



POLITECNICO
MILANO 1863

**SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE**

EXECUTIVE SUMMARY OF THE THESIS

Autonomous Endovascular Guidewire Navigation in a Simulated Unity Environment

LAUREA MAGISTRALE IN BIOMEDICAL ENGINEERING - INGEGNERIA BIOMEDICA

Author: ANDREA MONTALBANO

Advisor: PROF. ELENA DE MOMI, PROF. EMMANUEL VANDER POORTEN

Co-advisor: ANGELA PELOSO, MOULOUD OURAK

Academic year: 2025-2026

1. Introduction

Carotid Stenosis (CS) is defined as the narrowing or blockage of the carotid arteries. This condition is primarily caused by the accumulation of fatty deposits along the arterial walls, which progressively restricts blood flow. The Common Carotid Arteries (CCAs) constitute the primary vessels delivering blood to the brain and face, and represent critical anatomical landmarks in vascular, neurosurgical, and head-and-neck procedures. A reduction in blood flow due to CS can significantly increase the risk of cerebrovascular events. The global prevalence of significant CS (reduction in lumen diameter of $\geq 50\%$) is estimated to be approximately 1.2% in woman and 1.8% in men, with an annual incidence ranging from 2% to 6% [1]. Nowadays, the minimally invasive Transfemoral Carotid Artery Stenting (TFCAS) remains the most commonly used method for CS, especially considering that carotid endarterectomy is highly discouraged in patient with high carotid bifurcation, prior neck irradiation, or significant comorbidities. TFCAS requires catheter navigation from the femoral artery up to the carotid. This navigation technique can be limited by the tortuosity of the aortic arch, particularly in the presence of a type-3

aortic arch (according to Madwhal criteria [2]), which demands for advanced technical expertise and high procedural skill from the operator. Remote-controlled catheter by robotic technologies have been developed in order to mitigate limitations of manual navigation such as radiation exposure, precision, tremors, fatigue and added operator discomfort. Nevertheless, the outcome of the procedure is still limited by human error, and robotic systems' interfaces such as buttons and joysticks require skills that are different to those used in current clinical practice. In this context, the increasing procedural complexity and the need for enhanced precision and safety have motivated the development of autonomous navigation strategies for endovascular interventions. The objective of this thesis is to investigate AI-driven approaches for autonomous endovascular navigation across different anatomical scenarios through the development of a Unity-based simulation framework and to design the actuation behavior of the current robotic catheterization platform available at the Robot-Assisted Surgery Research Group of the Department of Mechanical Engineering of KU Leuven, Belgium.

2. State of the art

Different studies investigated the efficiency of AI-based model for autonomous endovascular navigation. Among them, only a limited subset specifically addresses the aortic arch, in most cases using Reinforcement Learning (RL). In RL, an agent interacts with an environment by observing its state and performing actions. Each action influences the environment, which returns feedback in the form of a reward signal. The objective of the agent is to learn an optimal policy that maximizes the expected cumulative reward over time. In [3], The Asynchronous Advantage Actor-Critic (A3C) algorithm has been trained in Unity Engine, while in [4], it was adopted in SOFA Framework and trained on a single modeled aortic arch. In [5], the Policy Improvement with Path Integrals (PI²) algorithm is employed to enhance the performance of Imitation Learning (IL) based on expert demonstrations across different pathological configurations of the same aortic arch. Finally, in [6], the Proximal Policy Optimization (PPO) algorithm in combination with Generative Adversarial Imitation Learning (GAIL) technique is explored for autonomous navigation within a type-1 aortic arch, then validated over a type-2 aortic arch. Despite these contributions, several limitations remain. First, to the best of the author's knowledge, none of the reviewed studies explicitly trains and evaluates autonomous navigation models on Type-3 aortic arch configurations, which are associated with increased anatomical complexity and procedural difficulty. Second, generalization across different aortic arch geometries remains largely unexplored. Only one study [7] considers training and validation across different arch types. Third, although Learning from Demonstration (LfD) techniques have been combined with RL, existing works in the vascular domain typically rely on a single demonstration-based strategy, while the combined use of multiple approaches, such as Behavioral Cloning (BC) and GAIL within the same framework, remains underexplored.

