Studienarbeit Nr. 836

Optimal Sensor Placement for Active Flow Control with Deep Reinforcement Learning

Tom Krogmann

Issued by: Prof. Dr.-Ing. R. Radespiel
Institut für Strömungsmechanik
Head of Institute: Prof. Dr.-Ing. R. Radespiel
Technische Universität Braunschweig

Supervisor: Dr.-Ing. Andre Weiner (TU Braunschweig)

Publication: 12.2022

Statutory Declaration in Lieu of an Oath

Hereby I, Tom Krogmann, declare that I have authored this thesis independently and without any external help and that I have not used any aids other than those indicated. All thoughts taken directly or indirectly from external sources are properly denoted as such.

Braunschweig, 13.12.2022

Abstract

Deep reinforcement learning over the years has proven its ability to learn strategies for actively controlling high dimensional and nonlinear fluid flows. The fluidic pinball, a triangular arrangement of 3 individually rotateable cylinders, represents an ideal benchmark study which, despite being cheap to evaluate, covers a range of complex flow dynamics from symmetric to chaotic vortex shedding. A crucial part for industrial implementation of active flow control is to minimize the required flow sensors, which is the task of this study. Therefore 3 optimal sensor placement methodologies for active flow control of the fluidic pinball, based on unactuated and actuated computational fluid dynamics simulations, were investigated. Optimal sensors were estimated on the unactuated flow by reconstructing the flow field with proper orthogonal decompositon modes and by computing random forest importance scores in a latin-hypercube sampling approach combined with a clustering algorithm. The performance of the obtained optimal sensors is compared with the attention mechanism which directly incorporates learning optimal sensors into the reinforcement learning loop. The results obtained for the asymmetric vortex shedding case reveal that effective drag reduction is achieveable with all sensor placement methodologies which comes at the cost of an increased lift coefficient. The method that works best is the attention approach, being able to achieve 80.98% drag reduction where at the same time a dense baseline sensor placement of 476 sensors was reduced to only 7 sensors without losing performance. Therefore it can be concluded that it is beneficial to integrate choosing optimal sensor into the deep reinforcement learning process. Moreover attention offers great potential to even further improve the active flow control performance and also comes with the nice side effect of getting insights into the agents flow control strategy.

Contents

Nomenclature

Latin symbols

A_t	Action at current timestep
a	Temporal coefficients of proper orthogonal decomposition modes
b	Bias
c	Number of response values
d	Response value
$\hat{d}$	Predicted response value
c_D	Drag coefficient
$\hat{c}_D$	Predicetd drag coefficient
c_L	Lift coefficient
$\hat{c}_D$	Predicetd lift coefficient
C	Measurement matrix
D	Cylinder diameter
e	Output vector of first attention layer
f	Output vector of second attention layer
F	Total aerodynamic force
F_D	Drag force
F_L	Lift force
F_P	Pressure force
F_S	Shear stress force
$G(\tau)$	expected discounted return
H	Snapshot matrix
h	Column vector of snapshot matrix
I	Importance score
i	Impurity
J_{ij}	Pearson correlation coefficient
l_r	Learning rate
m	Number of snapshots
n	Number of sensors
n_x, n_y	Unit normal vector in x- and y-direction
o	Observations
p	pressure
$\overline{p}$	Mean pressure
p'	Pressure fluctuation
p^*	dimensionless pressure
r	rank
R_t	Reward at current timestep
S_t	State at current timestep

Re	Reynolds number
s	Splitting value
S_t	State at current timestep
t	Time
t^*	Dimensionless time
T	Terminal state
U, V	Orthogonal matrix in the singular value decomposition
u, v	velocity in x- and y-direction
u^*	dimensionless velocity in x-direction
$V^\pi(S)$	State-value-function
W	Weights of neural network layer
x, y, z	Cartesian coordinates
x^*, y^*, z^*	Dimensionless cartesian coordinates
Z_t	Cell thickness in z-direction

Greek Symbols

α	Parameter of the beta distribution
β	Parameter of the beta distribution
γ	Discount factor
δ	Lift penalization coefficient
ϵ	Clipping parameter
η	Value network weights
θ	Policy network weights
Θ	Inversion Matrix
κ	Attention weights
λ	Singular values
μ	Mean value
∇	Nabla operator
ν	Kinematic viscosity
ρ	Density
σ	Standard deviation
τ	Trajectory
ϕ	Rotation angle
Φ	Proper orthogonal decomposition modes
χ	Index subset of optimal reconstruction sensors
Σ	Diagonal matrix containing singular values
ω	Rotational velocity
Ω	Rotation axis vector

