed a clockwise pattern around the source well, minimizing travel distance between droplets.

During transport, the agent maintained pretty good stability. Successful episodes showed the plunger depth locked at 0.85 (well above the 0.3 release threshold) throughout the entire transfer phase. The agent followed smooth arc trajectories rather than direct point-to-point paths, likely influenced by the jerk penalty.

The dispensing phase revealed unexpected sophistication. Rather than simultaneous release, the agent developed a staged dispensing strategy:

```
Step 143: [False, False, False]  # Plunger: 0.85
Step 148: [True, False, False]   # Plunger: 0.68
Step 151: [True, True, False]    # Plunger: 0.52
Step 155: [True, True, True]     # Plunger: 0.25
```

This behavior emerged without explicit reward shaping for sequential dispensing, indicating the agent discovered this approach improved success rates.

In terms of failure, analysis of failed episodes revealed three dominant patterns:

Incomplete Aspiration (40% of failures): The agent would successfully aspirate two droplets but fail to collect the third. These episodes typically ended with `aspirated_droplets = [True, True, False]` persisting until timeout. Video analysis showed this occurred when initial droplets were positioned on opposite sides of the well, making the third droplet difficult to reach without disturbing already-collected samples.

Transport Loss (35% of failures): Characterized by mid-episode regression in the aspiration array:

```
Step 85:  [True, True, True]   # Successful aspiration
Step 92:  [True, True, True]   # Normal transport
Step 98:  [True, True, False]  # Droplet lost
Step 103: [True, False, False] # Cascade failure
```

These failures correlated with rapid direction changes, suggesting the 0.5 weight on jerk penalty was insufficient for preventing aggressive maneuvers.

Dispensing Misalignment (25% of failures): The agent would successfully transport all droplets but fail during dispensing. Final states showed patterns like `droplets_in_target = [True, True, False]`, indicating the agent drifted outside the target well radius before completing the sequence.

Tracking array evolution across training episodes revealed clear skill development phases:

Episodes	Avg. Aspirated	Avg. Dispensed
1-1000	0.8	0.2
1000-5000	2.4	1.8
5000-10000	2.9	2.7
Final 1000	2.95	2.91

Table 8: Learning progression showing gradual mastery of task phases

These behavioral patterns provide valuable insights for designing more sophisticated laboratory automation systems, suggesting that complex manipulation strategies can emerge from relatively simple reward structures when properly configured.

6 Discussion

Our hypothesis was correct. Utimately, we were able to development an on-policy reinforcement learning framework using Proximal Policy Optimization (PPO) that can in fact learn accurate and adaptable pipetting behaviors in a simulated environment. Our hyperparameter analysis demonstrates that PPO can be greatly optimized. It can reliably learn complex pipetting behaviors when properly configured, but success depends critically on operating within narrow parameter ranges. Looking across our chosen parameters of learning rates, training iterations, batch sizes, entropy coefficients, and clipping ratios shows a clear ranking of parameter importance and provides practical guidance for deploying reinforcement learning in precision motor control tasks.

Learning rate and training iterations emerged as the most influential factors, with dramatic performance variations indicating their control over task completion. These parameters directly determine whether the agent can acquire the precise motor skills necessary for successful pipetting. Batch size showed the third-strongest impact, with smaller batches unexpectedly outperforming larger ones, challenging conventional assumptions about data efficiency in reinforcement learning.

Mid-level coefficients for entropy encouraged effective exploration while still ensuring that results remained precise. Extreme values in either direction heavily decreased performance. This finding highlights the necessity to balance exploration and exploitation in precision tasks. Balanced clipping ratios also stabilized policy updates and enabled consistent learning, while both overly strict settings (preventing necessary adjustments) and overly loose settings (allowing destructive policy changes) resulted in an inability to show any results.