This thesis is positioned within these gaps, by (i) extending the evaluation to more complex anatomical configurations, including Type-3 aortic arches, (ii) investigating the generalization capabilities of the proposed approach across different anatomical scenarios within a unified

simulation framework, and (iii) exploring the combined use of multiple LfD techniques alongside reinforcement learning to improve navigation performance.

3. Materials and Methods

The simulation environment was developed using Unity Engine, version-6000.40f1, a real-time 3D development platform widely adopted for interactive simulations and robotics research.

3.1. Actuators Command Logic Implementation

A complete high-level actuation control logic of the robotic system currently available in the laboratory (Figure 1) was designed and implemented in Unity. The available catheterization system controls three endovascular instruments: the guidewire, the catheter, and the introducer sheath. Each instrument is actuated by a pair of units (guidewire by Active Unit C and D, catheter by Active Unit Y and Passive Unit B, introducer sheath by Active unit A and Passive Unit X). The active units can selectively grip or release the instrument and enable both translational and rotational motion (2 Degree Of Freedom (DOF)), whereas the passive units remain permanently gripped and allow only translational movement (1 DOF). The implemented system operational phases are:

- Calibration Mode: initialization of the parameters' value (desired displacement, desired rotation, anti-kinking distance) and of the units' state within predefined mechanical constraints.
- Surgery Mode: push/pull and rotation of each of the instruments loaded.

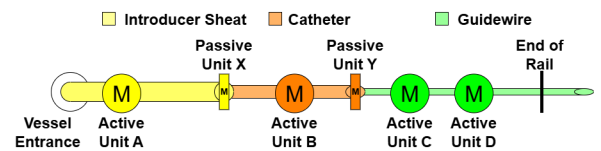


Figure 1: Illustration of the actuation modules of the laboratory robotic catheterization system.

3.2. Guidewire Modeling

Since the proposed framework focuses on the autonomous navigation of a guidewire within a vascular network, only the guidewire was modeled through a deformable representation, while

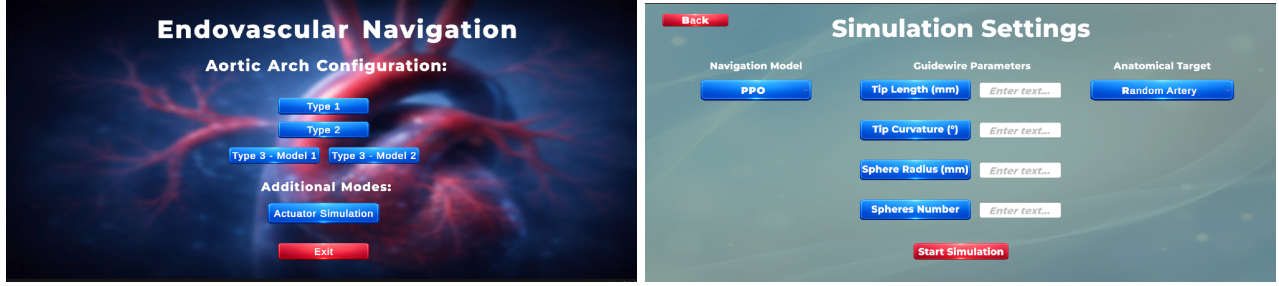


Figure 2: Graphical user interface of the proposed framework. Left: Main Menu; right: Simulation Settings panel.

the catheter and introducer sheath were approximated as rigid tubular structures. The guidewire was modeled as a chain of interconnected sphere GameObjects (Figure 3) embedded within four sections: embedded within four sections:

1. **Guidewire Core Sphere**, a high-mass sphere that receives translational and rotational commands and drives the motion of the entire system.
2. **Guidewire Tip**, modeled using the Piecewise Constant Curvature (PCC) approach.
3. **Guidewire Support Spheres**, placed before the tip to preserve realistic forward motion and prevent undesired bending toward the tip direction during advancement.
4. **Guidewire Length Visualization Spheres**, used exclusively for visualization purposes.

