Parallel Reinforcement Learning and Gaussian Process Regression for Improved Physics-Based Nasal Surgery Planning

Mario Rüttgers[1,3](✉), Fabian Hübenthal[2,3], Makoto Tsubokura[3,4], and Andreas Lintermann[1,3]

[1] Jülich Supercomputing Centre, Forschungszentrum Jülich, 52428 Jülich, Germany
m.ruettgers@fz-juelich.de
[2] Institute of Aerodynamics and Chair of Fluid Mechanics (AIA), RWTH Aachen University, 52062 Aachen, Germany
[3] Department of Computational Science, Graduate School of System Informatics, Kobe University, 657-8501 Kobe, Japan
[4] Complex Phenomena Unified Simulation Research Team, RIKEN Center for Computational Science, 650-0047 Kobe, Japan

Abstract. Septoplasty and turbinectomy are among the most frequent but also most debated interventions in the field of rhinology. A previously developed tool enhances surgery planning by physical aspects of respiration, i.e., for the first time a reinforcement learning (RL) algorithm is combined with large-scale computational fluid dynamics (CFD) simulations to plan anti-obstructive surgery. In the current study, an improvement of the tool's predictive capabilities is investigated for the aforementioned types of surgeries by considering two approaches: (i) training of parallel environments is executed on multiple ranks and the agents of each environment share their experience in a pre-defined interval and (ii) for some of the state-reward combinations the CFD solver is replaced by a Gaussian process regression (GPR) model for an improved computational efficiency. It is found that employing a parallel RL algorithm improves the reliability of the surgery planning tool in finding the global optimum. However, parallel training leads to a larger number of state-reward combinations that need to be computed by the CFD solver. This overhead is compensated by replacing some of the computations with the GPR algorithm, i.e., around 6% of the computations can be saved without significantly degrading the predictions' accuracy. Nevertheless, increasing the number of state-reward combinations predicted by the GPR algorithm only works to a certain extent, since this also leads to larger errors.

Keywords: Septoplasty · turbinectomy · computational fluid dynamics · reinforcement learning · surrogate-based optimization

R. Wyrzykowski et al. (Eds.): PPAM 2024, LNCS 15581, pp. 79–96, 2025.
https://doi.org/10.1007/978-3-031-85703-4_6

1 Introduction

The combination of computational fluid dynamics (CFD) simulations and reinforcement learning (RL) is experiencing an increasing popularity for control applications based on flow physics. In [7], a scalable RL framework is developed to derive data-driven turbulence models for large eddy simulations (LES). An LES uses sub-grid scale (SGS) models such as the Smagorinsky [21] or the dynamic Smagorinsky [3] SGS approaches to model unresolved turbulent scales in the regime above the inertial subrange. The potential of an RL-augmented CFD solver is demonstrated by finding a control strategy for optimal eddy viscosity selection. In [13], CFD-based RL is employed to control the rotation of a flap implemented on a NACA0012 airfoil. The RL-agent is able to improve the aerodynamic performance of the airfoil for different angles of attack in turbulent flows computed with structured meshes containing around $800,000$ cells. For modelling the turbulence, the Reynolds-averaged Navier-Stokes (RANS)-based k-ω Shear Stress Transport (SST) model is selected. In [4], an RL algorithm is coupled to a CFD solver to reduce drag in channel flows. Each agent observes the velocity fluctuations in the streamwise and wall-normal direction, the reward is the percentage variation of the wall-shear stress, and each agent acts by imposing a positive (blowing) or negative (suction) wall-normal velocity at the channel wall. The CFD solver is a pseudo-spectral code that uses the Chebyshev polynomials in the wall-normal directions, and the time-advancement numerical scheme is a second-order Crank-Nicholson algorithm for the linear terms and a third-order Runge-Kutta method for the nonlinear terms. The Computational meshes contain up to $266,240$ cells. In [16], a proximal policy optimization (PPO) algorithm is employed to control flow through a heated and perturbed channel by changing the size of a constriction. After 265 simulations, the trained algorithm predicts a variation of the constriction whose results deviate by less than 1% from the reference solution obtained from 3,400 numerical simulations using a parameter sweep. The CFD simulations are solved with a thermal lattice-Boltzmann method (TLBM) on grids containing around $100,000$ cells.

The previously mentioned attempts for combining RL and CFD simulations employ relatively small computational grids ($< 1,000,000$ cells), allowing to explore a large action space efficiently. However, for large-scale simulations with an increased number of cells, coupling RL algorithms with CFD solvers becomes more and more challenging. In [18], for the first time large-scale CFD simulations with grids on the order of hundreds of million cells are coupled to an RL algorithm. That is, a TLBM is coupled to a deep Q learning (DQL) algorithm to plan surgical interventions for obstructed noses. The DQL algorithm modifies a level-set field to yield geometric variations of the upper airway, and receives feedback in terms of the pressure loss and temperature increase between the inlets (Nostrils) and outlet (Pharynx). The method is demonstrated for two patients, the first one suffering from a deviated septum and the second one from enlarged turbinates. The simulation domain of the first patient is resolved by about 110×10^6 cells, and the domain of the second patient by about 220×10^6 cells, using mesh resolutions of $\Delta x = 0.1\ mm$ to accurately resolve narrow channels and thin boundary layers [8,9]. The flow simulations were conducted for several days on 16 NVIDIA A100 graphics processing units (GPUs) for the first patient, and on 32 NVIDIA A100 GPUs for the second patient.

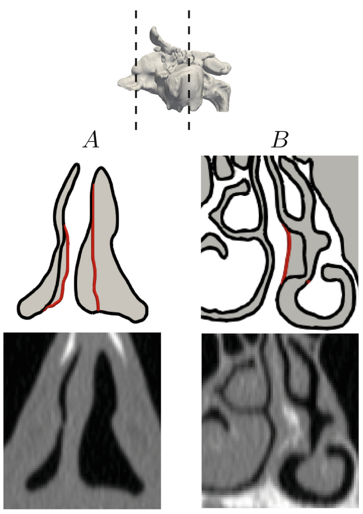
Coupling RL algorithms with large-scale CFD simulations requires an efficient exploration of the action space, to guarantee that agent's do not get stuck at local maxima of the reward function. A promising approach for an efficient exploration is parallel RL (PRL), where multiple environments run simultaneously and their agents exchange the collected experience in pre-defined intervals. In [22], the use of parallel environments during the learning process allows an efficient control of flow applications in a reasonable time. In [7], the CFD-RL framework can scale up to hundreds of parallel environments on thousands of cores, allowing to leverage modern HPC resources to either enable larger problems or faster turnaround times.

