VascularSim: An Open-Source Platform for Reinforcement Learning–Based Microbot Navigation in Vascular Networks

Hass Dhia
Smart Technology Investments Research Institute
Los Angeles, CA
partners@smarttechinvest.com

Abstract

Surgical microbots that navigate the vasculature could transform targeted drug delivery, thrombectomy, and diagnostic procedures, yet no open-source simulation platform integrates vascular anatomy, multi-physics modeling, and standardized reinforcement learning (RL) interfaces. We present VascularSim, a modular Python framework that provides: (1) graph-based vascular anatomy representation with a TubeTK medical imaging data pipeline, (2) three Gymnasium-compatible RL environments with progressively richer physics, (3) analytical hemodynamic and magnetic field models, (4) a neural flow surrogate that accelerates repeated flow queries on fixed graphs, and (5) a benchmark suite across five difficulty tiers. On the default 30-node training environment, a Proximal Policy Optimization (PPO) agent achieves 99% navigation success with a path length ratio of 1.00 (computed over successful episodes), matching the shortest-path oracle and outperforming a random baseline (52% success), after 200k training steps completed in 67 seconds on a single CPU. Across the five benchmark tiers, random-agent success ranges from 10% (147-node dense mesh) to 50% (ring topology), establishing a difficulty gradient for algorithm comparison. VascularSim is released under the MIT license at `https://github.com/HassDhia/vascularsim` and is installable via `pip install vascularsim`.

1 Introduction

Surgical microbots—untethered robots at the micrometer scale—have the potential to transform interventional medicine by navigating the vasculature to deliver drugs, clear thrombi, or perform biopsies at targets inaccessible to conventional catheters [Nelson et al., 2010, Sitti, 2017]. Recent advances in magnetic actuation have produced clinically relevant demonstrations: Landers et al. demonstrated clinically ready magnetic microrobots navigating complex brain vasculature [Landers et al., 2025], and multiple groups have demonstrated in-vivo vascular traversal in animal models [Li et al., 2017, Erin et al., 2020].

Reinforcement learning (RL) has emerged as a leading approach for autonomous navigation in these complex environments. Karstensen et al. achieved 98% success rates with PPO for endovascular navigation [Karstensen et al., 2024], while Medany et al. demonstrated 90% sim-to-real success for ultrasound-driven microrobot control via model-based RL [Medany et al., 2025]. However, each group builds custom, single-use simulation code. No shared, open-source platform exists that combines vascular anatomy, hemodynamic physics, magnetic field modeling, and RL-compatible interfaces.

This gap imposes significant overhead on every new research effort and prevents reproducible comparison of navigation algorithms. Commercial tools such as COMSOL handle individual physics but are proprietary ($4k–50k), too slow for RL training (minutes per timestep vs. the microseconds required), and lack standardized RL interfaces.

We introduce VascularSim, an open-source simulation platform designed to fill this gap. Our contributions are:

1. A **graph-based vascular representation** with a complete data pipeline from TubeTK medical imaging datasets to navigable environments.

2. **Three Gymnasium environments** of increasing physics fidelity: base navigation, flow-aware navigation with hemodynamic observations, and magnetic-field-aware navigation with gradient force feedback.

3. **Analytical physics models**: Hagen–Poiseuille hemodynamics with Murray's law bifurcation distribution, Helmholtz coil magnetic fields with gradient-based force and torque, and a pure-

NumPy neural surrogate for real-time flow prediction.

4. A **benchmark suite** across five difficulty tiers (20–147 nodes) with standardized metrics for reproducible evaluation.
5. **PPO baselines** achieving 99% navigation success with optimal path efficiency, trainable on a single CPU in under 70 seconds.

VascularSim is released under the MIT license and installable via PyPI (`pip install vascularsim`). The full source, tests (139 tests, <2 s runtime), and documentation are available at `https://github.com/HassDhia/vascularsim`.