To enable realistic navigation and AI-model training, collisions between the vascular phantoms and the guidewire were enabled in simulation.

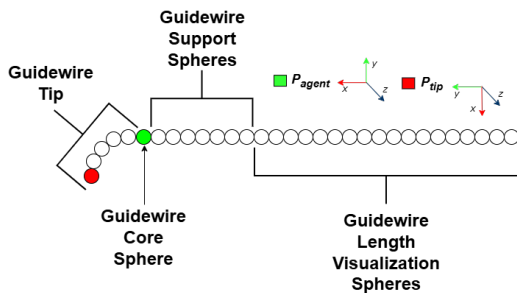


Figure 3: Guidewire design scheme.

3.3. Graphic User Interface (GUI)

The graphical user interface (Figure 2) provides an intuitive workflow for configuring and executing endovascular navigation tasks, including Aortic Arch Configuration selection (Figure 2, left), control mode choice (manual or au-

tonomous), and Simulation Settings (Figure 2, right), enabling the selection of the navigation model, the target anatomy, and the guidewire geometry. An additional mode enables the Actuator Simulation (Figure 2, left), in which it is possible to test the implemented actuator control logic.

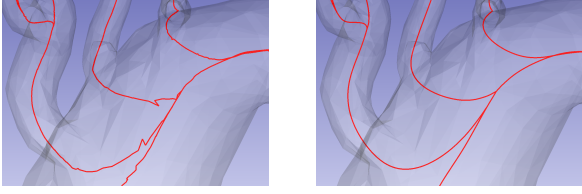
3.4. Anatomical Dataset

In total, four different anatomical models acquired from CT scans (three models were extracted from open-source datasets, while one model was provided by the laboratory) and imported into the Unity environment were used for training and validation. According to the Madhwal classification, the dataset included one type-1, one type-2, and two type-3 aortic arch models. Type-1, type-2, and type-3 arches correspond to increasing distances between the brachiocephalic artery origins and the top of the aortic arch.

3.5. Graph-Based Path Planning

On every anatomical model, graph-based path planning has been performed. The proposed path planning pipeline consists of three main stages: (i) centerline extraction and post-processing, (ii) graph generation from the extracted centerline network, and (iii) graph-based path generation between the selected start and target locations. Firstly, centerline extraction has been conducted through the "Extract Centerline" module available in 3D Slicer. The extracted centerlines were further processed (Figure 4) in Python to reduce high-frequency geometric artifacts. Each branch was resampled with a uniform spacing of 0.5 mm and smoothed using a cubic parametric spline interpolation. Subsequently, a moving average filter with a window size of 9 points was applied to further reduce local fluctuations while preserving the

branch endpoints and the connectivity at vessel bifurcations.



(a) Pre-processed center-lines with 3D Slicer only. (b) Post-processed center-lines with Python library.

Figure 4: Center-lines post-processing effect.

Based on the centerlines of the vascular model, a graph was generated to explicitly represent the connectivity between different branches and to enable the application of graph-based path planning algorithms. By converting the center-line network into a graph, the navigation problem was reformulated as a path planning task on a discrete topological structure (the graph itself). This representation allows the use of the Dijkstra algorithm, which computes the path of minimum total length between two given nodes P and Q within a weighted graph. The generated path provide continuous geometric guidance during the navigation of the agent. The overall workflow of the proposed framework is illustrated in Figure 5.

3.6. AI model

Proximal Policy Optimization (PPO) and Soft Actor-Critic (SAC) have been employed to learn navigation policies through interaction with the simulated environment. PPO is an on-policy reinforcement learning algorithm that improves policy stability by constraining updates through a clipped surrogate objective:

