







# Machine-Learning-Based Control of Perturbed and Heated Channel Flows

Mario Rüttgers<sup>1,2,3</sup> , Moritz Waldmann<sup>1,3</sup> , Wolfgang Schröder<sup>1,3</sup> ,  
and Andreas Lintermann<sup>2,3</sup> 

<sup>1</sup> Institute of Aerodynamics and Chair of Fluid Mechanics, RWTH Aachen University, Wüllnerstraße 5a, 52062 Aachen, Germany  
[m.ruettgers@aia.rwth-aachen.de](mailto:m.ruettgers@aia.rwth-aachen.de)

<sup>2</sup> Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Wilhelm-Johnen-Straße, 52425 Jülich, Germany

<sup>3</sup> Jülich Aachen Research Alliance Center for Simulation and Data Science, RWTH Aachen University and Forschungszentrum Jülich, Seffenter Weg 23, 52074 Aachen, Germany

**Abstract.** A reinforcement learning algorithm is coupled to a thermal lattice-Boltzmann method to control flow through a two-dimensional heated channel narrowed by a bump. The algorithm is allowed to change the disturbance factor of the bump and receives feedback in terms of the pressure loss and temperature increase between the inflow and outflow region of the channel. It is trained to modify the bump such that both fluid mechanical properties are rated equally important. After a modification, a new simulation is initialized using the modified geometry and the flow field computed in the previous run. The thermal lattice-Boltzmann method is validated for a fully developed isothermal channel flow. After 265 simulations, the trained algorithm predicts an averaged disturbance factor that deviates by less than 1% from the reference solution obtained from 3,400 numerical simulations using a parameter sweep over the disturbance factor. The error is reduced to less than 0.1% after 1,450 simulations. A comparison of the temperature, pressure, and streamwise velocity distributions of the reference solution with the solution after 1,450 simulations along the line of the maximum velocity component in streamwise direction shows only negligible differences. The presented method is hence a valid method for avoiding expensive parameter space explorations and promises to be effective in supporting shape optimizations for more complex configurations, e.g., in finding optimal nasal cavity shapes.

**Keywords:** Reinforcement learning · Proximal policy optimization · Thermal lattice-Boltzmann · Flow control

---

M. Rüttgers and M. Waldmann—Authors contributed equally.

© Springer Nature Switzerland AG 2021

H. Jagode et al. (Eds.): ISC High Performance 2021 Workshops, LNCS 12761, pp. 7–22, 2021.

[https://doi.org/10.1007/978-3-030-90539-2\\_1](https://doi.org/10.1007/978-3-030-90539-2_1)

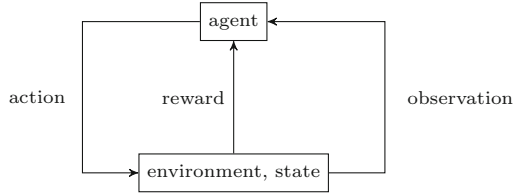
# 1 Introduction

A deviated septum is a common rhinological pathology, which may cause narrowed upper airway passages blocking the inhaled air. Frequently, removing tissue to widen the nasal airway is a popular surgical intervention to alleviate a patient’s complaints. The treatment is known as septoplasty. Unfortunately, patients are often unsatisfied with the outcome of such surgical treatments. In [22], it is reported that only 68.0% of patients experience improved nasal breathing and only 55.9% are satisfied after a septoplasty. To increase the success rate of the surgery, a method is required that suggests shape modifications while maintaining two of the main functionalities of the nose: (i) supplying the lung comfortably with air and (ii) ensuring that the incoming air is sufficiently heated up before entering the lungs.

These two functionalities can be evaluated from a fluid mechanics point of view [9, 10, 12, 14, 25]. At inspiration, the human diaphragm has to provide sufficient energy in the form of negative pressure to drive a flow. This flow experiences a pressure loss due to complex anatomical structures causing flow separation, recirculation zones etc. Comfortable breathing is thus characterized by only a small pressure loss and a small amount of work the diaphragm has to perform. While a narrowed nasal passage increases the pressure loss, it favors the ability to heat up the air due to the higher heat transfer in the vicinity of the wall. That is, considering these two functionalities, a balanced solution needs to be found with a low pressure loss while maintaining a high temperature increase. Hence, a major challenge for a surgeon is to find a suitable shape of the nasal channels that controls the flow in such a way that a compromise between the pressure loss and the temperature increase is found.

Recently, *reinforcement learning* (RL) techniques have shown great potential to control flow in different applications [3, 19, 24]. The main concept behind RL is shown in Figure 1. An agent is trained to interact with an environment. It therefore performs an action that changes the state of the environment and receives feedback from the environment in terms of an observation. An observation may include all information of a state or only a fraction. Subsequent to an action, the state is evaluated with respect to a pre-defined criterion. The agent is then rewarded, depending on its performance in relation to this criterion.