The narrow optimization windows observed across multiple hyperparameters suggest that reinforcement learning for precise motor control operates differently versus many other RL applications. Unlike tasks where approximate solutions may be acceptable, pipetting requires exact positioning and timing, creating sharp performance cliffs around optimal parameter settings. This sensitivity has profound implications for practical deployment, indicating that automated hyperparameter tuning strategies will be essential for reliable implementation in laboratory environments.

6.1 Data Efficiency and Exploration-Exploitation Balance

Smaller batch sizes unexpectedly outperformed larger ones, suggesting that data freshness matters more than quantity in motor skill acquisition. As the agent's policy improves, older experiences can interfere with learning patterns that may require more accuracy.

Lower entropy coefficients worked better in task completion, which contrasts with many RL applications, where aggressive exploration benefits performance. This highlights the need to balance exploration with the deterministic precision required for accurate liquid handling based off of the environment used. However, higher entropy may work better in more complex environments and less calibrated hyperparameters.

6.2 Policy Generalization

The spatial generalization demonstrated across different target positions and radii provides evidence that agents learn genuine pipetting skills rather than memorizing specific movement sequences. Consistent high performance across varied spatial configurations suggests that learned policies capture fundamental principles of liquid handling rather than task-specific tricks. The policy's accurate reactions to out of bounds positions also demonstrate policy safety, which is crucial to developing safe laboratory apparatuses.

The radius experiments show an expected downward trend in task completion while also maintaining reasonable success rates, even with significantly smaller targets. This ability to scale indicates that agents are able to adapt to the varied accuracy demands, showing the agent's capability of learning and adapting to new environmental constraints.

6.3 Clipping & Algorithmic Stability

The clipping ratio experiments provide valuable insights into stability requirements for policy optimization in motor control tasks. Complete failure with very conservative clipping (0.05) suggests that

lower flexibility around our policy can prevent additional complex motor skills understanding, while failing at highly permissive settings (0.50) indicates that unrestricted policy changes can destroy previously learned motor patterns. Optimal performance at moderate clipping levels reflects the need for stable but meaningful policy evolution in precision tasks, where consistency in refining precise motor control patterns is essential.

6.4 Limitations and Future Directions

All training was conducted entirely in simulation using a simplified droplet model without fluid simulation or real-world data. The sphere-based liquid model provides computational efficiency but lacks important fluid properties like viscosity, surface tension, and evaporation that affect real pipetting operations. We were limited by simulation environments that didn't support complex liquid dynamics at our scale, thus requiring development of custom physics engines to simulate simplified suction and liquid-like behaviors.

Initial attempts to handle variable numbers of droplets (1, 2, or 3) within a single policy failed completely with 0% success rate. The agent consistently over-aspirated, attempting to pick up three droplets regardless of actual count, appearing to ignore droplet count information and defaulting to previously learned three-droplet behavior. This suggests that simply providing task variation as additional observation was insufficient for proper task conditioning.

Future work will focus on incorporating additional liquid properties to better capture fluid dynamics, exploring curriculum learning techniques to improve training efficiency, and developing more complex architectural approaches such as task-specific network layers or hierarchical policies for handling multiple related objectives. Our extremely high success rates show that a focus on increasing environment complexity is the direct next step in determining the limits of our algorithm. The development of more realistic simulation environments that capture complex fluid behaviors while maintaining computational tractability represents a significant engineering challenge. Additionally, transitioning from simulation to real-world deployment will require careful consideration of domain adaptation techniques, sensor integration, and safety protocols. Current simulation results provide a foundational knowledge for these developments, suggesting that RL-based approaches can master the precise control required for laboratory automation when configured and trained properly.

6.5 Broader Impact and Project Challenges

As mentioned throughout this paper, this work has broader implications for lower-entry, cost-efficient laboratory automation. By replacing hand-coded scripts with learning-based systems, we lower the barrier to entry for smaller laboratories without dedicated automation engineers. However, we must consider safety implications of autonomous liquid handling systems, particularly when handling hazardous materials and patient samples, which are often difficult to collect.