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

where $r_t(\theta)$ is the ratio between the probability of selecting action a_t under the updated policy and the probability of selecting the same action under the previous policy, θ represents the set of neural network weights that define the agent's policy, $\hat{A}_t$ is the advantage estimate, measuring how much better or worse the selected action a_t is compared to the expected value of the current state s_t , $\text{clip}(\cdot)$ is the clipping operator and ϵ is the clipping parameter controlling the maximum allowed deviation between the old and the updated policy. In contrast, SAC is an off-policy reinforcement learning algorithm based on the maximum entropy framework, which promotes exploration by maximizing both the expected

cumulative reward and the entropy of the policy:

$$J(\theta) = E_{\tau \sim p_{\theta}(\tau)} \left[\sum_t r(s_t, a_t) + \alpha \mathcal{H}(\pi_{\theta}(\cdot | s_t)) \right], \quad (2)$$

where α controls the trade-off between reward maximization and policy stochasticity, $r(s_t, a_t)$ represents the reward received by the agent after taking action a_t in state s_t and $\mathcal{H}(\pi(\cdot | s_t))$ denotes the entropy of the policy π at state s_t , which measures the randomness of the action distribution. BC and GAIL were introduced to accelerate the learning process and improve initial policy performance by leveraging demonstration data. In particular, these LfD techniques allow the agent to benefit from prior knowledge, reducing the exploration burden typically required by reinforcement learning alone. BC formulates the problem as supervised learning, directly mapping states to expert actions, whereas GAIL adopts an adversarial framework to learn a policy that reproduces the expert behavior distribution through an implicit reward signal.

3.6.1 Observations

The observation space (Figure 6) consists of 34 features describing the state of the guidewire and its relation to the planned trajectory. All quantities are expressed in the local reference frame of the agent through the inverse rotation matrix $\mathbf{R}^{-1} \in R^{3 \times 3}$. The local position of the guidewire tip (3 observations) is computed as $\mathbf{o}_{\text{tip, pos}} = \mathbf{R}^{-1}(\mathbf{p}_{\text{tip}} - \mathbf{p}_{\text{agent}})$. The local orientation of the guidewire tip (3 observations) is represented as $\mathbf{o}_{\text{curve}} = \mathbf{R}^{-1} \mathbf{d}_{\text{curve}}$, where $\mathbf{d}_{\text{curve}} \in R^{3 \times 1}$ denotes the local axis of the guidewire tip used to represent its bending direction. The observation associated with the local target point (3 observations) is computed as $\mathbf{o}_{\text{target}} = \mathbf{R}^{-1}(\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{closest}})$, where $\mathbf{p}_{\text{closest}}$ denotes the closest point along the reference trajectory to the guidewire tip, and $\mathbf{p}_{\text{target}}$ denotes a target point sampled 4 mm ahead along the path and used to guide local navigation. The future sampled points along the trajectory (15 observations) are defined as $\mathbf{o}_{\text{future}, k} = \mathbf{R}^{-1}(\mathbf{p}_k - \mathbf{p}_{\text{closest}})$, where $\mathbf{p}_k$ denotes the k -th future point sampled at increasing arc-length distances after the local target point, with $k \in \{1, \dots, 5\}$. The local tangent direction of the trajectory (3 observations) is represented as:

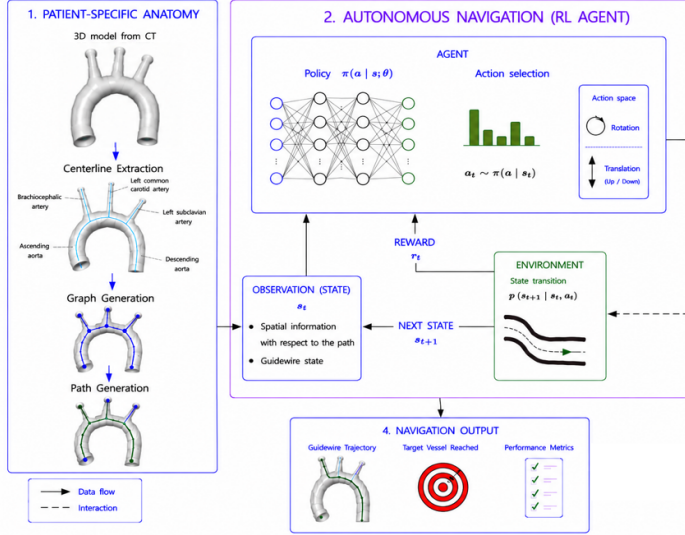


Figure 5: Block diagram of the proposed navigation pipeline.