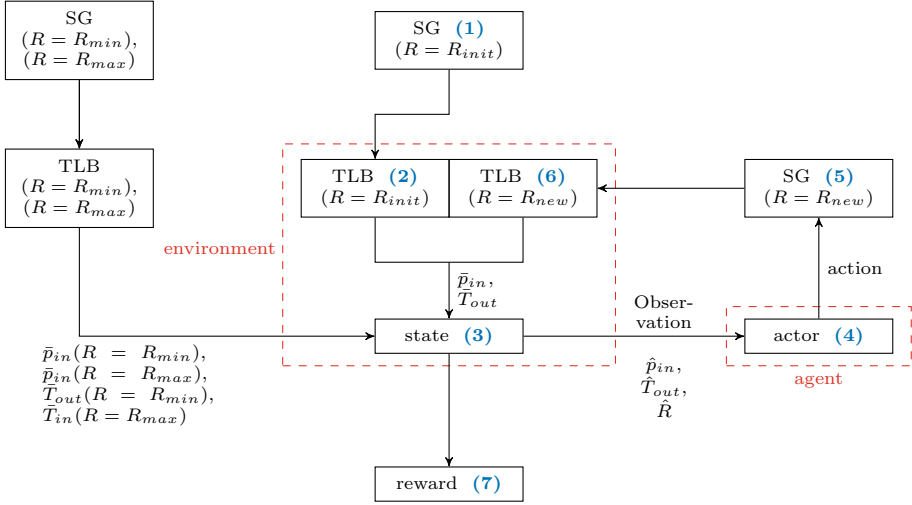
Novati et al. [17] study the swimming kinematics of a leading and a following fish. They use a *deep Q network* (DQN) to derive an energy efficient swimming strategy of the follower. Verma et al. [23] extend the problem from two fishes to a swarm of fishes and improve the algorithm by using recurrent neural networks with *long-short term memory* (LSTM) layers. They report collective energy savings by an efficient use of the wake generated by other swimmers. In [16], an agent of a *trust region policy optimization* (TRPO) algorithm controls the interaction of several fluid jets with rigid bodies. Instead of extracting observations directly from the flow field, a convolutional autoencoder is used to extract low-dimensional features. In [19], an agent of a *proximal policy optimization* (PPO) algorithm is successfully trained to reduce the cylinder drag in a two-dimensional flow by injecting air at the two minimum pressure locations on the cylinder



**Fig. 1.** Interaction between an agent and its environment in an RL algorithm.

contour in an irrotational freestream flow. The PPO algorithm has the stability and reliability of trust region methods. It is easy to implement and tune [21], and has been successfully used in previous flow control applications [19, 20]. It is therefore also used in this study. In [24], wing-like shapes are generated by an *asynchronous advantage actor critic* (A3C) algorithm that considers aerodynamic properties while optimizing the shapes. In these examples, numerical flow simulations are employed to determine the change in an environment after an action. RL algorithms are, however, also used in experimental fluid dynamics. Gueniat et al. [4] apply a *Q learning* strategy to reduce the drag of a cylinder by blowing and suction. In [3], an agent learns to reduce the cylinder drag force by controlling the rotation speed of two smaller cylinders that are located downstream of a main cylinder.

In the present study, a PPO-based RL agent is trained to find the previously described compromise between the pressure loss and the temperature distribution. To simplify the complex shape of a nasal cavity, a two-dimensional channel case is considered. The septum deviation is modeled by a bump that narrows the channel. The agent controls the flow by changing the size of the bump. Feedback on an action is delivered in terms of the aforementioned two fluid mechanical properties, computed by a numerical flow simulation. For the simulation, the *thermal lattice-Boltzmann* (TLB) solver of the **m-AIA** code [11] (formerly known as *Zonal Flow Solver - ZFS*) is employed. The code is developed by the Institute of Aerodynamics and Chair of Fluid Mechanics, RWTH Aachen University. In various studies, the TLB method of **m-AIA** is successfully used to analyze the fluid mechanical properties of human respiration [9, 10, 12, 14, 25]. When the agent changes the bump, the TLB reads the updated geometry and restarts the simulation based on the flow field computed in the previous run. To reduce the degrees of freedom, the agent is only allowed to scale the bump size. Note that investigating a generic two-dimensional channel flow is the first step to applying such methods to more complex cases such as the three-dimensional respiratory flow. The method will be extended to such cases in the future.



**Fig. 2.** Processing chain coupling the PPO RL method to m-AIA. Individual steps of the algorithm described in the text are colored blue. (Color figure online)

The manuscript is structured as follows. Sect. 2 presents the computational methods. This includes explanations on the channel flow setup, the TLB method, the boundary conditions, and the RL algorithm. In Sect. 3, a grid refinement study of the TLB method is conducted and results of the trained agent are reported. A summary and conclusions are given together with an outlook in Sect. 4.

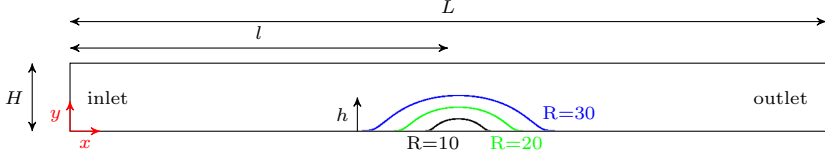
## 2 Numerical Methods

The shape of the bump in the channel is automatically modified using a PPO algorithm, which is coupled to m-AIA. The coupled approach follows the numerical processing chain depicted in Fig. 2. The individual components of this chain are explained in more detail in the following.

Section 2.1 explains the computational setup and how the channel domain is generated by an automated *shape generator* (SG). Subsequently, Sect. 2.2 introduces the TLB method and the relevant boundary conditions. Finally, Sect. 2.3 describes the PPO algorithm and specifies the role of each component of the processing chain.

### 2.1 Computational Domain and Shape Generator

A sketch of the computational domain of the channel with the streamwise and wall-normal directions  $x$  and  $y$  is shown in Fig. 3. It is characterized by the



**Fig. 3.** Two-dimensional channel case setup and examples of the bump for three disturbance factors.

channel height  $H = 20$ , the channel length  $L = 10H$ , and a bump at the lower wall. The bump contour  $\mathbf{B}$  is computed by transforming the bump function

$$b(i) = e^{\left(\frac{1}{(\frac{i}{100})^2 - 1}\right)}, \quad i = \mathbb{Z} \in [-99; 99] \quad (1)$$

into the coordinate system of the channel. That is, the bump function  $b(i)$  is multiplied by the disturbance factor  $R = h \cdot e$ . The control variable  $i$  is multiplied by  $R/100$  and shifted in horizontal direction by adding the shift factor  $l$ , i.e.,

$$\mathbf{B} = \begin{pmatrix} B_x(i) \\ B_y(i) \end{pmatrix} = \begin{pmatrix} \frac{R}{100} i + l \\ R \cdot b(i) \end{pmatrix}. \quad (2)$$

Figure 3 shows examples of the bump for  $R = 10$ ,  $R = 20$ , and  $R = 30$ . For  $R = 0$  the bump vanishes. The agent changes the size of the bump by varying the quantity  $R$ .