Indices

A	Cylinder A
B	Cylinder B
C	Cylinder C
c	controlled
max	Maximum vale
r	rank r truncation

ref	Reference
t	Time
tan	Tangential
u	uncontrolled
∞	Free stream

Abbreviations

AFC	Active Flow Control
AI	Artifical Intelligence
CFD	Computational Fluid Dynamics
DRL	Deep Reinforcement Learning
GAE	Generalized Advantage Estimate
LHS	Latin Hypercube Sampling
MDA	Mean Decrease in Accuracy
MDI	Mean Decrease in Impurity
ML	Machine Learning
MSE	Mean Squared Error
NN	Neural Network
PDF	Probability Density Function
PI	Permutation Importance
POD	Proper Orthogonal Decomposition
PPO	Proximal Policy Optimization
RF	Random Forest
RL	Reinforcement Learning
RSS	Residual Sum of Squares
STD	Standard Deviation
SVD	Singular Value Decomposition

Chapter 1

Introduction

1.1 Motivation

Global air transport has ever had a high demand and was expected to triple between 2020 and 2050 before the COVID-19 pandemic hit the world [12]. Meanwhile climate change puts pressure on not only aircraft manufactures but also on the transport industry to develop more energy efficient ways of traveling. Several technologies are currently investigated including electrification, sustainable aviation fuels as well as the application of hydrogen for propulsion. A further promising approach is to control the flow around an object, e.g. an aircraft wing. By means of passive flow control, structural changes like vortex generators or, inspired by the skin of sharks, riblets are applied to surfaces in the flow effectively reducing the drag force [36]. Contrary in active flow control (AFC) external energy is added to the flow, e.g. by synthetic jets [22]. Finding control laws for AFC with equation-based analysis of fluids is facing serious challenges like high dimensionality and nonlinearity which defy closed-form solutions [6]. During the last few years, data-driven approaches applying machine learning (ML) algorithms for flow control have achieved great progress, where especially deep reinforcement learning (DRL) accounted for a valid solution to learn control laws. DRL models for AFC are trained within computational fluid dynamics (CFD) simulations where at every time step the information about the whole flow field is accessible. However in practical applications, the flow field can only be measured by a finite amount of sensors. Training control laws with simulations is a great advantage, since optimal sensors can already be found before conducting experiments. Therefore to enable AFC by means of DRL to real world systems, finding a reduced subset of potential sensors is a crucial task which is investigated in this study.

1.2 Related Work

Artificial Intelligence (AI) and ML are rapidly growing research areas which started to be applied in many engineering disciplines that are subject to high dimensional and non linear differential equations. Driven by a steady increase of computational power not only ML but also CFD has gained more and more importance in the daily business of engineers producing vast amounts of data. Therefore the potential of combining ML and CFD was only a matter of time not only enabling DRL to the field of fluid mechanics but also using ML e.g. for shape optimization [35].

The first successful application of DRL in the field of active flow control was published by Rabault et al. [22] in 2019 where the proximal policy optimization (PPO) algorithm was used to train a DRL agent with the task of controlling the von Kármán Vortex Street appearing in the wake of a 2D cylinder flow at $Re = 100$. For the actuation two synthetic jets were used at at the top

and bottom of the cylinder where the mass flow rate was controlled such that no mass is created oder removed in the simulation. The agent was able to reduce the drag by approximately 8% upon varying the mass flow rate of the jets [22].
Rabault et al. [23] further improved the setup by applying parallelization to the CFD simulations and adapting the PPO approach such that independent trajectories were sampled. Depending on the number of parallelized simulations, a speed up of 20-60 compared to the serial training was achieved.
A different approach concerning the type of control for AFC on the 2D cylinder flow for $Re = 100$ was published by Tokarev et al. [34] in 2020. Instead of mass blowing jets they used to rotate the cylinder with a time-dependent angular velocity around its axis. Two different reward functions were tested accounting for 14% and 16% drag reduction where the maximum amplitude of the angular velocity was 8% and only 0.8%, respectively, of the incoming flow. Hereby they proved that higher rotational velocities are not necessarily beneficial for the control task.
The first approach to optimally place sensors for AFC with DRL was introduced by Paris et. al [20] in 2021. In this study the 2D flow of the cylinder was investigated using two synthetic jets as actuators and pressure sensors in the wake of the cylinder flow. The work presents the S-PPO-CMA algorithm that optimally places sensors in the wake in a 3x5 grid such that redundant or non-necessary information is discarded. The results revealed that using more than 5 sensors leads to equivalent drag reductions and no sensors in the symmetry line are chosen. The optimal sensor placement obtained in this study is shown in figure 1.1. As a comparison the