During the project, we encountered several significant challenges. The lack of existing fluid simulation support in standard RL environments forced us to develop custom physics approximations. Initial attempts with MuJoCo failed because it cannot model air pressure or fluid dynamics. We also struggled with training instability in early experiments, requiring extensive hyperparameter tuning to achieve stable learning. The most time-consuming aspect was debugging the reward function - early versions led to degenerate behaviors like the agent hovering indefinitely above droplets without attempting aspiration.

7 Conclusion

This work presents the first successful application of PPO reinforcement learning to automated pipetting tasks, achieving success rates of up to 99.78% through systematic hyperparameter optimization. Our experimental analysis revealed that optimal performance requires operation within narrow hyperparameter ranges, with learning rate and training iterations having the strongest influence on final performance. The counterintuitive finding that smaller training batches outperform larger ones challenges conventional assumptions about data efficiency in reinforcement learning.

The strong spatial generalization capabilities demonstrated across varied target positions provide evidence that our agents learn genuine pipetting skills. This finding is crucial for practical deployment, as laboratory automation systems must adapt to diverse experimental configurations and protocols.

Our key contributions include development of a computationally tractable pipetting simulation environment and systematic exploration of hyperparameter space that provides practical guidance for precision control tasks. While our approach represents significant simplification of real-world challenges, particularly regarding fluid dynamics, it establishes a crucial foundation for learning-based laboratory automation.

The narrow optimization windows identified across multiple hyperparameters highlight both the potential and challenges of deploying reinforcement learning in laboratory settings. Future work must address the transition from simplified simulation to real-world deployment through improved fluid modeling, sensor integration, and domain adaptation strategies.

This research establishes reinforcement learning as a viable approach for precision laboratory automation tasks and provides essential building blocks for replacing rigid, preprogrammed systems with adaptive, learning-based automation. The success achieved in these pipetting tasks suggest that stronger application of reinforcement learning towards laboratory automation is not only feasible but may be transformative for labs around the world.

8 Team Contributions

- **Ashley Chen:** Ashley was in charge of setting up the pipette environment, hyperparameter experiments, and worked on integrating the algorithm and the environment with Gurmen.
- **Gurmenjit Bahia:** Gurmen was in charge of the algorithm and reward function development, hyperparameter experimentation, and worked with Ashley on integrating the algorithm with the environment.

Changes from Proposal We pivoted from physical models to a completely simulated custom physics engine, as well as cut down on the parameters we would test on (and instead focused on tuning hyperparameters). These changes were made due to feasibility constraints around the programs we were using, as well as challenges around scoping our initial problem. For example, MuJoCo cannot accurately model fluid dynamics, nor can it model air pressure- two physics concepts that our initial objective relied on. Thus, we are now modeling liquids as individual small spheres- which are also easier to track and keep count of.

9 References

Burger, Benjamin, et al. "A Mobile Robotic Chemist." Nature, vol. 583, no. 7815, 2020, pp. 237-241.

Ferdosi, M., Ge, Y., & Kingsford, C. (2023). Reinforcement Learning for Robotic Liquid Handler Planning. arXiv preprint arXiv:2306.11536. https://arxiv.org/abs/2306.11536

Huang, J., Liu, H., Junginger, S., & Thurow, K. (2023). Mobile robots in automated laboratory workflows. SLAS Technology, 30, 100240. https://doi.org/10.1016/j.slast.2024.100240

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., & Levine, S. (2018). QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv. https://doi.org/10.48550/arXiv.1806.10293

King, Ross D., et al. "The Automation of Science." Science, vol. 324, no. 5923, 2009, pp. 85-89.

Levine, S., Finn, C., Darrell, T., & Abbeel, P. (2016). End-to-end training of deep visuomotor policies. The Journal of Machine Learning Research, 17(1), 1334-1373.

Ma, R. R., Rojas, N., & Dollar, A. M. (2016). Spherical hands: Toward underactuated, in-hand manipulation invariant to object size and grasp location. Journal of Mechanisms and Robotics, 8(6), 061021. https://doi.org/10.1115/1.4034787

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.