PRL increases the liability of finding the global maximum of the reward function, but also the total number of states explored by all agents, and, therefore, the computational efforts of the CFD simulations. Gaussian processes are suitable to counteract these additional costs by replacing some of the CFD computations. Gaussian process regression (GPR) has shown great potential to predict new states in an RL environment. GPR is a non-parametric Bayesian approach towards regression problems, that has the ability to provide uncertainty measurements on predictions [24]. In [15], GPR is used to successfully predict the temporal behavior in the well-known mountain-car problem [20], after the agents change the car's velocity. In [12], GPR is used as function approximation to model the state transition and the value functions of states for condition-based maintenance strategies in complex engineering systems. An RL algorithm is then developed to minimize the long-run average costs. In [5], GPR and RL are studied to control the system dynamics of an autonomous blimp. State transitions are modelled as a Gaussian process that is trained on the residual between a nonlinear ordinary differential equation (ODE) based on Newtonian principles and ground truth training data. The resulting GP-enhanced model is then coupled to an RL algorithm, allowing an estimate of uncertainty in addition to giving better state predictions than either ODE or the Gaussian process alone.

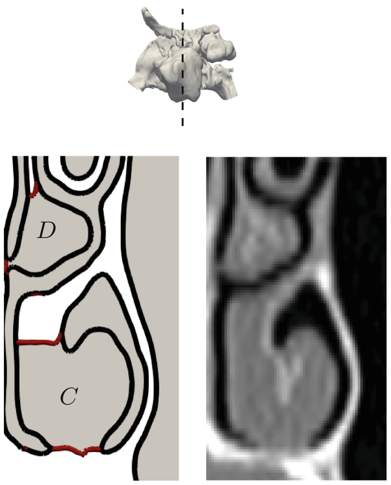
In the current study, GPR is for the first time combined with an PRL algorithm that receives feedback from large-scale CFD simulations. That is, the CFD and RL-based automated surgery planning, which has been introduced above [18], is conducted with multiple environments, and in each environment some of the state transitions are predicted by an GPR model to skip expensive CFD computations and accelerate training. The training procedure for each environment of the PRL algorithm is subdivided into two stages. In the first stage, the GPR model does not interact and feedback for the RL algorithm is solely provided by the CFD solver, as described in [18]. In the second stage, the GPR model is trained at the beginning of each episode with data from all states that have been computed based on CFD simulations so far. After each action, with the help of the uncertainty quantification of the GPR model, it is decided whether to use the GPR prediction or the CFD solver to determine a new state.

The manuscript is structured as follows. The medical data of the two patients are explained in Sect. 2, and the computational methods are presented in Sect. 3. In Sect. 4, surgery planning results for the two patients are presented in two parts. In the first part (Sect. 4.1), the reliability and computational costs based on training in a single environment are compared to parallel training with multiple simultaneous environments. In the second part (Sect. 4.2), results when solely coupling the PRL algorithm to the CFD solver are compared to results when some of the states are predicted by the GPR model. Finally, Sect. 5 provides a summary, conclusion, and outlook.

2 Medical Data



(a) First patient.



(b) Second patient.

Fig. 1. The nasal cavity of two patients, the first suffering from a deviated septum (A) and a bony spur (B), and the second from enlarged inferior (C) and middle (D) turbinates. The close-ups on cross-sectional areas illustrate the pre-surgical state (black), and planned interventions at locations A and B, as well as maxium possible interventions at locations C and D (red). They are juxtaposed to the corresponding pre-surgical CT images. (Color figure online)

Anonymized CT data of two patients are used. The first patient suffers from a deviated septum (location A) and a bony spur (location B), as shown in Fig. 1a, and the second patient from enlarged turbinates (locations C and D), illustrated in Fig. 1b. The patients gave informed consent for inclusion of the data in the current study. The CT data of the first patient are composed of 119 axial slices with 512×512 pixels each. The pixel spacing is 0.5 mm, and the space between the axial slices is 0.7 mm. The CT recordings of the second patient have 103 axial slices, again with 512×512 pixels each. The pixel spacing is 0.326 mm, and the space between the axial slices is 1.0 mm. The 3D model of the pre-surgical upper airway is extracted from the *Digital Imaging and Communications in Medicine* (DICOM) files of the CT data with the pipeline described in [17]. For the first patient, surgery planning of a septoplasty is investigated, and for the second patient, the surgical potential of a turbinectomy is analyzed. More details about the medical data, types of surgeries, and motivation from a medical background are given in [18,23].

3 Computational Methods

The general approach of combining CFD and RL for physics-based surgery planning is described in Sect. 3.1. This is followed by explaining the particularities of the PRL algorithm in Sect. 3.2, and the details of the GPR model in Sect. 3.3.

3.1 Physics-Based Surgery Planning

Surgery planning is realized with a combination of an RL algorithm and a CFD solver. Agents modify the geometry of the nasal cavity, and receive feedback from simulations conducted with the TLBM of the CFD solver m-AIA[1]. The number of agents depends on the number of surgical intervention locations ($\mathcal{G}$).

To modify the shape of the geometry, a level-set (LS) method is coupled to the TLBM. In the LS method, the geometry is represented using a signed distance function φ, the LS field. The LS field that represents the geometry after a modification by the agent (φ_1) is calculated using the linear interpolation [23]

$$\varphi_1 = (1 - \alpha_q)\varphi_2 + \alpha_q \varphi_3, \tag{1}$$

with the second and the third LS fields φ_2 and φ_3 containing information on the pre-surgical shape and the shape generated based on the surgeon's plan (first patient) or the maximum possible intervention (second patient). The factors $\alpha_q \in [0, 1]$ define the interpolation between the pre-surgical state $\alpha_q = 0$ and the state based on the surgeon's plan or the maximum possible intervention $\alpha_q = 1$, where $q = [1, ..., \mathcal{G}]$.

After an action, m-AIA receives α_q of the current state, and provides information about the new pressure loss and temperature increase to generate the reward $\mathcal{R}$

$$\mathcal{R} = \Delta p_{norm} + \Delta T_{norm}. \tag{2}$$

[1] *multiphysics - Aerodynamisches Institut Aachen* (m-AIA), an extended version of the formerly known *Zonal Flow Solver* (ZFS) [10], https://git.rwth-aachen.de/aia/MAIA/Solver.

The normalized pressure loss Δp_{norm} and the normalized temperature increase ΔT_{norm} between the nostrils and pharynx are given by

$$\Delta p_{norm} = \frac{\Delta p(\alpha_q) - \Delta p(\alpha = 0)}{\Delta p(\alpha = 1) - \Delta p(\alpha = 0)}, \tag{3}$$

$$\Delta T_{norm} = \frac{\Delta T(\alpha_q) - \Delta T(\alpha = 1)}{\Delta T(\alpha = 0) - \Delta T(\alpha = 1)}. \tag{4}$$

inspired by the work in [16].