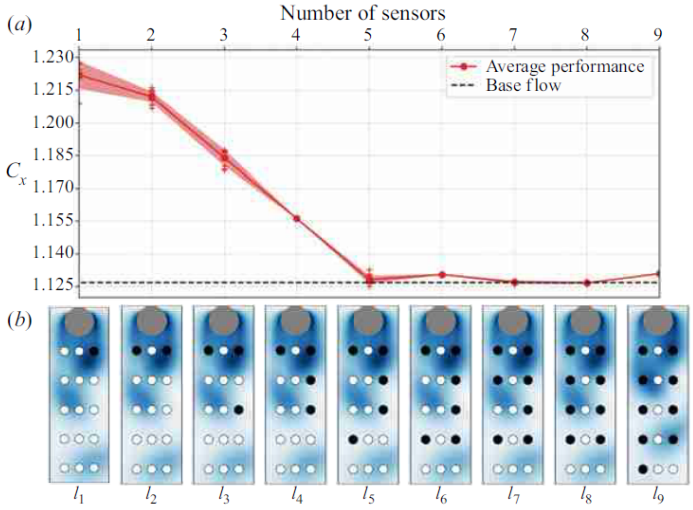

Figure 1.1: Optimised sensor layout for AFC past a cylinder, figure 20 from [20]

resulting optimal sensors were plotted into the first 3 proper orthogonal decomposition (POD) modes which is a frequently used sensor placement method in the field of fluid dynamics placing optimal sensors in regions where the modes are strong. It could be shown that S-PPO-CMA places the sensors accordingly to the minima/maxima of the modes. [20].
In the field of compressed sensing optimal sensor placement can be conducted by reconstructing a high dimensional signal from a low dimensional representation in a POD basis. This methodology was introduced by Brunton et al. [17] in 2017. Pointwise sensors are estimated using a greedy sampling approach with matrix QR-pivoting that optimally conditions the inversion of a measurement matrix. The aforementioned approach was used to optimally reconstruct the flow field of a 2D cylinder flow at $Re = 100$ from a numerical simulation where the mesh consisted of approximately 90000 grid points. A total of 100 flow snapshots was used to establish a low rank POD basis considering the first 42 modes to estimate optimal reconstruction sensors. The approach was able to reduce the 90000 grid points to 42 sensors in order to optimally reconstruct the flow field.

The most recent study of determining optimal sensors for AFC with DRL on the 2D cylinder flow was published in 2022 by Wei et al. [37] where the attention mechanism was used to incorporate learning the optimal sensors in the DRL loop. The authors add a series of additional layers to the value network that learn to ignore irrelevant sensors and to only attend to the relevant ones. The training was carried out using 151 sensors placed at the cylinders surface and in the wake where a reduction to 5 sensors was accomplished without losing performance in the drag reduction [37], as can be seen denoted as D-AFS in figure 1.2.

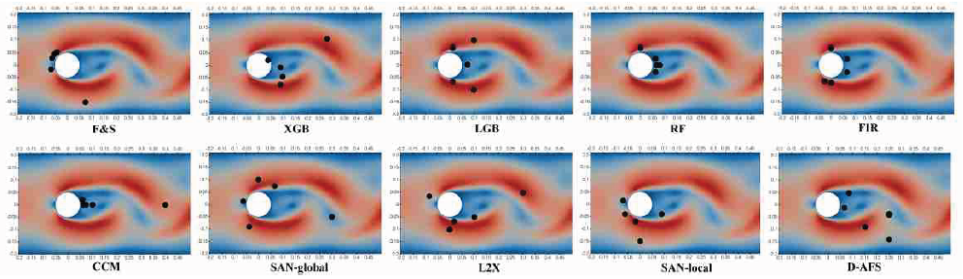

Figure 1.2: Optimal Sensor Placements of various methods for the single cylinder, figure 4 from [37]