The REYNOLDS number  $Re_{ch}$  of the flow is based on the channel height  $H$ , the kinematic viscosity  $\nu$ , and the bulk velocity  $\bar{v}$ . It is set to  $Re_{ch} = 100$ . The flow reaches its maximum velocity  $v_{max}$  at the smallest cross section of the channel. Obviously, for increasing  $R$ ,  $v_{max}$  also increases. The TLB is due to its derivation limited to quasi-incompressible flows at small MACH numbers  $Ma = v_{max}/c_s \ll 1$ , where  $c_s$  is the speed of sound. To avoid reaching the incompressibility limits of the TLB, a maximum disturbance factor of  $R_{max} = 33$  is chosen to guarantee numerically stable and valid simulations.

As illustrated in Fig. 2, the SG automatically generates the channel surface as a function of  $R$ . The output is handed to the TLB method for the computation of the corresponding flow field.

## 2.2 Thermal Lattice-Boltzmann Method

The TLB operates on hierarchical unstructured Cartesian meshes, which are generated using a massively parallel grid generator [13]. Starting with an initial square cell that covers the entire geometry, the mesh is created by iteratively refining the initial cell and all subsequent generated cells. The refinement splits each cell into four equally-sized child cells. At every iteration, cells located outside of the geometry are deleted and parent-child relations are stored. This procedure is repeated until the maximum refinement level is reached. Subsequently,

cells marked for local grid refinement are refined similarly. The hierarchy is represented by a quad-tree.

The TLB method is a highly parallelizable algorithm, which is well suited for efficiently simulating complex flows in intricate geometries [11, 15]. It solves the discrete formulation of the Boltzmann equation [6]

$$f_i(\mathbf{r} + \boldsymbol{\xi}_i \cdot \delta t, t + \delta t) = f_i(\mathbf{r}, t) + \frac{1}{\tau_f} \cdot (f_i^{eq}(\mathbf{r}, t) - f_i(\mathbf{r}, t)), \quad (3)$$

where  $\mathbf{r}$  is the location vector,  $t$  and  $\delta t$  denote the time and the time increment,  $\boldsymbol{\xi}_i$  is the vector of the discrete molecule velocity in the direction  $i \in \{0, \dots, 8\}$ , and  $f_i$  is the corresponding discrete *particle probability distribution function* (PPDF). Furthermore, the relaxation time is formulated as  $\tau_f = \nu/c_s^2$ . The discrete Maxwellian distribution functions  $f_i^{eq}$  are given by

$$f_i^{eq}(\mathbf{r}, t) = \rho \cdot t_p \cdot \left( 1 + \frac{\mathbf{v} \cdot \boldsymbol{\xi}_i}{c_s^2} + \frac{\mathbf{v} \cdot \mathbf{v}}{2c_s^2} \left( \frac{\boldsymbol{\xi}_i \cdot \boldsymbol{\xi}_i}{c_s^2} - \delta \right) \right), \quad (4)$$

where  $\rho$  is the fluid density,  $\mathbf{v}$  is the macroscopic velocity vector,  $\delta$  is the Kronecker delta, and  $t_p$  is a direction-dependent coefficient. The discretization model chosen for the two-dimensional simulations conducted in this study is the well known D2Q9 model [18].

To solve the decoupled energy equation, a passive scalar transport equation of the temperature  $T$  is used. That is, a second set of PPDFs  $g(\mathbf{r}, t)$  describing the diffusion and convection of the temperature is solved on the same lattice. The thermal PPDFs are calculated using [9]

$$g_i(\mathbf{r} + \boldsymbol{\xi}_i \cdot \delta t, t + \delta t) = g_i(\mathbf{r}, t) + \frac{1}{\tau_t} \cdot (g_i^{eq}(\mathbf{r}, t) - g_i(\mathbf{r}, t)), \quad (5)$$

where  $\tau_t = \alpha/c_s^2$  is the thermal relaxation based on the thermal conductivity  $\alpha$  and  $c_s$ . The equilibrium distribution functions  $g_i^{eq}$  are defined similarly to  $f_i^{eq}$  by

$$g_i^{eq}(\mathbf{r}, t) = T \cdot t_p \cdot \left( 1 + \frac{\mathbf{v} \cdot \boldsymbol{\xi}_i}{c_s^2} + \frac{\mathbf{v} \cdot \mathbf{v}}{2c_s^2} \left( \frac{\boldsymbol{\xi}_i \cdot \boldsymbol{\xi}_i}{c_s^2} - \delta \right) \right), \quad (6)$$

where  $\mathbf{v}$  is the macroscopic velocity vector calculated from the moments of the PPDFs  $f_i(\mathbf{r}, t)$ , i.e. by solving

$$\rho = \sum_{i=0}^8 f_i(\mathbf{r}, t) \quad (7)$$

$$\rho \mathbf{v} = \sum_{i=0}^8 \boldsymbol{\xi}_i \cdot f_i(\mathbf{r}, t). \quad (8)$$

The temperature is obtained from the moments of the PPDFs  $g_i(\mathbf{r}, t)$

$$T = \sum_{i=0}^8 g_i(\mathbf{r}, t) \quad (9)$$

and the pressure is calculated using the equation of state for an ideal gas

$$p = \rho \cdot c_s^2. \quad (10)$$

At the inlet, the velocity profile of a fully developed channel flow is prescribed, the density is extrapolated from the inner cells, and the temperature is set to unity. Additionally, a constant pressure is prescribed at the outlet of the channel. Here, the velocity and the temperature are extrapolated from the inner cells. The wall boundary conditions are based on a second-order accurate interpolated bounce-back scheme for the velocity [1] and an isothermal wall scheme for the temperature [8].