The pressure loss Δp and temperature increase ΔT are defined as

$$\Delta p = \frac{1}{N_a} \sum_{i=N-N_a}^{N} \left(\frac{1}{H_{out}} \sum_{j=0}^{H_{out}} p^i_{tot,j} - p_{amb} \right), \tag{5}$$

$$\Delta T = \frac{1}{N_a} \sum_{i=N-N_a}^{N} \left(\frac{1}{H_{out}} \sum_{j=0}^{H_{out}} T^i_j - T_{amb} \right), \tag{6}$$

where H_{out} is the number of boundary cells at the outflow region, and p_{amb} is the ambient pressure. A feedback loop takes $N = 50{,}000$ time steps and the feedback is averaged over the last $N_a = 10{,}000$ time steps. More details about the single environment RL algorithm, the TLBM of m-AIA, and the flow and boundary conditions are provided in [18].

3.2 Parallel Reinforcement Learning

The extension from single environment RL to PRL allows to train n environments simultaneously. In a synchronization interval of $EP_s = 7$ episodes, the agents of each environment adopt the weights and biases, as well as the optimizer state of the environment with the so far highest reward.

The precision of the surgical tools used for conducting intranasal surgery only allows for discrete actions [18]. The flow field and reward for a state is only computed when the agents of an environment reach a state for the first time. The reward is then stored in a list, and the list is shared among each environment. If the state is reached again, the CFD simulation does not need to be conducted again. Instead, the corresponding reward is taken from the list. This reduces the total number of simulations to the total number of state-reward combinations reached by the agents of all environments N_{st}.

3.3 Gaussian Process Regression

GPR is a data- and prior-driven regression technique often used in the context of Bayesian optimization (BO) [19]. The objective function $f(\boldsymbol{x}) \in \{\Delta p_{norm}, \Delta T_{norm}\}$ is modeled as a Gaussian process $\mathcal{GP}$ with prior mean function $m(\boldsymbol{x}|\boldsymbol{\theta}_m)$ and covariance resp. kernel function $k(\boldsymbol{x}, \boldsymbol{x}'|\boldsymbol{\theta}_k)$ assuming that the original objective $f(\boldsymbol{x})$ is distributed as a Gaussian process, i.e., $f(\boldsymbol{x}) \sim \mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right)$.

The simulation data D_n is arranged as a matrix $\boldsymbol{X} = [-\boldsymbol{x}_i-] \in \mathbb{R}^{n \times d_x}$ for the inputs $\boldsymbol{x}_i$ and a vector $\boldsymbol{y} = [y_i] \in \mathbb{R}^{n \times 1}$ for the noisy outputs y_i, such that $D_n = \{\boldsymbol{X}, \boldsymbol{y}\}$. Using the Gaussian process prior model on the data, $\boldsymbol{y}$ is a joint multivariate normal distribution $\mathcal{N}$ with the data prior mean vector $\boldsymbol{m}(\boldsymbol{X}) = [m(\boldsymbol{x}_i)]$ and data prior covariance matrix $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}') = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ evaluated at each $\boldsymbol{x}_i, \boldsymbol{x}_j$ of $\boldsymbol{X}$, i.e., $\boldsymbol{f} = [f_i] \sim \mathcal{N}(\boldsymbol{m}(\boldsymbol{X}), \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}'))$ for the latent prior. By adding noise along the diagonal, the noisy covariance matrix $\boldsymbol{K}_{\mathrm{n}}(\boldsymbol{X}, \boldsymbol{X}') = K(\boldsymbol{X}, \boldsymbol{X}') + \sigma_{\mathrm{n}}^2 \boldsymbol{I}$ is obtained. From a frequentist perspective, the noise hyperparameter σ_{n} serves as a regularization parameter to favor simplistic explanations over more complex ones and leads to the predictive prior distribution, i.e., $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{m}(\boldsymbol{X}), \boldsymbol{K}_{\mathrm{n}}(\boldsymbol{X}, \boldsymbol{X}'))$. The noise hyperparameter σ_{n}, i.e., $\boldsymbol{\theta}_l = [\sigma_{\mathrm{n}}]$, is part of the likelihood function $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$, which is assumed to be Gaussian for the objective functions to be modelled. The prior belief expressed as the prior mean function $m(\boldsymbol{x})$ and the prior kernel $k(\boldsymbol{x}, \boldsymbol{x}')$, is conditioned on the data D_n using Bayes' theorem to obtain the Gaussian process posterior in closed-form with the posterior mean function $m|D_n(\boldsymbol{x})$, the posterior kernel $k|D_n(\boldsymbol{x}, \boldsymbol{x}')$ and the posterior standard deviation $s|D_n$ given in Eqs. (7) to (9). Based on the posterior, a single prediction at $\boldsymbol{x}$ given data D_n can be made as $y|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y} = y|\boldsymbol{x}, D_n \sim \mathcal{GP}(m|D_n(\boldsymbol{x}), k|D_n(\boldsymbol{x}, \boldsymbol{x}'))$. [14]

$$m|D_n(\boldsymbol{x}) = m(\boldsymbol{x}) + \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x})^T \boldsymbol{K}_{\mathrm{n}}(\boldsymbol{X}, \boldsymbol{X}')^{-1}(\boldsymbol{y} - \boldsymbol{m}(\boldsymbol{X})) \tag{7}$$

$$k|D_n(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x})^T \boldsymbol{K}_{\mathrm{n}}(\boldsymbol{X}, \boldsymbol{X}')^{-1} \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}') \tag{8}$$

$$s|D_n(\boldsymbol{x}) = \sqrt{k|D_n(\boldsymbol{x}, \boldsymbol{x}' = \boldsymbol{x})} \tag{9}$$

By choosing the prior mean for global and the prior kernel functions for local characteristics, prior knowledge can be incorporated into the stochastic model of the objective function. In the current study, the prior mean function $m(\boldsymbol{x})$ is set to a constant value C which is learned from the data during the hyperparameter tuning process, cf. Eq. (10).

For the covariance function, the squared exponential function resp. radial basis function (RBF) with automatic relevance determination (ARD), cf. Eq. (11), assuming a smooth, i.e., infinitely differentiable, objective function is chosen. In general, the prior mean m and the prior kernel k have hyperparameters, i.e., $\boldsymbol{\theta}_m$ and $\boldsymbol{\theta}_k$, respectively. In the case of using the data mean, there are no prior mean hyperparameters to tune, i.e., $\boldsymbol{\theta}_m = \varnothing$, and for the constant prior mean there is one mean hyperparameter, i.e. $\boldsymbol{\theta}_m = [C]$. However, the squared exponential kernel resp. RBF kernel with the scale matrix $\boldsymbol{M} = \mathrm{diag}(\boldsymbol{l})^{-2}$ possesses hyperparameters $\boldsymbol{\theta}_k = \{\sigma_f, \boldsymbol{l}\}$ including the signal variance σ_f and lengthscales $\boldsymbol{l} = [l_\lambda, l_A, l_T]^{\mathrm{T}}$ for each dimension of $\boldsymbol{x}$ according to ARD. The hyperparameters of the prior mean, the prior kernel and the likelihood function $\boldsymbol{\theta} = \{\boldsymbol{\theta}_m, \boldsymbol{\theta}_k, \boldsymbol{\theta}_l\}$ are tuned by maximum likelihood estimation (MLE) to best fit the data with respect to the Gaussian likelihood function $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$, i.e.,