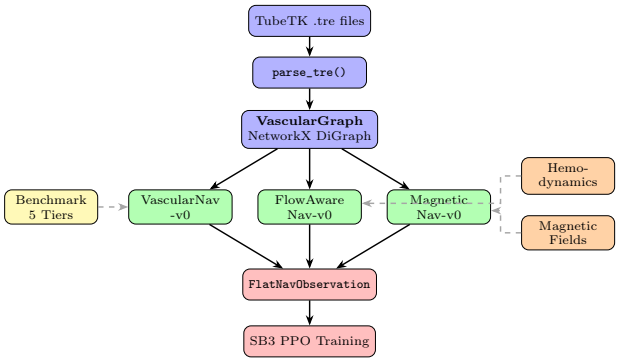

Figure 1: VascularSim system architecture. Solid arrows show data flow from medical imaging to RL training. Dashed arrows indicate physics integration into specialized environments.

2 Related Work

Endovascular navigation simulators. CathSim [Jianu et al., 2022] provides a PyBullet-based simulator for catheter navigation with SAC agents but targets millimeter-scale devices rather than microbots and does not model hemodynamics. The stEVE framework [Karstensen et al., 2024] demonstrated PPO-based endovascular navigation achieving 98% success rates but uses a custom non-reusable simulation backend. Neither provides standardized benchmark environments or multi-physics integration.

Surgical and hemodynamic simulation. The SOFA framework [Faure et al., 2012] provides a mature open-source platform for real-time surgical simulation with support for deformable tissue and fluid–structure interaction; however, it targets interactive surgical training rather than RL-compatible batch environments, lacks standardized Gymnasium interfaces, and its per-step latency (10–50 ms for coupled physics) exceeds the sub-millisecond budget required for RL training at scale. SimVascular [Updegrove et al., 2017] is the leading open-source cardiovascular simulation platform for patient-specific modeling. HemeLB [Mazzeo and Coveney, 2008] implements lattice Boltzmann methods for cerebrovascular hemodynamics. Both SimVascular and HemeLB are designed for offline high-fidelity computation and cannot provide the real-time performance (<1 ms per step) required for RL training.

RL for microrobot control. Yang et al. [Yang et al., 2022] applied hierarchical deep RL at cellular scale with red blood cell interactions. Jia et al. [Jia et al., 2025] introduced PINN-based flow-aware navigation. Medany et al. [Medany et al., 2025] demonstrated that model-based RL trained in simulation achieves 90% real-world success for autonomous microrobot navigation. These works use custom simulators that are not publicly available.

Medical imaging datasets. The TubeTK project [Aylward and Bullitt, 2002] provides 42 MRA volumes with pre-computed vessel centerlines and radii, making it ideal for graph-based vascular environments. IXI[1] provides 600 brain MRA scans, and TopCoW [Yang et al., 2023] offers Circle of Willis annotations.

VascularSim differentiates from prior work by integrating vascular anatomy loading, multi-physics modeling, standardized RL interfaces, and reproducible benchmarks into a single lightweight package with no GPU requirement.

3 System Architecture

VascularSim organizes into five modules: the TubeTK data pipeline, graph representation, RL environments, physics models, and benchmarking. Figure 1 illustrates the system architecture and data flow.

3.1 Vascular Graph Representation

The `VascularGraph` class wraps a NetworkX directed graph where each node stores a 3D position $\mathbf{p} \in \mathbb{R}^3$ and vessel radius $r > 0$, and each edge stores Euclidean length ℓ and mean radius $\bar{r}$. The `from_tubes()` factory method constructs the graph from `Tube` objects. Node IDs follow the format `"{tube_id}_{point_idx}"`, preserving the anatomical hierarchy of parent–child vessel relationships.

[1] `https://brain-development.org/ixi-dataset/`

For RL navigation, we construct an undirected view of the graph so that the agent can traverse edges in both directions, reflecting the physical bidirectionality of blood vessels.