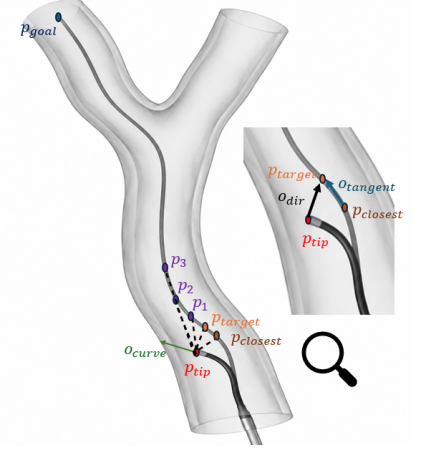


Figure 6: Graphical representation of the observation space.

$$\mathbf{o}_{\text{tangent}} = \mathbf{R}^{-1} \left(\frac{\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{closest}}}{\|\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{closest}}\|} \right) \quad (3)$$

The final navigation goal (3 observations) is represented as $\mathbf{o}_{\text{goal}} = \mathbf{R}^{-1}(\mathbf{p}_{\text{goal}} - \mathbf{p}_{\text{closest}})$.

Finally, the normalized direction vector from the guidewire tip to the local target point (3 observations) and the Euclidean distance between them (1 observation) are computed as:

$$\mathbf{o}_{\text{dir}} = \mathbf{R}^{-1} \left(\frac{\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{tip}}}{\|\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{tip}}\|} \right), \quad o_{\text{dist}} = \|\mathbf{p}_{\text{target}} - \mathbf{p}_{\text{tip}}\| \quad (4)$$

3.6.2 Reward Function

The reward function is designed to promote progress along the path, alignment with the centerline, and efficient task completion. The values reported below are based on a trial and error approach. At each time step:

- A reward of +0.0007 is assigned if the agent makes forward progress along the path, otherwise a penalty of -0.0002 is applied.
- Milestone rewards are introduced to guide learning: +0.2, +0.5, +1, and +3 are assigned when reaching $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$, and the final target, respectively.
- A penalty proportional to the distance from the path is applied as $-0.0001 \cdot \text{DistanceToPath}$ to enforce alignment with the centerline.
- A constant time penalty of -0.0003 per step is added to encourage shorter trajectories.
- Unsuccessful episodes are terminated with a reward of -5 if no progress is observed for more than 5000 steps.

- A penalty in the range $[-5, -2]$ is assigned, and the episode is terminated, according to:

$$R_{\text{path}} = \begin{cases} -5 + 3 \frac{s_{\text{tip}}}{L_{\text{tot}}}, & \text{if } \|\mathbf{p}_{\text{tip}} - \mathbf{p}_{\text{closest}}\| > 0.015 \text{ m} \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where R_{path} denotes the penalty associated with excessive deviation from the planned path, s_{tip} denotes the curvilinear abscissa of the guidewire tip along the path, and L_{tot} denotes the total length of the planned path.

4. Experimental Set-up

4.1. Training Phase

The training was performed using the Unity ML-Agents package. The task of the simulation is to correctly perform a guidewire autonomous over-arching of the aortic arch without any need for external support inside the Unity simulation environment. During the training phase, the learning agents were trained exclusively on the type-3 aortic arch configuration, representing the most challenging anatomical scenario due to its increased tortuosity and geometric complexity. This choice was made to encourage robust policy learning under demanding navigation conditions. The path of the task is chosen accordingly to the standard TFCAS procedural workflow, following the clinical navigation route from the descending aorta toward the supra-aortic target vessels (Figure 7). BC and GAIL were employed both individually and in combination, and integrated with reinforcement learning algorithms PPO or SAC to evaluate their contribution to

navigation performance. Both approaches were trained using a dataset of 25 demonstrations collected from a non-expert operator within the simulation environment through the use of a joystick, providing a baseline behavior that the agent can imitate and refine during training. The combination of trained algorithms were: 1) PPO, 2) PPO+BC, 3) PPO+GAIL, 4) PPO+BC+GAIL, 5) SAC+BC, 6) SAC+GAIL, 7) SAC+BC+GAIL.