$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathrm{MLE}(\boldsymbol{\theta})$ with $\mathrm{MLE}(\boldsymbol{\theta}) = \ln p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})$, using GPyTorch [2]. A detailed explanation of the mathematical background of GPR can be found in Rasmussen et al. [14].

$$m(\boldsymbol{x}) = m(\boldsymbol{x}|\boldsymbol{\theta}_m) = \mathrm{mean}(\boldsymbol{y}) = \quad C \tag{10}$$

$$k(\boldsymbol{x},\boldsymbol{x}') = k(\boldsymbol{x},\boldsymbol{x}'|\boldsymbol{\theta}_k) = \sigma_f^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^T \boldsymbol{M}(\boldsymbol{x}-\boldsymbol{x}')\right) = \mathrm{rbf}(\boldsymbol{x},\boldsymbol{x}') \tag{11}$$

Figure 2 provides an overview over the agent-environment interaction of the combined PRL-GPR approach. After an action, m-AIA is employed if the current number of state-reward combinations N_{st} has not surpassed the number of training data N_t. Otherwise, the GPR model is employed. However, the GPR model's results are only used if

$$CI = 1.96 \times s|D_n(\boldsymbol{x}) < CI_l \times \mathcal{GP}\left(m_n(\boldsymbol{x}), k_n(\boldsymbol{x},\boldsymbol{x}')\right), \tag{12}$$

where CI stands for the confidence interval and CI_l represents the confidence interval limit. If this condition is not fulfilled, the results are computed by m-AIA.

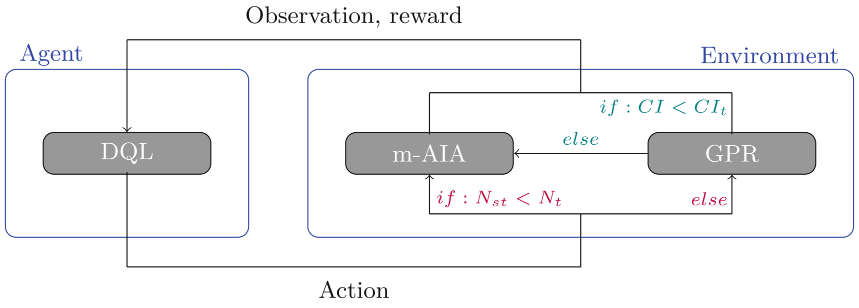


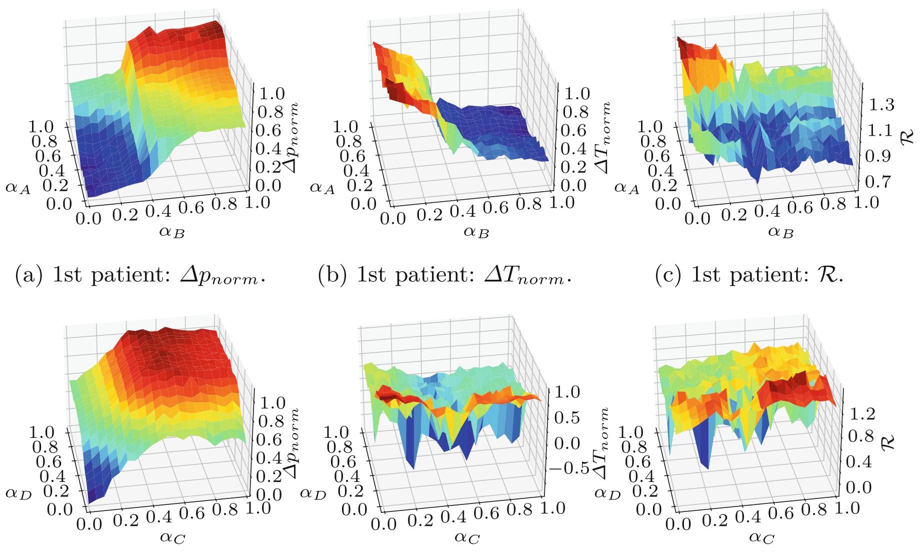
Fig. 2. Agent-environment interaction in the combined PRL-GPR approach.

4 Results

In Sect. 4.1 the performance of the PRL algorithm is analyzed for a varying number of simultaneously running training environments. In Sect. 4.2 the trade-off between training time and accuracy when employing the GPR model is investigated. The PRL and GPR algorithms were trained on the CPU partition of the *Jülich Research on Exascale Cluster Architectures* (JURECA-DC), *Forschungszentrum Jülich* [6]. The flow simulations for the first patient were conducted on 4 nodes of the GPU partition of JURECA-DC, i.e., on a total number of 16 NVIDIA A100 GPUs, and for the second patient on 8 nodes, i.e., on a total number of 32 NVIDIA A100 GPUs.

4.1 Parallel Reinforcement Learning

Simultaneous training in multiple environments is evaluated by comparing the number of numerical simulations needed to find the global optimum. Figure 3 shows Δp_{norm}, ΔT_{norm}, and $\mathcal{R}$ for all possible state-reward combinations of both patients. For the first patient, the topology of Δp_{norm} (Fig. 3a) and ΔT_{norm} (Fig. 3b) is smooth and the global optimum of $\mathcal{R}$ is found at $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.8, 0.0)$ (Fig. 3c). For the second patient, the topology of Δp_{norm} (Fig. 3d) is similarly smooth compared to the first patient. However, the topology of ΔT_{norm} (Fig. 3e) is much rougher than the one of the first patient, which results in a jagged topology of $\mathcal{R}$ (Fig. 3f), where the global maximum lies at $(\alpha_C^{opt}, \alpha_D^{opt}) = (0.85, 0.25)$. The geometric changes that correspond to the global maximum of each patient are illustrated in [18].



(a) 1st patient: Δp_{norm}. (b) 1st patient: ΔT_{norm}. (c) 1st patient: $\mathcal{R}$.

(d) 2nd patient: Δp_{norm}. (e) 2nd patient: ΔT_{norm}. (f) 2nd patient: $\mathcal{R}$.

Fig. 3. Δp_{norm}, ΔT_{norm}, and $\mathcal{R}$ for all 441 possible state-reward combinations of both patients based on simulations of m-AIA. The colors indicate increasing quantities from blue to red. (Color figure online)

For each patient, ten training runs are conducted on $n = 1, 2, 3$ environments, and each run is stopped once an environment reaches 150 episodes. Table 1 provides results in terms of N_{st} averaged over the ten runs, and the number of runs in which the agents found the global maximum (Success rate). Note that the maximum possible number of state-reward combinations N_{max} is 441, since each patient has two surgical interventions and a discretized action interval of $\Delta\alpha = 0.05$ is chosen.