In 2016 Richard Semaan [30] published the first article on using random forest variable importance for optimal sensor placement in the field of flow control. An optimal sensor placement for closed-loop flow control of an airfoil equipped with a coanda flap containing 96 sensors on the upper surface was tested in different actuations conditions. The control was established varying the type of the sensor and the flow variables to be predicted. The obtained sensor positions were compared against the standard method of maximum POD mode amplitude location and with a brute force approach, showing good agreement with the provided methodology. Further the results revealed that the choice of the sensors is mainly depending on the type of the sensor and the flow conbdition, whereas the prediction variable has limited influence [30].
Since the benchmark case of controlling the 2D cylinder flow is a rather simple and generic example not representing real world high dimensional flow conditions, a new benchmark problem for AFC considering a triangular arrangement of 3 individually rotateable cylinders was introduced by Noack et al. [19]. The so called fluidic pinball has a fairly simple geometry making it cheap to evaluate but at the same time covers a range of complex flow dynamics therefore providing an optimal candidate to further progress research on AFC.
The first study of AFC with DRL on the fluidic pinball was presented by Marius Holm [18] in 2020 where PPO was applied to learn control laws with the goal of reducing and increasing the drag of the 2D flow in the wake of the cluster of the pinball cylinders. PPO training was carried out for the two Reynolds number $Re = 100$ and $Re = 150$ accounting for asymmetric vortex shedding and chaotic vortex shedding, respectively. The results show that for the asymmetric vortex shedding case an agent was trained effectively reducing the drag by approximately 28% and increasing the drag by up to 45% compared to the uncontrolled flow. In contrast at the higher Reynolds number of $Re = 150$ control laws reducing the drag by $\approx 32\%$ and increasing the drag by up to 65% were established. Further the DRL control laws were compared to simpler control functions like constant and sinusoidal actuations where in most cases comparable but slightly worse performance was observed.

1.3 Procedure

This thesis is mainly based on the drlfoam framework [38], which is a summary of the PPO implementations used in the previous studies of Darshan Thummar [33], Fabian Gabriel [11] and Erik Schulze [28]. The framework is extended by adding a test case for the fluidic pinball.
The overall objective of this study is to compare three optimal sensor placement methodologies based on unactuated and actuated CFD simulations regarding the AFC performance of the fluidic pinball on three different Reynolds numbers. In a first step three Reynolds numbers covering the symmetric, asymmetric and chaotic vortex shedding regime are chosen within a preliminary analysis of the flow dynamics and simulations for the aforementioned cases are executed. Based on the produced simulation data, random forest importance scores are computed and a methodology for reconstructing the flow field was utilized to optimally place sensors for the unactuated flow. After having adopted the PPO implementation to the fluidic pinball, the setup is validated within a hyperparameter study on the action magnitudes and the reward function, using a total of 476 pressure sensors as a baseline. The third sensor placement approach using the attention mechanism targets to directly choose optimal sensors for the actuated flow within the DRL loop. In the final step, all the different sensor placements are used to learn control laws for each flow regime, concluding by comparing the results and investigating whether the sensors estimated for the unactuated flow are still optimal in the actuated flow dynamics.

Chapter 2

Theoretical Foundations

2.1 Singular Value Decomposition

The singular value decomposition (SVD) is a matrix factorization technique that generalizes the eigendecomposition to rectangular matrices. Analyzing computational fluid dynamics (CFD) data consists of handling a time series of a fluid property at a grid of spatial locations. This data may be structured with column vectors $\mathbf{h}_k \in \mathbb{R}^n$ where h is the examined fluid flow variable at the k-th time step and n is the number of spatial points. This vector is also called a snapshot of the data, usually with $n \gg m$, and organizing all available snapshots results in a tall and skinny matrix $\mathbf{H} \in \mathbb{R}^{n \times m}$

$$\mathbf{H} = \begin{bmatrix} | & | & & | \\ \mathbf{h}_1 & \mathbf{h}_2 & \cdots & \mathbf{h}_m \\ | & | & & | \end{bmatrix} \tag{2.1}$$

Using the SVD the data matrix $\boldsymbol{H}$ can be decomposed into 3 matrices

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \approx \mathbf{U}_r \Sigma_r \mathbf{V}_r^T \tag{2.2}$$

where $\mathbf{U} \in \mathbb{R}^{n \times n}$ is an orthogonal matrix, $\mathbf{\Sigma} \in \mathbb{R}^{n \times m}$ is a diagonal matrix with m singular values $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m \geq 0$ on the diagonal and $\mathbf{V} \in \mathbb{R}^{m \times m}$ is an orthogonal matrix as well. The columns of $\boldsymbol{U}$ are called left singular vectors describing spatially correlated modes of the flow property and the columns of $\boldsymbol{V}$ are right singular vectors related to temporal correlations. The SVD is a numerically stable method equivalent to the Proper Orthogonal Decomposition (POD) which was introduced to extract dominant coherent structures from turbulent flows. In POD the spatial mode matrix $\boldsymbol{U}$ is denoted as Ψ and the product of ΣV^T is referred to the temporal coefficients matrix. Most commonly before computing the SVD the temporal mean $\overline{H}$ of $\boldsymbol{H}$ is substracted such that the SVD is carried out on the matrix carrying the fluctuating components. An important feature of the SVD is that it can be used for dimensionality reduction of the data matrix with an optimal low-rank approximation of it. Therefore a rank r truncation is carried out according to the singular values which yield a measure of the flow's total energy of the the corresponding modes. With this only the $r \times r$ sub-matrix $\mathbf{\Sigma}_r$ and the first r columns of the spatial- and temporal modes $\boldsymbol{U}_r$ and $\boldsymbol{V}_r$, respectively, are kept. Often only a few of the modes driving the majority of the energy are sufficient for reconstruction and the rest can be omitted [39, 5, 15].