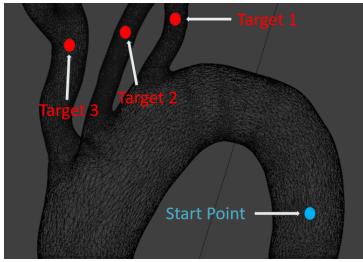


Figure 7: Illustration of the navigation task in the aortic arch anatomy. The blue point represents the start position located in the descending aorta, while the red points indicate the target locations positioned at the outlets of the left subclavian artery (Target 1), left common carotid artery (Target 2), and brachiocephalic artery (Target 3).

4.2. Validation Phase

The validation phase was conducted on type-1, type-2 and type-3 aortic arch configurations, which were not encountered during training. This evaluation aimed to assess the generalization capability of the learned policies across anatomically distinct vascular geometries. For each aortic arch configuration, 20 evaluation episodes were performed for each of the three navigation targets (Figure 7): the left subclavian artery (LSC), the left common carotid artery (LCCA), and the brachiocephalic artery (BCA).

4.3. Evaluation Metrics

The performance of the proposed reinforcement learning agents is evaluated using quantitative metrics that assess both learning behavior and task efficiency:

- **Average Cumulative Reward:** provides insight into the optimization dynamics of the learning process:

$$\bar{R} = \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=0}^{T_i} r_t^{(i)} \right) \quad (6)$$

where $r_t^{(i)}$ denotes the reward received at timestep t during episode i , T_i represents the duration of episode i , and N is the number of considered episodes.

- **Episode Length:** defined as the number of steps required to complete an episode, is analyzed as an indicator of navigation efficiency and trajectory optimization.
- **Success Rate:** the percentage of episodes in which the agent successfully reaches the target branch.
- **Procedure Time:** the total elapsed simulation time required for the agent to successfully reach the desired vascular branch.

Statistical comparisons between models are evaluated with respect to the PPO baseline and performed using the **Mann–Whitney U-test** for procedure time and **Fisher’s exact test** for success rate, with Holm correction applied to control for multiple comparisons.

5. Results and Discussion

The performance metrics of the trained algorithms are summarized in Table 1. SAC-based models are not included in the quantitative evaluation, as they did not converge during training and therefore did not produce reliable performance metrics, as shown in Figure ???. The failure of SAC to converge is likely due to its sensitivity to hyperparameter tuning and the difficulty of learning stable value estimates in a structured and constrained navigation task with complex reward shaping. As it regards PPO-based models, the inclusion of BC leads to a clear improvement in simpler anatomies (Type 1), where PPO+BC achieves 100% success with statistical significance ($p = 0.018$) compared to the PPO baseline. For more complex configurations (Type 2 and Type 3), the success rate remains comparable across methods (except for PPO+GAIL), indicating that BC does not degrade performance but does not consistently provide further gains. Regarding procedure time, the results are more heterogeneous. In multiple cases (5/9 targets), the PPO baseline achieves lower completion times and shows statistically significant differences ($p < 0.001$) respect to PPO+BC, indicating that the introduction of BC may lead to slower trajectories despite maintaining high success rates. On the other hand, PPO+BC+GAIL often provides a