Table 1. State-reward combinations and the success rate achieved by the PRL algorithm trained with $n = 1, 2, 3$ environments.

n	First patient			Second patient		
	N_{st}	$\frac{N_{st}}{N_{max}}$	Success rate	N_{st}	$\frac{N_{st}}{N_{max}}$	Success rate
1	124	0.28	8/10	167	0.38	6/10
2	260	0.59	10/10	265	0.60	7/10
3	290	0.66	10/10	326	0.74	10/10

Generally, finding the global optimum seems to be an easier task for the first patient, since training with two environments already has a success rate of 10/10 and 59% of N_{max} are required. In contrast, reaching such a success rate for the second patient requires three environments and 74% of N_{max} are required. It becomes clear that the disadvantage of increasing the number of parallel environments is an increased number of state-reward combinations N_{st}. Each of those combinations stands for a new surface modification and the LB solver needs 50,000 time steps to compute Δp_{norm} and ΔT_{norm}. For all cases ($n = 1, 2, 3$) N_{st} for the second patient is higher than for the first patient. This comes from the fact that the smooth topology of Δp_{norm} and ΔT_{norm} of the first patient yield a clear global optimum (See Fig. 3c), whereas the rough topology of ΔT_{norm} of the second patient results in a topology of $\mathcal{R}$ with multiple local optima that are close to the global optimum (See Fig. 3f).

4.2 Parallel Reinforcement Learning and Gaussian Process Regression

In this section, the surgical interventions proposed by the PRL algorithm are analyzed for cases where Δp_{norm} and ΔT_{norm} of some states are each predicted by an individual GPR model, i.e., $\Delta p_{norm} \sim \mathcal{GP}\left(m_p(\boldsymbol{x}), k_p(\boldsymbol{x}, \boldsymbol{x}')\right)$ and $\Delta T_{norm} \sim \mathcal{GP}\left(m_T(\boldsymbol{x}), k_T(\boldsymbol{x}, \boldsymbol{x}')\right)$. Each case is evaluated by the errors $\bar{E}_p$ and $\bar{E}_T$

$$\bar{E}_p = \frac{\sum^{N_{gpr}} E_p}{N_{gpr}} = \frac{\sum^{N_{gpr}} \frac{\Delta p_{norm}(\alpha_q^{sim}) - \Delta p_{norm}(\alpha_q^{pred})}{\Delta p_{norm}(\alpha_1^{sim})}}{N_{gpr}}, \tag{13}$$

$$\bar{E}_T = \frac{\sum^{N_{gpr}} E_T}{N_{gpr}} = \frac{\sum^{N_{gpr}} \frac{\Delta T_{norm}(\alpha_q^{sim}) - \Delta T_{norm}(\alpha_q^{pred})}{\Delta T_{norm}(\alpha_1^{sim})}}{N_{gpr}}, \tag{14}$$

where N_{gpr} stands for the number of states in which the m-AIA simulation is replaced by the GPR model. In the following, results for cases with varying N_t and CI_l and training on $n = 3$ ranks are presented for both patients.

Results for ten runs with $N_t = 250$ and $CI_l = 0.05$ are shown in Table 2 for the first patient and in Table 3 for the second patient.

In case of the first patient, $N_{st} < 250$ is observed for two runs in which the GPR model is not activated. In the remaining runs, the GPR model is

Table 2. Results for ten runs of the first patient with $N_t = 250$ and $CI_l = 0.05$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
185	–	–	–
326	27 (8.3%)	0.016	0.027
279	22 (7.9%)	0.057	0.060
333	43 (13.0%)	0.032	0.063
255	3 (1.2%)	0.027	0.062
266	10 (3.8%)	0.076	0.133
286	32 (11.2%)	0.032	0.082
225	–	–	–
270	3 (0.01%)	0.049	0.053
254	4 (1.6%)	0.024	0.051
284	18 (6.3%)	0.039	0.066

Table 3. Results for ten runs of the second patient with $N_t = 250$ and $CI_l = 0.05$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
371	31 (8.4%)	0.025	0.077
321	9 (0.3%)	0.021	0.072
303	22 (7.3%)	0.027	0.059
288	16 (5.6%)	0.017	0.070
245	–	–	–
380	43 (11.3%)	0.019	0.078
348	19 (5.5%)	0.023	0.081
264	5 (1.9%)	0.025	0.057
389	25 (6.4%)	0.019	0.089
311	4 (1.3%)	0.018	0.081
330	19 (5.8%)	0.025	0.083

used on average for the computation of 18 states, which is 6.3% of the used state-reward combinations. The mean error for the temperature increase is with $\bar{E}_T = 6.6\%$ larger than the error for the pressure loss with $\bar{E}_p = 3.9\%$. In all runs $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.8, 0.0)$ has been found with $\Delta p_{norm} = 0.414$ and $\Delta T_{norm} = 1.005$ yielding $\mathcal{R} = 1.419$.

In case of the second patient, $N_{st} < 250$ is observed for a single run. In the remaining runs, the GPR model is used for computing 19 states on average, which is with 5.8% of the used state-reward combinations in a similar range compared to the first patient. Again, $\bar{E}_T = 8.3\%$ is larger than $\bar{E}_p = 2.5\%$. The relatively large $\bar{E}_T$ in case of the second patient comes from the rough and jagged topology of ΔT_{norm} (Fig. 3e), which makes predictions challenging. In all runs $(\alpha_C^{opt}, \alpha_D^{opt}) = (0.85, 0.25)$ has been found with $\Delta p_{norm} = 0.853$ and $\Delta T_{norm} = 0.722$ yielding $\mathcal{R} = 1.575$.

Tables 4 and 5 present the results for the first and second patient for ten runs with $N_t = 200$ and $CI_l = 0.05$. In case of the first patient, the GPR model is activated for all runs. Reducing N_t from 250 to 200 increases the mean N_{gpr} from 6.3% to 18.9%, with a trade-off in terms of $\bar{E}_p = 0.042$ and $\bar{E}_T = 0.077$, compared to $\bar{E}_p = 0.039$ and $\bar{E}_T = 0.066$ in Table 2. These increased errors lead to two runs in which $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.8, 0.0)$ is not found, i.e., the first run yields $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.95, 0.0)$, and the fifth run $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.75, 0.1)$. For the first run, this would mean CFD results of $\Delta p_{norm} = 0.440$ and $\Delta T_{norm} = 0.897$ yielding $\mathcal{R} = 1.337$, and for the fifth run $\Delta p_{norm} = 0.431$ and $\Delta T_{norm} = 0.865$ with $\mathcal{R} = 1.296$, resulting in larger weights on ΔT than on Δp compared to the results for $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.8, 0.0)$ presented above. Especially, the fifth run is characterized by large errors ($\bar{E}_p = 0.081$ and $\bar{E}_T = 0.150$), which implies that for this run the training data is insufficient.