3.2 TubeTK Data Pipeline

We implement a parser for the TubeTK MetaIO `.tre` format, which stores segmented vessel centerlines with per-point radii from MRA imaging. The parser extracts tube ID, parent ID, and (x, y, z, r) point data for each vessel segment, handling format variations across TubeTK releases. A download utility fetches sample data from the Kitware Girder API.

3.3 Gymnasium Environments

We provide three Gymnasium environments with progressively richer physics observations and reward shaping.

VascularNav-v0 is the base environment. The agent navigates node-to-node on the vascular graph, selecting from $k + 1$ discrete actions where k is the maximum node degree (the additional action is "stay in place"). The observation space is a dictionary containing the agent's 3D position, target position, node index, and Euclidean distance to target. The reward function combines:

$$r_t = r_{\text{time}} + r_{\text{progress}} + r_{\text{goal}} \tag{1}$$

where $r_{\text{time}} = -0.01$ (per-step penalty), $r_{\text{progress}} = (\|\mathbf{p}_{t-1} - \mathbf{g}\| - \|\mathbf{p}_t - \mathbf{g}\|)/d_0$ (normalized distance reduction), and $r_{\text{goal}} = +10.0$ upon reaching the target. Invalid or "stay" actions incur an additional -0.1 penalty. The environment initializes episodes with start–target pairs at least 5 graph hops apart when possible.

FlowAwareNav-v0 extends VascularNav-v0 with hemodynamic observations. The observation space adds: flow velocity at the current node (scalar) and flow velocities at neighboring edges (vector of size k). The reward includes a flow-alignment bonus: moving *with* the flow direction earns $+0.05$, while moving *against* incurs -0.05.

MagneticNav-v0 extends VascularNav-v0 with magnetic field observations from a configurable `CoilSystem`. The observation space adds: the B-field vector $\mathbf{B} \in \mathbb{R}^3$ and gradient force vector $\mathbf{F} \in \mathbb{R}^3$ at the agent's position. The reward includes a magnetic alignment term: $r_{\text{mag}} = 0.03 \cdot \cos\theta$, where θ is the angle between the agent's movement direction and the magnetic force vector.

3.4 Observation Flattening

A `FlatNavObservation` wrapper bridges the dictionary-based environment observations to the flat vector inputs required by standard RL libraries. It flattens each observation into a Box$(8,)$ vector $[\mathbf{p}_{\text{agent}}, \mathbf{p}_{\text{target}}, \hat{n}, d]$ suitable for Stable-Baselines3's `MlpPolicy`, enabling direct training without custom feature extractors.

4 Physics Models

4.1 Analytical Hemodynamics

We implement steady-state Hagen–Poiseuille flow with Murray's law for bifurcation distribution. For each edge with radius r and length L, the mean flow velocity is:

$$v = \frac{\Delta P \cdot r^2}{8\mu L} \tag{2}$$

where ΔP is the pressure drop across the edge and $\mu = 3.5 \times 10^{-3}\,\text{Pa·s}$ is the blood viscosity.

We distribute node pressures linearly from the inlet pressure $P_{\text{in}} = 13{,}332\,\text{Pa}$ (100 mmHg) to the outlet pressure $P_{\text{out}} = 2{,}666\,\text{Pa}$ (20 mmHg) based on topological distance from the graph root.

At bifurcation points, flow is distributed according to Murray's law:

$$Q_i \propto r_i^3 \tag{3}$$

where Q_i is the volumetric flow rate and r_i the radius of child branch i.

We compute wall shear stress as $\tau_w = 4\mu v/r$.