Aortic Arch	Model	Success Rate (SR) (%)	SR p-value (Fisher)	Target 1		Target 2		Target 3	
				Time(s) (Mean±Std)	p-value (MWU)	Time(s) (Mean±Std)	p-value (MWU)	Time(s) (Mean±Std)	p-value (MWU)
Type 1	PPO	86.66	/	5.47 ± 1.47	/	9.36 ± 0.89	/	11.18 ± 1.37	/
	PPO+BC	100	0.018*	6.23 ± 0.72	<0.001***	10.88 ± 1.21	<0.001***	10.36 ± 0.30	0.551
	PPO+GAIL	60	0.006**	9.28 ± 1.67	<0.001***	29.39 ± 5.62	<0.001***	43.57 ± 0.66	0.065
	PPO+BC+GAIL	88.33	1	5.44 ± 0.89	0.320	10.36 ± 0.84	<0.001***	11.28 ± 1.23	0.826
Type 2	PPO	95	/	10.60 ± 3.19	/	12.79 ± 3.03	/	13.89 ± 0.79	/
	PPO+BC	93.33	1	9.33 ± 0.57	0.020*	12.93 ± 0.94	0.095	13.65 ± 0.94	0.546
	PPO+GAIL	70	0.003**	33.17 ± 10.90	<0.001***	29.01 ± 5.49	<0.001***	21.85 ± 1.99	<0.001***
	PPO+BC+GAIL	93.33	1	9.66 ± 0.99	0.093	13.61 ± 2.49	0.095	13.91 ± 0.77	0.903
Type 3	PPO	98.33	/	7.20 ± 0.13	/	10.46 ± 1.41	/	14.61 ± 1.65	/
	PPO+BC	98.33	1	8.93 ± 0.59	<0.001***	12.36 ± 0.62	<0.001***	19.95 ± 0.60	<0.001***
	PPO+GAIL	88.33	0.367	20.94 ± 2.80	<0.001***	42.85 ± 8.02	<0.001***	57.81 ± 23.03	<0.001***
	PPO+BC+GAIL	98.33	1	7.57 ± 0.19	<0.001***	11.13 ± 0.37	<0.001***	15.52 ± 0.93	0.152

Table 1: Summary of model performance across different aortic arch configurations. Significance levels are indicated as follows: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$, while "/" denotes the baseline reference PPO. Target 1: cannulation in the left subclavian artery; Target 2: cannulation in the left common carotid artery; Target 3: cannulation in the brachiocephalic artery.

more balanced behavior, with execution times closer to or slightly higher than PPO (only 3/9 targets show significant differences $p < 0.001$), suggesting a trade-off between stability and efficiency. In contrast, PPO+GAIL consistently exhibits inferior performance, with significantly higher procedure times and statistically significant differences (9/9 targets $p < 0.001$) across all anatomical types and targets, as well as reduced success rates in simpler scenarios. This indicates that GAIL alone does not provide a reliable learning signal in this setting and may negatively affect both efficiency and robustness. This behavior is consistent with the characteristics of the considered methods. While BC can provide stable improvements even with limited and suboptimal demonstrations, GAIL is more sensitive to the quality and diversity of the dataset, and may lead to degraded performance when the demonstrations do not accurately represent the desired behavior (non-expert demonstrations). Overall, despite these differences, all PPO-based models demonstrate strong performance in completing the navigation tasks, and consistent results across different anatomical configurations suggest a good level of generalization of the learned policies.

6. Conclusion and future work

This work demonstrates the feasibility of combining reinforcement learning with a physics-based simulation framework for autonomous endovascular navigation. The proposed framework establishes a flexible and scalable foundation that can be further extended and integrated with more complex models and real-world ap-

plications. In this context, future developments would move in two directions:

- Improving the physical and procedural realism of the simulation environment by including hemodynamic effects, incorporating deformable vessel walls and improving the mechanical properties of the guidewire;
- Transition from simulation to real-world applications by transferring the learned policies to physical setups, evaluating their performance in controlled laboratory conditions, and addressing the discrepancies between simulation and real-world behavior.

References

- [1] Behr et al. Deep reinforcement learning for the navigation of neurovascular catheters. 2019.
- [2] Madhwal et al. Predictors of difficult carotid stenting as determined by aortic arch angiography. 2008.
- [3] Meng et al. Evaluation of a reinforcement learning algorithm for vascular intervention surgery. 2021.
- [4] Meng et al. Evaluation of an autonomous navigation method for vascular interventional surgery in virtual environment. 2022.
- [5] Chi et al. Trajectory optimization of robot-assisted endovascular catheterization with reinforcement learning. 2018.
- [6] Rafii-Tari et al. Learning-based modeling of endovascular navigation for collaborative robotic catheterization. 2013.
- [7] Chi et al. Collaborative robot-assisted endovascular catheterization with generative adversarial imitation learning. 2020.