### 2.3 Proximal Policy Algorithm

Before the training is started, the SG creates surfaces with  $R = R_{min}$  and  $R = R_{max}$ , and the corresponding flow fields are computed with the TLB method. From these results, the area-averaged pressures and temperatures at the in- and outlets are calculated according to

$$\bar{p}_{\{in,out\}} = \frac{1}{N} \sum_{n=0}^N p_{\{in,out\}}(n) \quad (11)$$

$$\bar{T}_{\{in,out\}} = \frac{1}{N} \sum_{n=0}^N T_{\{in,out\}}(n), \quad (12)$$

where  $n = 0, \dots, N$  is the running index of the cells in the  $y$ -direction at the in- or outlets and  $p_{\{in,out\}}(n)$  and  $T_{\{in,out\}}(n)$  are the pressure and the temperature at cell  $n$ , c.f. Figure 2. In the training of the PPO algorithm, these quantities are used to calculate the area-averaged normalized pressure at the inlet and area-averaged normalized temperature at the outlet

$$\hat{p}_{in} = \frac{\bar{p}_{in}(R) - \bar{p}_{in}(R_{min})}{\bar{p}_{in}(R_{max}) - \bar{p}_{in}(R_{min})} \quad (13)$$

$$\hat{T}_{out} = \frac{\bar{T}_{out}(R) - \bar{T}_{out}(R_{min})}{\bar{T}_{out}(R_{max}) - \bar{T}_{out}(R_{min})}. \quad (14)$$

Two phases take turns in the training. In the first phase, a batch, the agent interacts with the environment. A batch is composed of  $E = 3$  episodes. In each episode, the agent performs  $W = 10$  interactions, yielding a total number of  $K = 30$  interactions per batch. The workflow of the first phase is shown in Fig. 2. Its starting point is step (1), where a surface is created by the SG with  $R_{init} = 10$ . In step (2), the flow field is computed for this surface. The state  $s$  is then complemented with  $\hat{p}_{in}$ ,  $\hat{T}_{out}$ , and  $\hat{R}$  in step (3), where  $\hat{R} = (R - R_{min})/(R_{max} - R_{min})$ . The complete information of  $s$  is passed as an observation to the agent.

The actor network uses the observation as input in step (4). The input layer is followed by two fully connected layers with 64 neurons each, and a final layer with a single neuron. The two fully connected layers have *rectified linear unit* (ReLU) activation functions. The final layer has a tanh activation function, mapping the output to the interval  $[-1; 1]$ . The output then functions as a mean to generate a Gaussian normal distribution with a standard deviation of  $\sigma = 0.5$ .

From this distribution, the normalized action  $a$  is sampled close to the mean of the distribution. The action in form of a change of the disturbance factor  $\Delta R$  is computed by  $\Delta R = \beta a$ , with  $\beta = 3$  to avoid relatively large changes of the disturbance factor and guarantee numerically stable simulations. The new disturbance factor is then determined by  $R_{new} = R + \Delta R$ . If the new disturbance factor exceeds  $R_{min}$  or  $R_{max}$ , an episode ends.

In step (5), the SG generates a new surface with the current disturbance factor. Subsequent to creating the surface, new flow properties are computed using the TLB method in step (6), which restarts from the results of the previous flow field. With these properties, a reward  $r$  is calculated in step (7). The agent is rewarded according to a pre-defined criterion. With the criterion of the current case the agent is trained to change  $R$  such that the pressure loss and the temperature gain between the inlet and outlet are rated equally important.

Note that an equal consideration of both properties is not the only existing criterion definition. In case of a septoplasty, surgeons might weight the properties according to their preferences. The current criterion is expressed by the following reward function

$$r = \frac{(\mu - |\hat{p}_{in} - 1| - \hat{T}_{out})^\kappa}{\mu^\kappa}, \quad (15)$$

where  $\mu = 2$  is the theoretical maximum of the second term of the numerator, keeping the reward positive. The reward is scaled by an exponent  $\kappa$  and normalized by  $\mu^\kappa$ . The exponent must be chosen such that high weights are assigned on higher rewards, but at the same the gradients between lower rewards are not too small. A preliminary study revealed an exponent of  $\kappa = 5$  to function best with the current setup.

If the agent exceeds  $R_{min}$  or  $R_{max}$ , it is punished with  $r = 0$ . If the agent stays within  $R_{min}$  and  $R_{max}$ , and reaches a target zone of  $|\hat{p}_{in} - 1| - \hat{T}_{out} \leq \lambda$ , an episode is stopped. In case of surgical interventions, surgeons could specify  $\lambda$  depending on their tolerance. Here, a target zone of  $\lambda = 0.1$  is chosen. If the agent is not interrupted, the processing chain continues with step (4) and progresses until an episode ends. To assess long-term rewards, for each interaction in an episode, the rewards-to-go  $r_{tg}$  are computed by

$$r_{tg}(w) = \sum_{j=w}^W \gamma^{j-w} r(j). \quad (16)$$

The discount factor  $\gamma$  weights late rewards in an episode. The higher  $\gamma$ , the more weight is assigned to rewards received at later interactions. Here, a discount factor of  $\gamma = 0.95$  is used [2]. For each interaction,  $r_{tg}$  and the probabilities of a normalized action  $a_{prob}$  are collected.