In case of the second patient, $N_{st} > 200$ is observed for all runs meaning the GPR algorithm is always activated again. In contrast to the first patient,

Table 4. Results for the first patient with $N_t = 200$ and $CI_l = 0.05$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
277	45 (16.3%)	0.027	0.082
265	39 (14.7%)	0.032	0.064
326	98 (30.1%)	0.030	0.045
248	25 (10.1%)	0.038	0.078
274	63 (23.0%)	0.081	0.150
284	50 (17.6%)	0.04	0.066
268	37 (13.8%)	0.022	0.038
288	33 (11.5%)	0.063	0.099
310	75 (24.2%)	0.049	0.070
313	71 (22.7%)	0.038	0.074
285	54 (18.9%)	0.042	0.077

Table 5. Results for the second patient with $N_t = 200$ and $CI_l = 0.05$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
414	41 (9.9%)	0.022	0.067
343	29 (8.5%)	0.024	0.086
260	14 (5.4%)	0.035	0.077
292	15 (5.1%)	0.026	0.085
264	5 (1.9%)	0.036	0.063
289	14 (4.8%)	0.023	0.090
275	20 (7.2%)	0.028	0.080
284	21 (7.4%)	0.028	0.070
271	15 (5.5%)	0.022	0.052
264	10 (3.8%)	0.023	0.051
296	18 (6.1%)	0.027	0.072

reducing N_t from 250 to 200 does not change the mean N_{gpr} for the second patient with 6.1% significantly compared to 5.8% in Table 3. This means that the reduced training data for the GPR algorithm, which starts to be trained at $N_{ST} > 200$, cause generally large confidence intervals of the algorithm's predictions. This highlights again the challenge of predicting the second patient's temperature increase.

In Tables 6 and 7, the results for the first and second patient for ten runs with $N_t = 250$ and $CI_l = 0.1$ are provided. By increasing the threshold of the confidence interval from $CI_l = 0.05$ to $CI_l = 0.1$, for the first patient the mean N_{gpr} is increased from 6.3% to 12.5%, and for the second patient from

Table 6. Results for the first patient with $N_t = 250$ and $CI_l = 0.1$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
367	86 (23.4%)	0.033	0.051
226	–	–	–
293	36 (12.3%)	0.024	0.085
268	15 (5.6%)	0.035	0.104
248	–	–	–
221	10 (3.8%)	0.076	0.133
189	–	–	–
231	–	–	–
259	8 (3.1%)	0.024	0.036
374	65 (17.4%)	0.028	0.052
297	37 (12.5%)	0.037	0.077

Table 7. Results for the second patient with $N_t = 250$ and $CI_l = 0.1$.

N_{st}	N_{gpr}	$\bar{E}_p$	$\bar{E}_T$
222	–	–	–
293	18 (6.1%)	0.022	0.112
258	5 (1.9%)	0.022	0.077
368	44 (12.0%)	0.023	0.101
329	28 (8.5%)	0.029	0.082
359	48 (13.4%)	0.027	0.087
261	5 (1.9%)	0.016	0.044
307	36 (11.3%)	0.026	0.085
321	21 (6.5%)	0.021	0.054
349	20 (5.7%)	0.017	0.066
316	25 (7.9%)	0.023	0.079

5.8% to 7.9%. However, predicting more states by the GPR model due to the larger confidence interval limit comes at the cost of more inaccurate predictions. That is, for the first patient instead of $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.8, 0.0)$, in the first and tenth runs $(\alpha_A^{opt}, \alpha_B^{opt}) = (0.9, 0.0)$ are yielded, which would mean CFD results of $\Delta p_{norm} = 0.437$ and $\Delta T_{norm} = 0.956$ with $\mathcal{R} = 1.393$. For the second patient instead of $(\alpha_C^{opt}, \alpha_D^{opt}) = (0.85, 0.25)$, in the third run $(\alpha_C^{opt}, \alpha_D^{opt}) = (0.8, 0.3)$ and in run eight $(\alpha_C^{opt}, \alpha_D^{opt}) = (0.6, 0.15)$ are found. For the third run, this would mean CFD results of $\Delta p_{norm} = 0.933$ and $\Delta T_{norm} = 0.552$ yielding $\mathcal{R} = 1.485$, and for run number eight $\Delta p_{norm} = 0.789$ and $\Delta T_{norm} = 0.731$ with $\mathcal{R} = 1.52$.

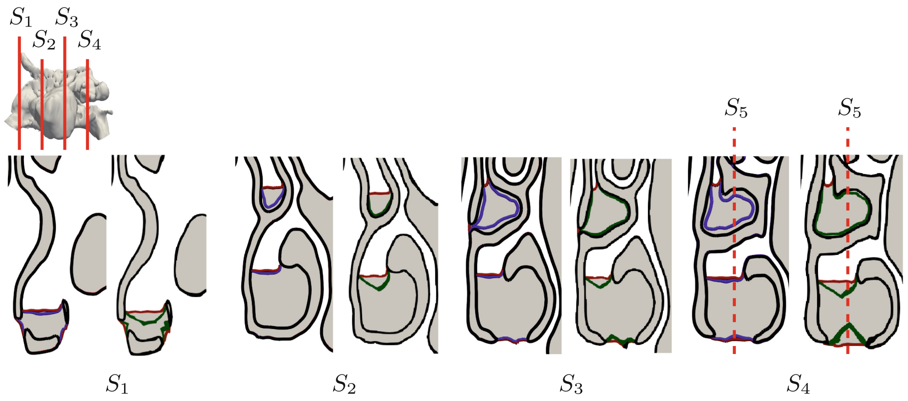


Fig. 4. The pre-surgical state [black], the state based on the maximum possible intervention [red], the state for (α_C, α_D)=(0.85, 0.25) [blue, left], and for (α_C, α_D)=(0.6, 0.15)[green, right] at the four cross-sectional areas S_1-S_4. S_5 is illustrated for a better orientation of the cross-sectional area shown in Fig. 5. Note that these are only representative cross-sectional areas. By changing α_q, the 3D region defined by φ_1 is modified. (Color figure online)

Run number eight of Table 7 has the overall largest deviations in terms of α_q, i.e., a deviation of 0.25 for α_C^{opt}, and 0.1 for α_D^{opt}. These deviations are illustrated for the four cross-sectional areas $S_1 - S_4$ in Fig. 4. The figure shows the pre-surgical state [black], the state based on the maximum possible intervention [red], the state for (α_C, α_D)=(0.85, 0.25) [blue, left], and for (α_C, α_D)=(0.6, 0.15)[green, right]. Whereas the deviation of α_C has only a minor impact on the nasal cavity shape, the deviation of α_D causes a narrower passage at the inferior turbinate. The influence of this narrowing on the pressure loss and temperature increase between the inlets and locations inside of the nasal cavity is further shown in Fig. 5 for the cross-sectional area S_5. The red circles highlight the region of large deviations between the two cases (α_C, α_D)=(0.85, 0.25) [left] and (α_C, α_D)=(0.6, 0.15) [right]. The narrowed nasal passage near the inferior turbinate causes the higher pressure loss but also the increased heating of incoming air before entering the lung.