4.2 Magnetic Field Modeling

We model a three-axis Helmholtz coil system for magnetic microbot steering. The on-axis field of a single circular loop with n turns, current I, and radius R at axial distance z from center is:

$$B_z = \frac{\mu_0 n I R^2}{2(R^2 + z^2)^{3/2}} \tag{4}$$

The three-axis `CoilSystem` superimposes three orthogonal Helmholtz pairs (default $R = 50\,\text{mm}$, $I = 1.0\,\text{A}$, $n = 100$ turns per coil) to produce a controllable 3D field and gradient at any point in the workspace.

The gradient force on a magnetic microbot with moment $\mathbf{m}$ is $\mathbf{F} = (\nabla\mathbf{B})^\top \mathbf{m}$, and the alignment torque is $\boldsymbol{\tau} = \mathbf{m} \times \mathbf{B}$. The default magnetic moment $|\mathbf{m}| = 10^{-12}\,\text{A·m}^2$ corresponds to a typical $10\,\mu\text{m}$ iron oxide microbot.

Table 1: Benchmark tier specifications.

Tier	Name	Nodes	Topology
1	Straight	20	Linear vessel
2	Bifurcation	35	Single branch point
3	Tree	50	Two-level branching
4	Ring	43	Vessel loops
5	Dense Mesh	147	Mixed topology

4.3 Neural Flow Surrogate

For scenarios requiring faster-than-analytical flow prediction, we provide a pure-NumPy neural surrogate. The `FlowSurrogate` is a 2-hidden-layer MLP ($[64, 64]$ with ReLU activations and linear output) trained in log-space to handle the wide dynamic range (4+ orders of magnitude) of vascular flow velocities.

The input feature vector for each edge is $\mathbf{x} = [\bar{r},\ L,\ P_{\text{up}},\ P_{\text{down}},\ p_x,\ p_y,\ p_z]$, comprising mean radius, edge length, upstream/downstream pressures, and edge midpoint position.

Training uses He initialization, mini-batch SGD with learning rate 10^{-3}, and MSE loss on log-transformed velocities:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\left(\log(\hat{v}_i + \epsilon) - \log(v_i + \epsilon)\right)^2 \qquad (5)$$

The `from_graph()` factory method trains the surrogate end-to-end from a `VascularGraph`. On a 30-node test graph, the surrogate achieves $R^2 > 0.99$ against the analytical Hagen–Poiseuille ground truth after 200 training epochs, with mean absolute error below 1% of the velocity range. The surrogate's primary value is for scenarios with fixed graph topology where repeated forward queries dominate (e.g., multi-episode RL training on a single anatomy), amortizing the per-edge analytical computation into a single batched inference call. The pure-NumPy implementation ensures zero external dependencies beyond NumPy.

5 Benchmark Suite

We define five benchmark tiers of increasing topological complexity, summarized in Table 1. We procedurally generate each tier with a fixed random seed for reproducibility.

Tier 1 (Straight) consists of 20 nodes along a linear path, testing basic goal-directed navigation. **Tier 2 (Bifurcation)** adds a trunk of 15 nodes with two 10-node branches diverging at $\pm 30°$, requiring branch selection. **Tier 3 (Tree)** generates a three-level binary tree with randomized branch angles, testing multi-level decision-making. **Tier 4 (Ring)** introduces loops via upper and lower arcs connecting the trunk, requiring cycle handling. **Tier 5 (Dense Mesh)** creates a 20-node trunk with 8 primary branches, sub-branches, loop connections, and dead-end stubs, producing a 147-node graph that tests navigation in realistic vascular complexity.

The benchmark runner evaluates agents via a callable interface, collecting per-tier success rate, mean episode length, mean reward, and computation time per step. The runner exports results as JSON for reproducible comparison.

6 Experiments

6.1 Training Setup

We train a PPO agent [Schulman et al., 2017] on VascularNav-v0 using Stable-Baselines3 [Raffin et al., 2021]. The agent observes a flattened 8-dimensional vector (agent position, target position, normalized node index, distance to target) and selects from a discrete action space.