In the second phase, actor and critic networks are trained based on the collected data from the first phase. The critic network differs from the actor network at the final layer, where it has a linear activation. The output of the critic network is the value  $V$ . Before actor and critic losses can be computed, the advantage estimates  $A$  are calculated for all interactions of a batch by

$$A = r_{tg} - V. \quad (17)$$

The actor loss is determined by the loss function

$$L_{act} = \frac{1}{K} \sum_{k=0}^K -\min \left( \frac{a_{prob}(k)}{a_{prob}^{tr}(k)} A(k), \text{clip} \left( \frac{a_{prob}(k)}{a_{prob}^{tr}(k)}, 1 - \epsilon, 1 + \epsilon \right) A(k) \right), \quad (18)$$

where  $a_{prob}^{tr}(k)$  is the probability of an action at interaction  $k$  that is predicted while training the actor network. In the *clip*-function, the probability ratio is clipped at  $1 \pm \epsilon$ , depending on whether  $A(k)$  is positive or negative. A clipping parameter of  $\epsilon = 0.2$  is applied [21]. The critic loss is computed by

$$L_{crit} = \frac{1}{K} \sum_{k=0}^K (V^{tr}(k) - r_{tg}(k))^2, \quad (19)$$

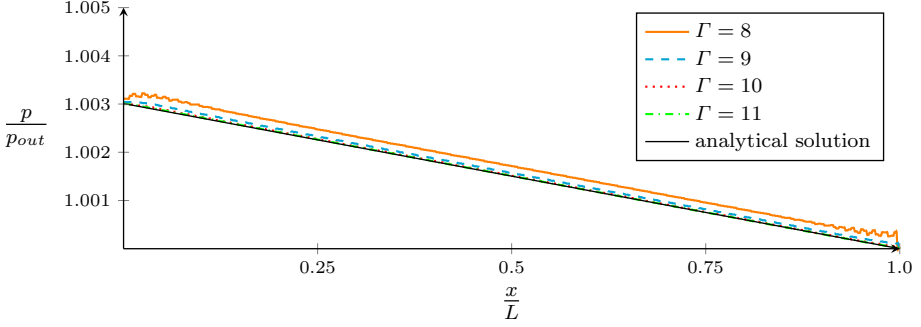
where  $V^{tr}$  is predicted while training the critic network. After each batch, the network updates are repeated five times. Weights are updated by an *adaptive moments* (ADAM) optimizer [7]. The ADAM optimizer adjusts the learning rate  $LR$  by considering an exponentially decaying average of gradients in the previous update steps. The learning rate is initialized with  $LR = 0.005$ .

### 3 Results

In this section, the numerical methods introduced in Sect. 2 are employed to investigate the flow through a channel constricted by a bump, cf. Fig. 3. Before the corresponding results of the performance of the RL algorithm are presented in Sect. 3.2, the simulation method is validated using a generic test cases in the subsequent Sect. 3.1.

#### 3.1 Validation of the Simulation Method

For validation purposes, a two-dimensional fully developed isothermal channel flow is simulated. The TLB approach was already validated by Lintermann et al. in [12].



**Fig. 4.** Pressure distribution along the centerline of the channel for several refinement levels  $\Gamma \in \{8, \dots, 11\}$  at  $Re_{ch} = 100$ . Additionally, the analytical solution is given.

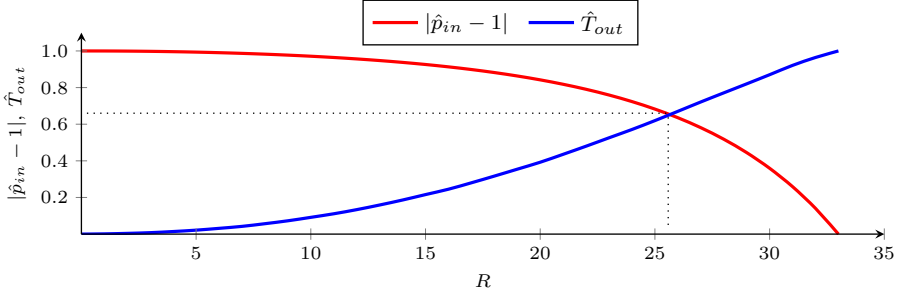
The calculated pressure loss of the fully developed channel flow is compared to the analytical solution given by

$$\Delta p = \frac{8v_{max}\rho\nu L}{H^2} = 12 \frac{\bar{v}^2 \rho}{Re_{ch}} \frac{L}{H}, \quad (20)$$

with the density  $\rho$ , the maximum channel velocity magnitude  $v_{max} = (3/2)\bar{v}$ , and the channel length  $L$ . The solution is derived from the well-known Hagen-Poiseuille law [5]. The channel geometry equals the geometry shown in Fig. 3 with  $R = 0$ . The REYNOLDS number is again set to  $Re_{ch} = 100$ . The boundary conditions at the inlet, the outlet, and the channel walls are set according to the description in Sect. 2.2. The pressure distributions along the centerline of the channel, obtained from flow simulations, are plotted for different mesh refinement levels against the analytical solution in Fig. 4. For refinement level  $\Gamma = 8$ , the channel is resolved by  $256 \times 26$  cells. The number of cells per direction doubles with each refinement level increase. At the maximum considered level  $\Gamma = 11$ , the channel is resolved by  $2,048 \times 208$  cells. Figure 4 shows that the pressure gradient in the middle of the computational domain remains the same for all refinement levels. However, near the in- and the outlets, the pressure curves differ slightly. In these areas, only simulations with a refinement level of  $\Gamma \geq 10$  produce sufficiently accurate results. Therefore, a uniform refined mesh at refinement level  $\Gamma = 10$  is chosen to test the performance of the RL algorithm, see Sect. 3.2.

### 3.2 Comparison of Simulative and RL-Based Results

Simulations are conducted on 2 nodes with 24 cores per node on the supercomputer CLAIX at RWTH Aachen University. In simulations at the beginning of an episode around 3 min are needed to reach a solution that is independent from the initial condition. After that, the simulations time is reduced to 90 s, since simulations can be restarted from previous results and therefore reach such a solution faster. Training time for the actor and critic networks are negligible compared to the simulation time.



**Fig. 5.** Temperature and pressure development of the channel flow for different disturbance factors  $R$  of the bump.