2.2 Random Forest Regression

Random Forests are non-linear regression tools which consist of an ensemble of regression trees. The prediction of a Random Forest is the average over the predictions of its trees. Each regression tree is unique utilizing bootstrap aggregation (bagging) on the input data. This sampling method refers to training each tree only on a subset of the training data, resulting in de-correlated trees that increase the overall prediction accuracy. The training data of a regression tree consists of n inputs and c response values for each of m observations: (h_i, d_i) for $i = 1, 2, \ldots, m$ with $h_i = (h_{i1}, h_{i2}, \ldots, h_{ic})$ and $d_i = (d_{i1}, d_{i2}, \ldots, d_{ic})$. Trees are built with a greedy top-down approach which is called recursive binary splitting. Starting from the root node at the top of the tree, consisting of all available observations, the tree is successively split into two new nodes p_L and p_R, where the split is defined on a splitting variable H_j and a split value s

$$p_L(j, s) = \{\boldsymbol{H} \mid H_j \leq s\} \text{ and } p_R(j, s) = \{\boldsymbol{H} \mid H_j > s\} \tag{2.3}$$

The goal is to reduce the residual sum of squares (RSS) in both nodes after the current split. Therefore the approach is called greedy since the tree is not considering splits that lead to better predictions in subsequent splits because this would be computationally infeasible. A split seeks for the minimization of the following equation

$$\sum_{i:h_i \in p_L(j,s)} \left(d_i - \hat{d}_{p_L}\right)^2 + \sum_{i:h_i \in p_R(j,s)} \left(d_i - \hat{d}_{p_R}\right)^2 \tag{2.4}$$

where $\hat{d}_{p_L}$ and $\hat{d}_{p_R}$ are the predictions in node p_L and p_R, respectively. The prediction is estimated as the average of the reponse values for all observations inside a node. This procedure is repeated until a stopping criterion is reached either defining hyperparameters for the spliting rules or the tree architecture. Nodes that are not further split are called leaf nodes. On inference the input data is processed from top to bottom of the tree according to the constructed splitting rules, until reaching a leaf node. The prediction of the tree then is the average over the remaining observations of the response function in that leaf node. An example of a regression tree is given in figure 2.1 since an advantage of such trees is the ease of visual interpretability [13, 14].

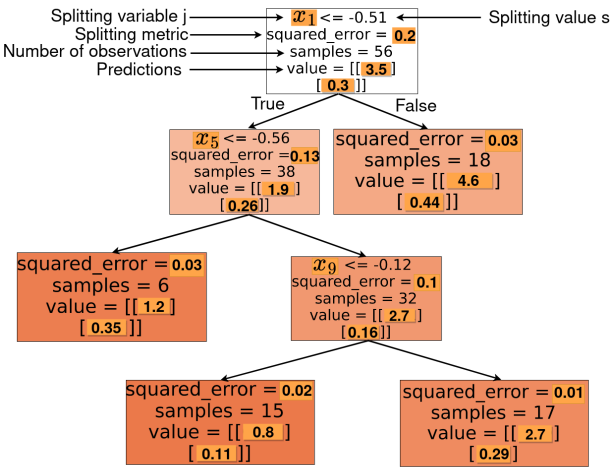

Figure 2.1: Regression Tree

2.3 Reinforcement Learning

Reinforcement learning (RL) is one of the 3 machine learning paradigms and can be used to learn control laws for active flow control by trial-and-error. RL refers to the task of sequential decision making where an agent interacts with an environment. At a given timestep t, depending on the observed state S_t of the environment, the agent executes an action A_t, receives a reward signal R_t from it and transitions into the next state S_{t+1}, as depicted in figure 2.2. Thereby the agent forms a strategy, called policy π_k, which is a learned mapping from a state to an action. The goal of the learning process of the agent is encoded in the reward function which the agent tries to maximize. The reward is a measure of the goodness of an action, where the higher the reward, the better the action is that has been taken in the current state.