Hyperparameters: learning rate 3×10^{-4}, rollout length 2048 steps, minibatch size 64, 10 epochs per update, discount $\gamma = 0.99$, entropy coefficient 0.01. The default environment uses a 30-node graph (20-node trunk + 10-node branch) with a 500-step episode limit.

We compare against two baselines: (1) a **random agent** that samples uniformly from the action space, and (2) a **shortest-path oracle** that follows the NetworkX-computed shortest path at each step, representing the theoretical optimum for discrete graph navigation.

6.2 Training Dynamics

Figure 2 shows the mean episode reward during training. The agent transitions from exploratory behavior (negative reward, $\sim$4k episodes) to near-optimal navigation within approximately 80k timesteps, converging to a mean reward of $\sim$10.9 (close to the +10.0 goal reward minus minimal step penalties).

6.3 Agent Comparison

Table 2 compares all three agents on the default 30-node environment over 100 evaluation episodes.

The PPO agent achieves 99% success with a path length ratio of 1.00 (computed over successful episodes only; the single failure is excluded from this ratio), indicating it follows the shortest graph path in nearly all episodes. Its mean episode length (15.5 steps) is

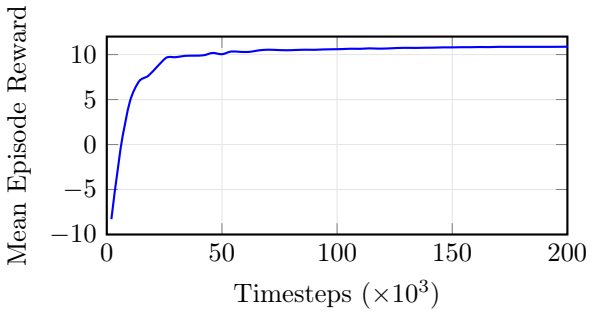

Figure 2: PPO training curve on VascularNav-v0 (200k steps, seed 42). Reward converges to ∼10.9 within 80k steps; evaluation reward (Table 2) is lower due to episode randomization.

Table 2: Agent comparison on VascularNav-v0 (100 episodes, seed 42).

Metric	PPO	Random	Oracle
Success rate	99%	52%	**100%**
Mean episode length	15.5	363.2	**10.6**
Mean reward	10.7	−16.8	**10.9**
Path length ratio	**1.00**	25.85	**1.00**

within 46% of the oracle (10.6 steps), with the gap attributable to the ∼5-hop minimum start–target distance randomization introducing variance in episode difficulty. The random baseline succeeds in only 52% of episodes, requiring ∼23× more steps than the PPO agent.

6.4 Cross-Tier Benchmark

Table 3 reports per-tier results for both the random agent and shortest-path oracle across all five benchmark tiers (50 episodes per tier). These results establish baseline performance curves for the benchmark suite.

Random agent success degrades sharply with topological complexity: from 42% on Tier 1 (linear) to 10% on Tier 5 (dense mesh with 147 nodes). The ring topology (Tier 4) is a notable exception at 50%, as loop structures provide multiple paths to each target. The oracle achieves 100% across all tiers with mean path lengths scaling from 8.8 to 17.3 steps—establishing the difficulty gradient that makes this benchmark suite useful for algorithm comparison.

6.5 Computational Performance

Table 4 reports simulation throughput.

Table 3: Per-tier benchmark results (50 episodes each, seed 42). Success rate (%) and mean steps.

		Random		Oracle	
Tier	Name	Succ.	Steps	Succ.	Steps
1	Straight	42%	395.8	100%	11.2
2	Bifurcation	18%	437.8	100%	10.0
3	Tree	12%	458.9	100%	15.2
4	Ring	50%	361.7	100%	8.8
5	Dense Mesh	10%	473.2	100%	17.3

Table 4: Simulation performance on Apple M-series CPU.