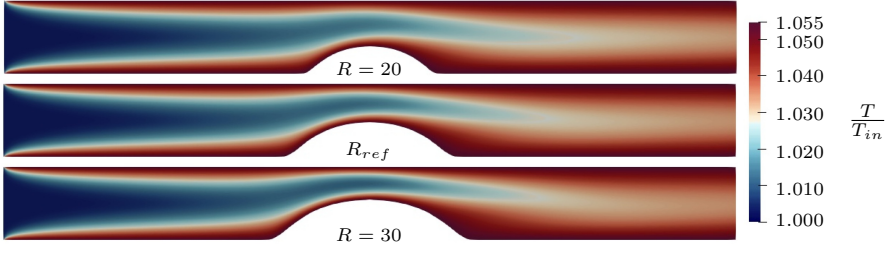
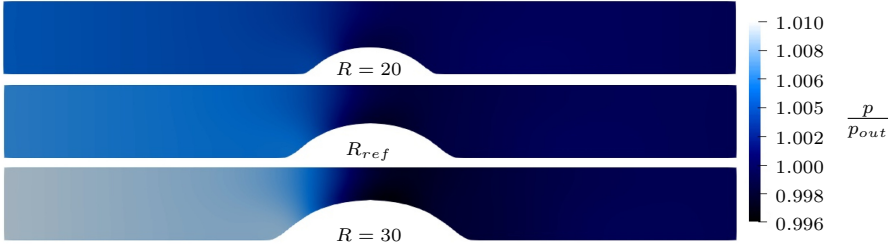
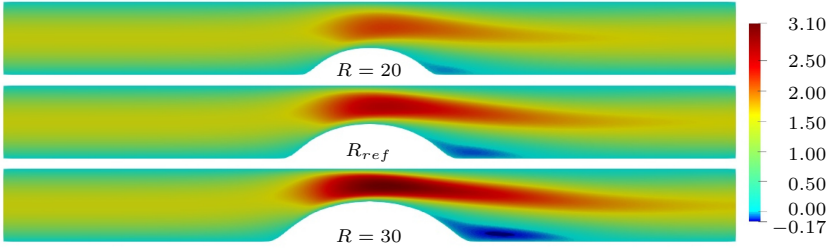
To analyze the behavior of the RL algorithm, a number of 3,400 simulations are conducted, in which the disturbance factor of the bump is continuously increased from  $R = R_{min}$  to  $R = R_{max}$  in steps of 0.01, to achieve an accuracy of two decimal places. For each simulation, the pressure in the form of  $|\hat{p}_{in} - 1|$  and the temperature  $\hat{T}_{out}$  are evaluated. Figure 5 shows that both graphs cut each other for a simulated disturbance factor of  $R_{ref} = 25.58$ . At this location,  $|\hat{p}_{in} - 1|$  and  $\hat{T}_{out}$  have the same value which perfectly fulfills the criterion formulated in Eq. (15). The corresponding results are therefore defined as the reference solution.

The temperature, the static pressure, and the velocity distributions for  $R = 20$ ,  $R_{ref}$ , and  $R = 30$  are shown in Fig. 6. On the one hand, it is obvious from Fig. 6a that the temperature increases in the vicinity of the bump, especially in its wake region. While the increase is upstream of the bump lower, the temperature towards the outlet, downstream of the bump, increases stronger. In case of  $R_{ref}$ , the average temperature at the outlet  $\hat{T}_{out}$  reaches 98.63 % of the wall temperature. On the other hand, a strong drop of the pressure due to the acceleration of the flow in the constricted region is visible in Fig. 6b. With an increasing bump size, the inlet pressure is raised, i.e., a maximum inlet pressure is obtained at  $R = 30$ . Large velocities are observed in the constricted region, see Fig. 6c. Consequently, the maximum streamwise velocity components are also found at  $R = 30$ . For all configurations a recirculation zone with negative streamwise velocities is observed, indicating flow separation. The zone increases with larger bump sizes.

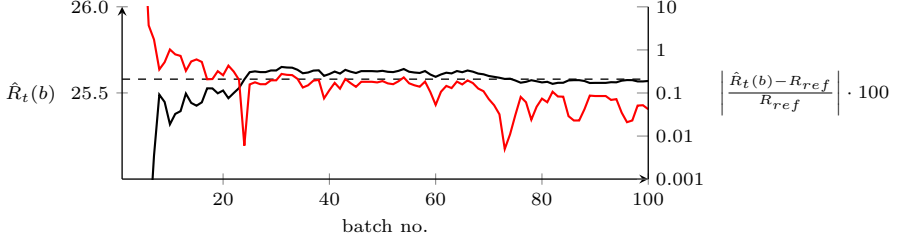
The performance of the agent that uses the processing chain from Fig. 2 is evaluated in terms of the averaged disturbance factor at each batch

$$\hat{R}_t(b) = \frac{1}{E_t(b)} \sum_{b=0}^{E_t(b)} R_t, \quad (21)$$

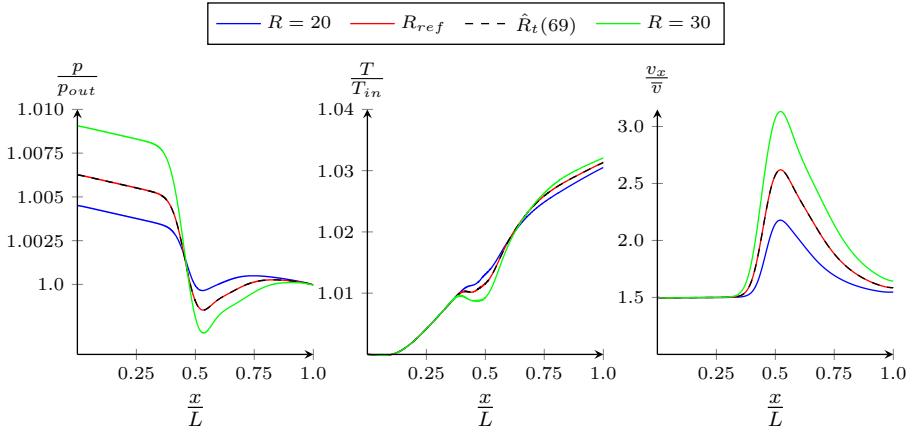
where  $R_t$  is the target disturbance factor in case the target zone is reached in an episode, and  $E_t(b)$  is the number of episodes in which the target zone has been reached from the beginning of the training until batch  $b$ . Figure 7 shows  $\hat{R}_t(b)$