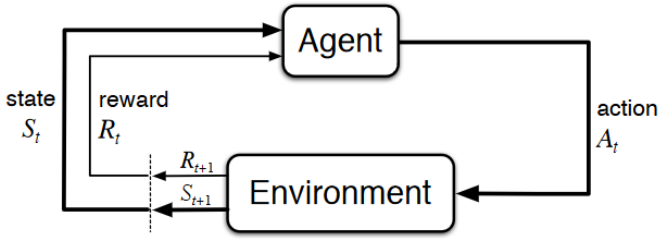

Figure 2.2: Interaction between the RL Agent and its Environment, figure 3.1 from [32]

While the reward defines the instantaneous measure of what good actions are, the ultimate goal of the agent is to learn the value of being in a certain state to maximize the sequence of rewards over the long term, also called the cumulative reward or return. Long term in this case is defined by a set of timesteps t the agent is interacting with the environment, until a terminal state T is reached. The amount of timesteps is defined as the horizon and the the collection of all states, rewards, actions and consecutive states during one interaction episode is called the trajectory τ. Therefore the return is computed over a trajectory and weighted with a discount factor $\gamma \in [0, 1]$, resulting in the expected discounted return $G(\tau)$ where usually $\gamma = 0.99$ is chosen.

$$G(\tau) = \sum_{t=0}^{T} \gamma^t R_t \tag{2.5}$$

By exponentially weighting the rewards with γ over the timesteps, receiving the same reward earlier on results in a higher return, therefore expressing some kind of urgency to the agent.
One of the main challenges in RL is the balance between exploration and exloitation. To find good policies, the agent at first has to explore the state space of the environment to search which state corresponds to a high reward and to learn actions that result in such states. By reducing the probability of taking actions associated with low rewards, the agent further exploits its knowledge and puts more confidence into its learned strategy. However without exploration at the beginning the agent might not find the global optimal policy because it has not searched for actions that might result in even higher rewards than the current strategy [32, 35].

Deep Reinforcement Learning

If environments consist of high dimensional and continous state- and action-spaces, the RL agent will only be able to explore a subset of it due to computational limitations. Therefore generalization is necessary from the encountered states and actions. This is done using function approximation with neural networks (NN) which is also referred to as Deep Reinforcement Learning (DRL). Within DRL, the value of a state considering the current policy, the state-value-function $V^{\pi}(S)$, and the policy are learned by training a neural network each [35, 32].

Chapter 3

Fluidic Pinball

In previous studies, Rabault et al. showed the applicability of learning a control law with DRL to stabilize vortex shedding and reduce drag on a single cylinder, a generic and well understood flow charcteristic [22]. To understand the potential of AFC on more realistic flow scenarios, it is desireable to research similar, but more complex flow dynamics being more representative for high-dimensional real world flow problems. Noack et al. proposed a benchmark of a triangular arrangement of 3 individually rotated circular cylinders, the so-called fluidic pinball. This configuration allows to analyze a range of complex flow dynamics from steady state to chaotic vortex shedding which at the same time fulfills the requirement of maintaining a fairly simple simulation geometry since the majority of the computational effort in learning AFC laws is spent on the numerical simulation [19, 10]. In the following sections, the numerical simulation setup and the selection of 3 characteristic flow regimes, covered in this study, is presented finishing with a description of the modified boundary condition applying the flow control strategy to the flow solver.

3.1 Flow Problem Introduction

The simulation domain is described in a 2D cartesian coordinate system, where x^* and y^* are the non-dimensionalized coordinates with respect to the cylinders diameter D. The length of the domain is 22 and the height is 12. The 3 equally sized circular cylinders are placed on the vertices of a triangle with side lengths of $3D/2$ where the front, upper and lower cylinders are referred to as cylinder A, B and C, respectively. The origin of the coordinate system is placed in the middle of the back two cylinders considering the symmetry of the configuration.