Metric	Value
PPO training throughput	3,007 steps/s
Total training time (200k steps)	66.5 s
Hemodynamics computation	<10 ms per graph
Magnetic field evaluation	<1 ms per point
Neural surrogate inference	<0.5 ms per batch
Test suite (139 tests)	<2 s total

The lightweight analytical physics models enable real-time RL training without GPU acceleration, a key design goal for accessibility to research groups without dedicated compute infrastructure.

7 Discussion

Design trade-offs. VascularSim deliberately uses analytical (closed-form) physics rather than CFD simulation. This trades fidelity for speed: Hagen–Poiseuille flow assumes steady-state, Newtonian, laminar flow in rigid tubes—assumptions that hold reasonably well in arteries above ∼100 μm diameter but break down at capillary scale where non-Newtonian effects and red blood cell interactions become significant. For RL algorithm development and benchmarking, this trade-off is appropriate: the simulation is physically grounded while maintaining the real-time performance (>1000 steps/s) required for policy optimization.

Extensibility. The modular architecture supports progressive physics refinement. The neural surrogate can be replaced with a physics-informed neural network (PINN) trained on high-fidelity CFD data. The magnetic model accepts arbitrary coil geometries. New environments can extend `VascularNavEnv` to add observations and reward terms without modifying the base implementation.

Limitations and sim-to-real gap. Three deliberate simplifications bound the current fidelity. First, navigation is discrete (node-to-node selection) rather than continuous position control; this precludes fine-grained steering maneuvers that real microbots require and limits the action space to the graph connectivity. Second, vessel walls are rigid—the model omits compliance, deformation, and wall interaction forces—which means that contact-dependent behaviors such as wall-following or vessel dilation are not captured. Third, the hemodynamic model assumes steady-state, Newtonian, laminar flow (Hagen–Poiseuille); it does not capture pulsatile cardiac cycles, non-Newtonian shear-thinning of blood, or red blood cell interactions that dominate below $\sim$100 μm diameter.

These simplifications widen the sim-to-real gap. Policies trained in VascularSim would require domain randomization, system identification, or sim-to-real transfer techniques (e.g., the progressive fidelity approach of Medany et al. [2025]) before deployment on physical microrobots. We view VascularSim as an algorithm development and benchmarking platform—not a deployment-ready digital twin—and the modular architecture is designed so that each physics module can be independently replaced with higher-fidelity alternatives as the field matures.

Additionally, the PPO results in Section 6 are evaluated only on the default 30-node environment; systematic per-tier PPO training across all five benchmark tiers remains future work.

8 Conclusion

We presented VascularSim, an open-source Python platform for training RL agents to navigate vascular networks with surgical microbots. The framework integrates graph-based anatomy, analytical hemodynamics and magnetic field models, a neural flow surrogate, three Gymnasium environments, and a five-tier benchmark suite into a single lightweight package.

PPO agents trained in VascularSim achieve 99% navigation success with optimal path efficiency, matching the shortest-path oracle and completing training in under 70 seconds on a single CPU. The five-tier benchmark suite establishes a difficulty gradient from 42% (random, Tier 1) to 10% (random, Tier 5), providing a meaningful evaluation framework for future navigation algorithms.

The MIT-licensed release via PyPI lowers the barrier to entry for vascular navigation research. Future work includes continuous position control, pulsatile hemodynamics, deformable vessel walls, PINN-based high-fidelity surrogates, swarm coordination environments, and patient-specific anatomy generation from clinical imaging data.

Availability. Source code, documentation, and pre-trained baselines are available at `https://github.com/HassDhia/vascularsim`. Install via `pip install vascularsim`.

References

Stephen R Aylward and Elizabeth Bullitt. Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction. *IEEE Transactions on Medical Imaging*, 21(2):61–75, 2002.

Onder Erin, Yusuf Emre Afsar, and Metin Sitti. Magnetic steering of magnetotactic bacteria for micromanipulation and drug delivery. *Advanced Intelligent Systems*, 2(9):2000066, 2020.