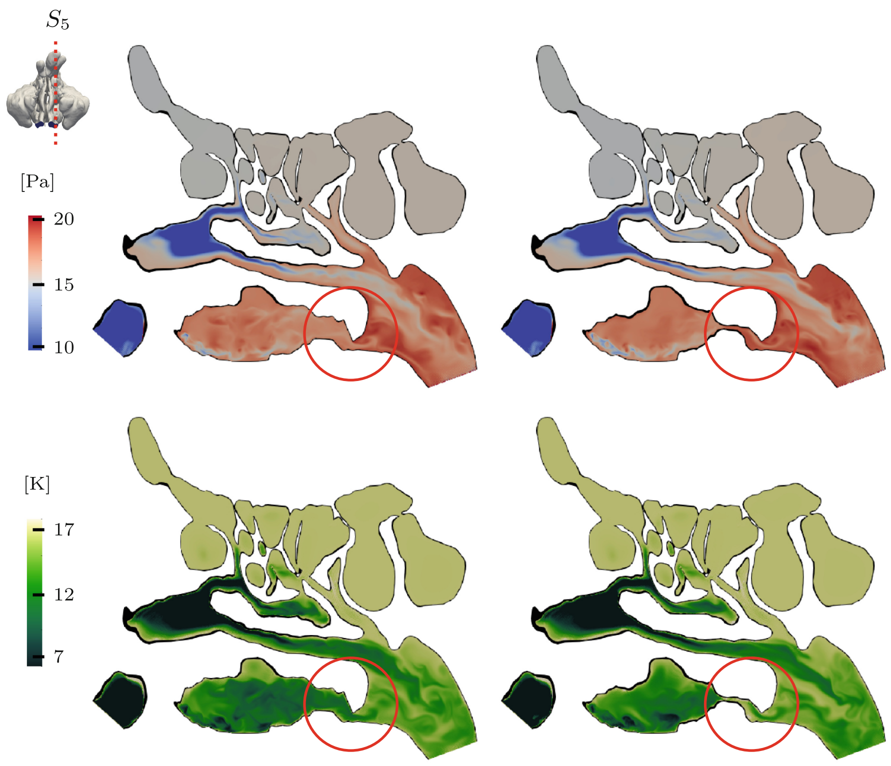


Fig. 5. Total pressure loss [top] and temperature increase [bottom] between the inlets and locations inside of the nasal cavity for (α_C, α_D)=(0.85, 0.25) [left] and (α_C, α_D)=(0.6, 0.15) [right] at the cross-sectional area S_5, which is illustrated by the dashed line in this figure and in Fig. 4. The red circles highlight the region of large deviations between the two cases. (Color figure online)

Replacing the CFD simulation by the GPR algorithm for one surface modification saves 50,000 time steps that are needed to reach a converged flow field and get physically meaningful results. For the first patient with 110×10^6 cells on 16 NVIDIA A100 GPUs and for the second patient with 220×10^6 cells on 32 NVIDIA A100 GPUs this takes around 20 minutes. Given the example in Table 4, $N_{gpr} = 54$ would mean savings of 1,080 minutes (18 hours) of compute time. The consumption for training the DQL algorithm is negligible, since a multilayer perceptron network with only three fully connected layers and 64 neurons per layer is employed. The same accounts for training and eploying the GPR algorithm, whose training data cannot exceed the pressure and temperature data of 441 surface modifications. Therefore, the number of training iterations for the GPR algorithm has been set with 150 already quite high and is not considered a critical parameter in the current study.

The current manuscript describes only the last step of the entire workflow from patient data acquisition to final surgical planning. Once CT data of patients are generated, the workflow consists of three steps:

1. Super-resolution from coarse to fine CT data to allow reliable CFD simulations
2. Automated segmentation and extraction of the 3D nasal cavity geometry from the CT data
3. CFD-RL-based surgery planning

In [11,17,18], detailed results, practical benefits, and potential imitations for each of these steps are reported. The reader is referred to these publications to get insights into the entire workflow.

5 Summary, Conclusion and Outlook

The challenges of septoplasties and turbinectomies necessitate new approaches for physics-based surgery planning. In a previous work, a DQL algorithm has been for the first time combined with large-scale CFD simulations to propose surgical interventions exploiting fluid mechanics knowledge about the pressure loss and temperature increase between the inflow (nostrils) and the outflow (pharynx) regions [18]. Two patients have been investigated, the first patient suffering from a deviated septum accompanied by a bony spur, and the second patient from enlarged turbinates in the left nasal passage. The previous work was a first try for automated physics-based surgery planning of obstructed noses, but left room for improvements in terms of finding the optimum surgical interventions efficiently. The current study addresses this challenge by two approaches. In the first approach, training of parallel environments is executed on multiple ranks and the agents of each environment share their experience in a pre-defined interval. In the second approach, for some of the state-reward combinations the CFD solver (m-AIA) is replaced by an GPR model for an improved efficiency.

Employing an PRL algorithm improves the reliability of the surgery planning tool, i.e., training multiple environments on 3 ranks yields a success rate of 10/10 in finding the global optimum. However, parallel training is accompanied by a larger exploration of the action space, yielding more state-reward combinations that need to be computed by m-AIA. This overhead in computations can partly be compensated by replacing some of the computations with the GPR model. A baseline case with $N_t = 250$ and $CI_l = 0.05$ reveals that for the first patient 6.3% of the m-AIA computations can be saved, and for the second patient 5.8%. Nevertheless, increasing the number of state-reward combinations predicted by the GPR model by enlarging the hyperparameters N_t and CI_l only works to a certain extent, since this leads to larger errors and difficulties in finding the global maximum.

Overall, even with the option of replacing some of the computations done by m-AIA with an GPR algorithm, there is a need for elaborating ways to reduce computational costs. One such a way is currently being investigated by the authors in a follow-up study that contains three steps:

1. In the first step, a pre-defined number of flow configurations for varying interpolation factors is computed by m-AIA.
2. In the second step, the flow fields of these configurations function as training data for a graph convolutional neural network (GCNN) which is trained to predict flow fields of varying geometries for the same patient. GCNNs are capable of predicting flow fields around irregularly shaped bodies whose meshes can easily be converted into graphs [1].
3. In the third step, the trained GCNN is coupled to an PRL algorithm to determine the optimized shape. This approach allows a much larger number of state-reward computations, and, therefore, an increased action space.

The goal is to train agents in modifying the CT data directly to explore geometry variations that go beyond the action space that is pre-defined by a surgeon.