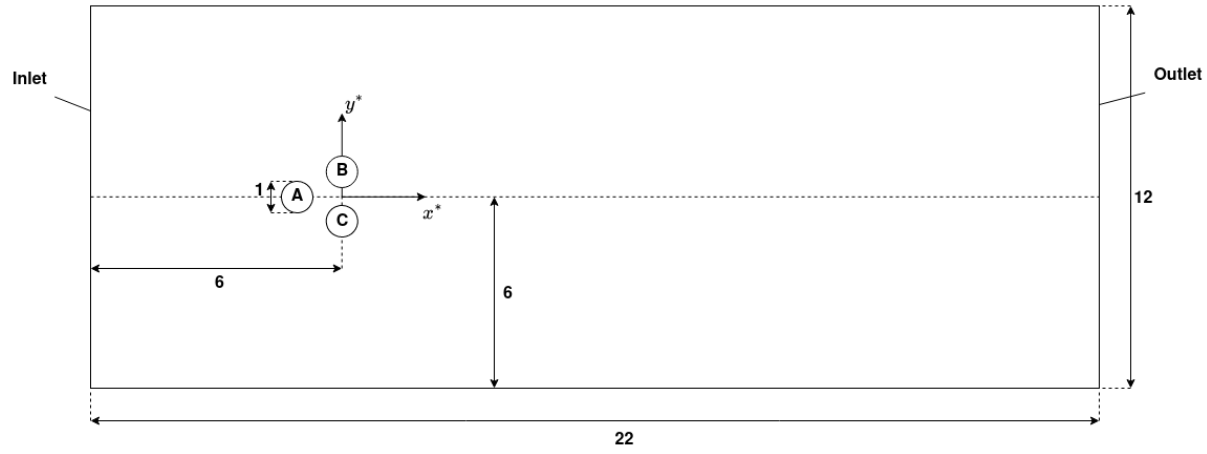

Figure 3.1: Simulation domain geometry

The coordinates of the cylinder centers are

$$\begin{aligned} x_A^* &= -3/2 \cdot \cos(30°) \approx -1.299, & y_A^* &= 0 \\ x_B^* &= 0, & y_B^* &= 0.75 \\ x_C^* &= 0, & y_C^* &= -0.75 \end{aligned}$$

and the cylinders remain static, only being allowed to rotate around the z-axis [19, 8, 10].

Flow properties

The flow consists of an incompressible Newtonian fluid with a constant density of $\rho = 1,0$ kg/m^3 and the Reynolds number

$$Re = \frac{u_\infty D}{\nu} \tag{3.1}$$

where $u_\infty = 1$ m/s is the mean inflow velocity and ν is the kinematic viscosity. Time t, velocity u, Nabla operator ∇ and pressure p are non-dimensionalized with equations 3.2 - 3.5

$$t^* = \frac{t}{D} \cdot u_\infty \tag{3.2}$$

$$\mathbf{u}^* = \frac{\mathbf{u}}{u_\infty} \tag{3.3}$$

$$\nabla^* = D\nabla \tag{3.4}$$

$$p^* = \frac{p \cdot D}{\nu \cdot \rho \cdot u_\infty} \tag{3.5}$$

with $\mathbf{u}^*$ representing the dimensionless form of the velocity vector $\mathbf{u} = (u, v)^T$. Thereby u is the velocity component in x-direction and v the velocity in the y-direction. The resulting non-dimensional properties are then used to formulate the dimensionless mass- and momentum conservation equations [21]

$$\nabla \cdot \mathbf{u}^* = 0 \tag{3.6}$$

$$\frac{\partial \mathbf{u}^*}{\partial t} + (\mathbf{u}^* \cdot \nabla^*)\, \mathbf{u}^* = -\nabla p^* + \frac{1}{Re} \nabla^{*2} \mathbf{u}^* \tag{3.7}$$

The flow induces a drag- and lift force on each cylinder caused by pressure forces F_P and shear stress forces F_S where the resulting force F can be decomposed into the drag force F_D acting in the x-direction and the lift force F_L acting in the y-direction.

The forces on each cylinder i are computed with an integral over the cylinders surface S

$$F_{D,i} = \int_S \left(\rho\nu \frac{\partial v_t}{\partial n} n_y - p n_x \right) dS \tag{3.8}$$

$$F_{L,i} = -\int_S \left(\rho\nu \frac{\partial v_t}{\partial n} n_x + p n_y \right) dS \tag{3.9}$$

where v_t denotes the tangential velocity on the surface and $\boldsymbol{n}$ is the unit normal vector on the surface with n_x being the component in the x-direction and n_y in the y-direction. With the drag- and lift forces the dimensionless drag- and lift coefficients c_D and c_L of each cylinder can be computed with equations 3.10 and 3.11 [25]

$$c_{D,i} = \frac{2F_{D,i}}{\rho U_\infty^2 A_{ref}} \tag{3.10}$$

$$c_{L,i} = \frac{2F_{L,i}}{\rho U_\infty^2 A_{ref}} \tag{3.11}$$

The reference area A_{ref} of each cylinder here is determined as the product of the reference diameter D_{ref} and the cell thickness Z_t in z-direction

$$A_{ref} = D_{ref} \cdot Z_t \tag{3.12}$$

where the reference diameter D_{ref} is D, $3/4D$ and $3/4D$ for cylinder A, B and C, respectively. Finally the total drag coefficient and total lift coefficient, in the following indicated as c_D and c_L, can be estimated each as the sum of the individual coefficients of all 3 cylinders.