François Faure, Christian Duriez, Hervé Delingette, Jérémie Allard, Benjamin Gilles, Stéphanie Marchesseau, Hugo Talbot, Hadrien Courtecuisse, Guillaume Bousquet, Igor Peterlik, and Stéphane Cotin. SOFA: A multi-model framework for interactive physical simulation. In *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, pages 283–321. Springer, 2012. doi: 10.1007/8415_2012_125.

Yongyi Jia, Shu Miao, Jiayu Wu, Ming Yang, Chengzhi Hu, and Xiang Li. Flow-aware navigation of magnetic micro-robots in complex fluids via PINN-based prediction. *arXiv preprint arXiv:2503.11124*, 2025.

Tudor Jianu, Baoru Huang, Mohamed E. M. K. Abdelaziz, Minh Nhat Vu, Sebastiano Fichera, Chun-Yi Lee, Pierre Berthet-Rayne, and Anh Nguyen. CathSim: An open-source simulator for endovascular intervention. *arXiv preprint arXiv:2208.01455*, 2022.

Lennart Karstensen, Harry Robertshaw, Jacob Hatzl, Benjamin Jackson, Jörg Langejürgen, Katharina Breininger, Christian Uhl, S. M. Hadi Sadati, Thomas Booth, Christos Bergeles, and Franziska Mathis-Ullrich. Learning-based autonomous navigation, benchmark environments and simulation framework for endovascular interventions. *arXiv preprint arXiv:2410.01956*, 2024.

Fabian C. Landers, Laurin Hertle, Vitali Pustovalov, Disaiha Sivakumaran, Ceren Mutlu Oral, et al. Clin-

ically ready magnetic microrobots for targeted therapies. *Science*, 2025. doi: 10.1126/science.adx1708.

Jinxing Li, Berta Esteban-Fernández de Ávila, Wei Gao, Liangfang Zhang, and Joseph Wang. Micro/nanorobots for biomedicine: Delivery, surgery, sensing, and detoxification. *Science Robotics*, 2(4): eaam6431, 2017.

Marco D Mazzeo and Peter V Coveney. HemeLB: A high performance parallel lattice-Boltzmann code for large scale fluid flow in complex geometries. *Computer Physics Communications*, 178(12):894–914, 2008.

Mahmoud Medany, Lorenzo Piglia, Liam Achenbach, S. Karthik Mukkavilli, and Daniel Ahmed. Model-based reinforcement learning for ultrasound-driven autonomous microrobots. *Nature Machine Intelligence*, 2025. doi: 10.1038/s42256-025-01054-2.

Bradley J Nelson, Ioannis K Kaliakatsos, and Jake J Abbott. Microrobots for minimally invasive medicine. *Annual Review of Biomedical Engineering*, 12:55–85, 2010.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

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

Metin Sitti. *Mobile Microrobotics*. MIT Press, 2017.

Adam Updegrove, Nathan M Wilson, Jameson Merkow, Hongzhi Lan, Alison L Marsden, and Shawn C Shadden. SimVascular: An open source pipeline for cardiovascular simulation. *Annals of Biomedical Engineering*, 45(3):525–541, 2017. doi: 10.1007/s10439-016-1762-8.

Kaiyuan Yang, Fabio Musio, Yihui Ma, Norman Juchler, Johannes C. Paetzold, Rami Al-Maskari, Luciano Höher, Hongwei Bran Li, et al. Benchmarking the CoW with the TopCoW challenge: Topology-aware anatomical segmentation of the circle of Willis for CTA and MRA. *arXiv preprint arXiv:2312.17670*, 2023.

Yuguang Yang, Michael A. Bevan, and Bo Li. Hierarchical planning with deep reinforcement learning for 3D navigation of microrobots in blood vessels. *Advanced Intelligent Systems*, 4(11):2200168, 2022. doi: 10.1002/aisy.202200168.