(a) Temperature  $T$  normalized by the inflow temperature  $T_{in}$ .(b) Static pressure  $p$  normalized by the pressure at the outlet  $p_{out}$ .(c) Streamwise velocity  $v_x$  normalized by the bulk velocity  $\bar{v}$ .**Fig. 6.** Simulation results for three channel geometries with  $R \in \{20, R_{ref}, 30\}$ , where  $R_{ref} = 25.58$ .

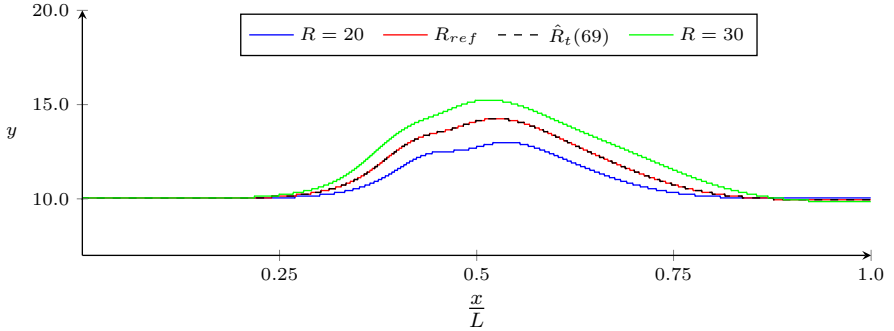
and the error of  $\hat{R}_t(b)$  in relation to  $R_{ref}$ . At batch 6, after 177 simulations, the agent hits the target zone for the first time at  $\hat{R}_t(6) = 24.62$ , yielding an error of 3.7%. After 265 simulations in 10 batches, the error remains below an acceptable error of 1.0% for the rest of the training. The averaged disturbance factor is  $R_t(10) = 25.32$ . At 69 batches, or 1,450 simulations, an averaged disturbance factor of  $R_t(69) = 25.61$  misses  $R_{ref}$  by less than only 0.1%. From here on the error remains below 0.1% for the rest of the training.



**Fig. 7.** Averaged disturbance factor from the beginning of the training to each batch  $\hat{R}_t(b)$  (black) and relative error of  $\hat{R}_t(b)$  to  $R_{ref}$  (red) for 100 batches. The dashed black line represents  $R_{ref}$ . (Color figure online)



**(a)** Normalized pressure (l.), temperature (c.), and streamwise velocity (r.) distributions along the lines shown in Fig. 8b.



**(b)** Lines following the maximum streamwise velocity component.

**Fig. 8.** Quantitative results for different  $R \in \{20, R_{ref}, \hat{R}_t(69), 30\}$ .

A further quantitative analysis is performed by comparing the distributions of the static pressure, the temperature, and the streamwise velocity along the line with the maximum streamwise velocity component for  $R \in \{20, R_{ref}, \hat{R}_t(69), 30\}$ . The results are depicted in Fig. 8a and the corresponding lines in Fig. 8b. Plots of all fluid mechanical properties for the reference solution and the RL-based method show an almost perfect alignment proving the RL-based processing chain to be a valid alternative to using a brute-force parameter sweeping.

## 4 Summary, Conclusion, and Outlook

A PPO algorithm is combined with a TLB simulation method to change the shape of a heated narrowed channel, while considering the pressure loss and temperature difference between the inflow and outflow region. The algorithm controls the flow by changing the disturbance factor of a bump. It is capable of finding a target disturbance factor that weights both fluid mechanical properties equally. Averaging the target factor over multiple tries yields a deviation from the reference disturbance factor, which is determined iteratively with 3,400 numerical simulations, of less than 1.0% after 265 simulations in 10 batches, and less than 0.1 % after 1,450 simulations in 69 batches. A comparison of the temperature, pressure, and streamwise velocity distributions between the predicted and the reference solutions along the line of the maximum velocity component in streamwise direction shows only negligible differences.

Surgical interventions in rhinology often include the removal of anatomical structures. Such removals influence the capability to breathe comfortably. Ideally, the pressure loss and the heating capability of a nasal cavity are balanced such that respiration is energy efficient and the air reaches body temperature.

The current results build the foundation for applying RL algorithms to shape optimization problems in rhinology. To close the gap between the simplified channel flow and a nasal cavity flow, the method will be extended to more complex flow configurations such as two-dimensional domains with multiple bumps or three-dimensional geometries like a stenotic pipe. The increase of the domain complexity will lead to enlarged action and state spaces, and variations of the actor and critic networks can be investigated. Furthermore, the PPO algorithm will be compared with other RL algorithms. It is the ultimate vision, to develop a tool that assists surgeons in their decision making process.

**Acknowledgments.** The research leading to these results has been conducted in the CoE RAISE project, which receives funding from the European Union’s Horizon 2020 – Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733. The training of the RL algorithm has been performed on two nodes of the CLAIX HPC system at RWTH Aachen University. Each node is equipped with two Intel Xeon Platinum 8160 Processors “SkyLake”, clocked at 2.1GHz, with 24 cores each and 192 GB main memory per node. The authors gratefully acknowledge the computing time granted by the JARA Vergabegremium and provided on the JARA Partition part of the supercomputer CLAIX at RWTH Aachen University. This

work was performed as part of the Helmholtz School for Data Science in Life, Earth and Energy (HDS-LEE).