Acknowledgments. The research leading to these results has been conducted in the HANAMI project, which receives funding from the European Union Horizon Europe Programme - Grant Agreement Number 101136269 under the call HORIZON-EUROHPC-JU-2022-INCO-04. The authors gratefully acknowledge the computing time granted by the JARA Vergabegremium and provided on the JARA Partition part of the supercomputer JURECA [6] at *Forschungszentrum Jülich*.

Disclosure of Interests. This statement is to declare that the authors of this manuscript, Mario Rüttgers, Fabian Hübenthal, Makoto Tsubokura, and Andreas Lintermann do not possess any financial dependence that might bias this work. The authors hereby declare that no conflict of interest exists in this work.

References

1. Chen, J., Hachem, E., Viquerat, J.: Graph neural networks for laminar flow prediction around random two-dimensional shapes. Phys. Fluids **33**(12), 123607 (2021). https://doi.org/10.1063/5.0064108
2. Gardner, J.R., Pleiss, G., Bindel, D., Weinberger, K.Q., Wilson, A.G.: GPyTorch: blackbox matrix-matrix gaussian process inference with GPU acceleration. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 7587–7597. NIPS 2018, Curran Associates Inc., Red Hook, NY, USA (2018). https://doi.org/10.5555/3327757.3327857
3. Germano, M., Piomelli, U., Moin, P., Cabot, W.H.: A dynamic subgrid? Scale eddy viscosity model. Phys. Fluids A Fluid Dyn. **3**(7), 1760–1765 (1991). https://doi.org/10.1063/1.857955
4. Guastoni, L., Rabault, J., Schlatter, P., Azizpour, H., Vinuesa, R.: Deep reinforcement learning for turbulent drag reduction in channel flows. Eur. Phys. J. E, Soft Matter **46**, 27 (2023). https://doi.org/10.1140/epje/s10189-023-00285-8
5. Ko, J., Klein, D.J., Fox, D., Haehnel, D.: Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In: Proceedings 2007 IEEE International Conference on Robotics and Automation, pp. 742–747. IEEE (2007). https://doi.org/10.1109/ROBOT.2007.363075
6. Krause, D., Thörnig, P.: JURECA: Modular supercomputer at Jülich Supercomputing Centre. J. Large-scale Res. Facil. **4**, A132 (2018). https://doi.org/10.17815/jlsrf-4-121-1

7. Kurz, M., Offenhäuser, P., Viola, D., Shcherbakov, O., Resch, M., Beck, A.: Deep reinforcement learning for computational fluid dynamics on HPC systems. J. Comput. Sci. **65**, 101884 (2022). https://doi.org/10.1016/j.jocs.2022.101884
8. Lintermann, A., Meinke, M., Schröder, W.: Fluid mechanics based classification of the respiratory efficiency of several nasal cavities. Comput. Biol. Med. **43**(11), 1833–1852 (2013). https://doi.org/10.1016/j.compbiomed.2013.09.003
9. Lintermann, A., Eitel-Amor, G., Meinke, M., Schröder, W.: Lattice-boltzmann solutions with local grid refinement for nasal cavity flows. In: New Results in Numerical and Experimental Fluid Mechanics VIII, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 121, pp. 583–590. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-35680-3_69
10. Lintermann, A., Meinke, M., Schröder, W.: Zonal flow solver (ZFS): a highly efficient multi-physics simulation framework. Int. J. Comput. Fluid Dyn. **34**(7–8), 458–485 (2020). https://doi.org/10.1080/10618562.2020.1742328
11. Liu, X., et al.: Refining computer tomography data with super-resolution networks to increase the accuracy of respiratory flow simulations. Futur. Gener. Comput. Syst. **159**, 474–488 (2024). https://doi.org/10.1016/j.future.2024.05.020
12. Peng, S., Feng, Q.M.: Reinforcement learning with gaussian processes for condition-based maintenance. Comput. Ind. Eng. **158**, 107321 (2021). https://doi.org/10.1016/j.cie.2021.107321
13. Portal-Porras, K., Fernandez-Gamiz, U., Zulueta, E., Garcia-Fernandez, R., Etxebarria Berrizbeitia, S.: Active flow control on airfoils by reinforcement learning. Ocean Eng. **287**, 115775 (2023). https://doi.org/10.1016/j.oceaneng.2023.115775
14. Rasmussen, C.E., Williams, C.K.: Gaussian Processes for Machine Learning, vol. 2. MIT press Cambridge, MA (2006)
15. Rasmussen, C., Kuss, M., Thrun, S., Saul, L., Schölkopf, B.: Gaussian processes in reinforcement learning. In: Advances in Neural Information Processing Systems, vol. 16, pp. 751–759 (2004)
16. Rüttgers, M., Waldmann, M., Schröder, W., Lintermann, A.: Machine learning-based control of perturbed and heated channel flows. In: Jagode, H., Anzt, H., Ltaief, H., Luszczek, P. (eds.) High Performance Computing, pp. 7–22. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-90539-2_1
17. Rüttgers, M., Waldmann, M., Schröder, W., Lintermann, A.: A machine-learning-based method for automatizing lattice-boltzmann simulations of respiratory flows. Appl. Intell. **52**, 9080–9100 (2022). https://doi.org/10.1007/s10489-021-02808-2
18. Rüttgers, M., Waldmann, M., Vogt, K., Ilgner, J., Schröder, W., Lintermann, A.: Automated surgery planning for an obstructed nose by combining computational fluid dynamics with reinforcement learning. Comput. Biol. Med. **173**, 108383 (2024). https://doi.org/10.1016/j.compbiomed.2024.108383
19. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., de Freitas, N.: Taking the human out of the loop: a review of Bayesian optimization. Proc. IEEE **104**(1), 148–175 (2016). https://doi.org/10.1109/JPROC.2015.2494218
20. Singh, S., Sutton, R., Kaelbling, P.: Reinforcement learning with replacing eligibility traces. Mach. Learn. **22** (1995). https://doi.org/10.1023/A:1018012322525
21. Smagorinsky, J.: General circulation experiments with the primitive equations: I. the basic experiment. Monthly Weather Rev. **91**(3), 99–164 (1963). https://doi.org/10.1175/1520-0493(1963)091$<$0099:GCEWTP$>$2.3.CO;2
22. Viquerat, J., Hachem, E.: Parallel bootstrap-based on-policy deep reinforcement learning for continuous fluid flow control applications. Fluids **8**(7) (2023). https://doi.org/10.3390/fluids8070208

23. Waldmann, M., Rüttgers, M., Lintermann, A., Schröder, W.: Virtual surgeries of nasal cavities using a coupled lattice-boltzmannlevel-set approach. J. Eng. Sci. Med. Diag. Therapy **5**(3) (2022). https://doi.org/10.1115/1.4054042
24. Williams, C., Rasmussen, C.: Gaussian processes for regression. In: Touretzky, D., Mozer, M., Hasselmo, M. (eds.) Advances in Neural Information Processing Systems, vol. 8. MIT Press (1995)