## References

1. Bouzidi, M., Firdaouss, M., Lallemand, P.: Momentum transfer of a Boltzmann-lattice fluid with boundaries. *Phys. Fluids* **13**(11), 3452–3459 (2001). <https://doi.org/10.1063/1.1399290>
2. Fakoor, R., Chaudhari, P., Smola, A.J.: P3O: olicy-on policy-off policy optimization. In: Adams, R.P., Gogate, V. (eds.) *Proceedings of the 35th Uncertainty in Artificial Intelligence Conference. Proceedings of Machine Learning Research*, vol. 115, pp. 1017–1027. PMLR, 22–25 July 2020
3. Fan, D., Yang, L., Triantafyllou, M., Karniadakis, G.: Reinforcement learning for active flow control in experiments, March 2020
4. Guéniat, F., Mathelin, L., Hussaini, M.Y.: A statistical learning strategy for closed-loop control of fluid flows. *Theor. Comput. Fluid Dyn.* **30**(6), 497–510 (2016). <https://doi.org/10.1007/s00162-016-0392-y>
5. Hagenbach, E.: Über die bestimmung der zähigkeit einer flüssigkeit durch den ausfluss aus röhren. *Poggendorf's Annalen der Physik und Chemie* **108**, 385–426 (1860)
6. He, X., Luo, L.S.: Theory of the lattice Boltzmann method: from the Boltzmann equation to the lattice Boltzmann equation. *Phys. Rev. E* **56**(6), 6811–6817 (1997). <https://doi.org/10.1103/PhysRevE.56.6811>
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
8. Li, L., Mei, R., Klausner, J.F.: Boundary conditions for thermal lattice Boltzmann equation method. *J. Comput. Phys.* **237**, 366–395 (2013). <https://doi.org/10.1016/j.jcp.2012.11.027>
9. Lintermann, A., Meinke, M., Schröder, W.: Investigations of the inspiration and heating capability of the human nasal cavity based on a lattice-Boltzmann method. In: *Proceedings of the ECCOMAS Thematic International Conference on Simulation and Modeling of Biological Flows (SIMBIO 2011)*, Brussels, Belgium (2011)
10. Lintermann, A., Meinke, M., Schröder, W.: Investigations of nasal cavity flows based on a lattice-Boltzmann method. In: Resch, M., Wang, X., Bez, W., Focht, E., Kobayashi, H., Roller, S. (eds.) *High Performance Computing on Vector Systems 2011*, pp. 143–158. Springer, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-22244-3>
11. Lintermann, A., Meinke, M., Schröder, W.: Zonal Flow Solver (ZFS): a highly efficient multi-physics simulation framework. *Int. J. Comput. Fluid Dyn.* 1–28 (2020). <https://doi.org/10.1080/10618562.2020.1742328>
12. 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.combiomed.2013.09.003>
13. Lintermann, A., Schlimpert, S., Grimmen, J., Günther, C., Meinke, M., Schröder, W.: Massively parallel grid generation on HPC systems. *Comput. Methods Appl. Mech. Eng.* **277**, 131–153 (2014). <https://doi.org/10.1016/j.cma.2014.04.009>
14. Lintermann, A., Schröder, W.: A hierarchical numerical journey through the nasal cavity: from nose-like models to real anatomies. *Flow, Turbul. Combust.* **102**(1), 89–116 (2017). <https://doi.org/10.1007/s10494-017-9876-0>

15. Lintermann, A., Schröder, W.: Lattice–Boltzmann simulations for complex geometries on high-performance computers. *CEAS Aeronaut. J.* **11**(3), 745–766 (2020). <https://doi.org/10.1007/s13272-020-00450-1>
16. Ma, P., Tian, Y., Pan, Z., Ren, B., Manocha, D.: Fluid directed rigid body control using deep reinforcement learning. *ACM Trans. Graph.* **37**(4), 1–11 (2018). <https://doi.org/10.1145/3197517.3201334>
17. Novati, G., Verma, S., Alexeev, D., Rossinelli, D., van Rees, W.M., Koumoutsakos, P.: Synchronisation through learning for two self-propelled swimmers. *Bioinspiration Biomimetics* **12**, 3 (2017)
18. Qian, Y.H., D’Humières, D., Lallemand, P.: Lattice BGK models for Navier-stokes equation. *Europhys. Lett. (EPL)* **6**, 479–484 (1992). <https://doi.org/10.1209/0295-5075/17/6/001>
19. Rabault, J., Kuchta, M., Jensen, A., Réglade, U., Cerardi, N.: Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *J. Fluid Mech.* **865**, 281–302 (2019). <https://doi.org/10.1017/jfm.2019.62>
20. Rabault, J., Kuhnle, A.: Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach. *Phys. Fluids* **31**, 094105 (2019). <https://doi.org/10.1063/1.5116415>
21. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms, July 2017
22. Toyserkani, N., Frisch, T.: Are too many septal deviations operated on? A retrospective patient’s satisfaction questionnaire with 11 years follow-up. *Rhinology* **50**, 185–190 (2012). <https://doi.org/10.4193/Rhino11.218>
23. Verma, S., Novati, G., Koumoutsakos, P.: Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proc. Natl. Acad. Sci.* **115**(23), 5849–5854 (2018). <https://doi.org/10.1073/pnas.1800923115>
24. Viquerat, J., Rabault, J., Kuhnle, A., Ghraieb, H., Hachem, E.: Direct shape optimization through deep reinforcement learning, August 2019. <https://doi.org/10.13140/RG.2.2.19572.50566>
25. Waldmann, M., Lintermann, A., Choi, Y.J., Schröder, W.: Analysis of the effects of MARME treatment on respiratory flow using the lattice-Boltzmann method. In: Dillmann, A., Heller, G., Krämer, E., Wagner, C., Tropea, C., Jakirlić, S. (eds.) *DGLR 2018. NNFMMMD*, vol. 142, pp. 853–863. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-25253-3\\_80](https://doi.org/10.1007/978-3-030-25253-3_80)