Boundary Conditions

To solve the mass- and momentum conservation equations, also referred to as the Navier-Stokes equations, boundary conditions have to be applied at the borders of the computational domain. The left part of the domain makes the inlet, described by a constant inflow velocity

$$\boldsymbol{u}^*(0, y^*) = \boldsymbol{u}^*(x^*, 6) = \boldsymbol{u}^*(x^*, -6) = (1, 0)^T \tag{3.13}$$

where the same boundary condition applies at the top and bottom of the domain to provide frees-stream coniditons. Also the pressure gradient is zero at the inlet, the outlet and at the cylinder surfaces

$$\frac{dp^*(0, y^*)}{d\boldsymbol{x^*}} = \frac{dp^*(22, y^*)}{d\boldsymbol{x^*}} = \frac{dp^*(x^*_{cylinder,i}, y^*_{cylinder,i})}{d\boldsymbol{x^*}} = (0, 0)^T \tag{3.14}$$

with the position vector $\mathbf{x}^* = (x^*, y^*)^T$. A no slip boundary condition yielding

$$\mathbf{u}^*(x^*_{cylinder,i}, y^*_{cylinder,i}) = (0, 0)^T \tag{3.15}$$

is applied at each cylinder surface and the gradient of the dimensionless velocity is set to zero at the outlet

$$\frac{du^*}{d\boldsymbol{x^*}} = (0, 0)^T \tag{3.16}$$

As the scope of this work is to conduct 2D simulations, solving the flow in the z-direction is suppressed at the front and back of the simulation domain which is done in OpenFOAM by the „empty“ boundary condition.

Mesh Creation

The Mesh was created within the *blockMesh* utility of OpenFOAM which divides the domain into hexahedral blocks and fills each block with cells depending on the number of cells N_{x^*} and N_{y^*} specified in x^*- and y^*-direction, respectively. The resulting mesh from the mesh dependency study, which was carried out in a previous work [7], consists of 22 blocks filled with 52950 cells and is displayed in figure 3.2 where the inflow area and the wake is divided into 2 blocks each on the symmetry line, the upper and lower part into 3 blocks each and every cylinder is surrounded by 4 adjacent blocks. The refinement of the adjacent blocks increases towards the surface of each cylinder to accurately resolved the boundary layer. The highest refinement rate is used in the 2 adjacent blocks of the wake of the upper and lower cylinder since the highest gradients are expected there. The mesh convergence was validated at $Re = 200$.

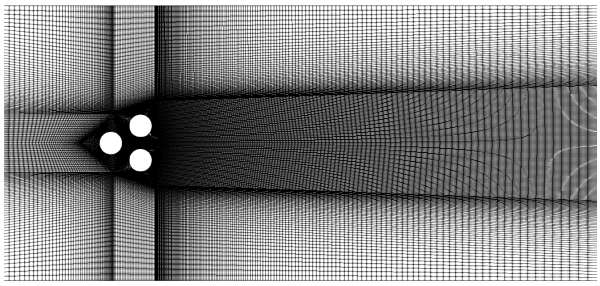

Figure 3.2: Discretized Domain

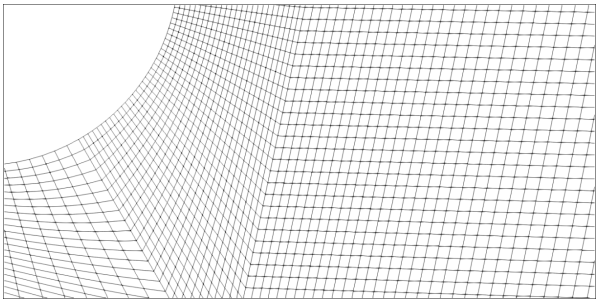

Figure 3.3: Mesh refinement in the wake

3.2 Regimes of Vortex Shedding

Upon increasing the Reynolds number, the Fluidic pinball passes different flow regimes changing the vortex shedding dynamics. Simulation results of the absolute mean of the total lift coefficient $|\overline{\mu_{c_L}}|$ and standard deviation σ_{c_L} in the interval of mesh convergence are shown in figure 3.4.

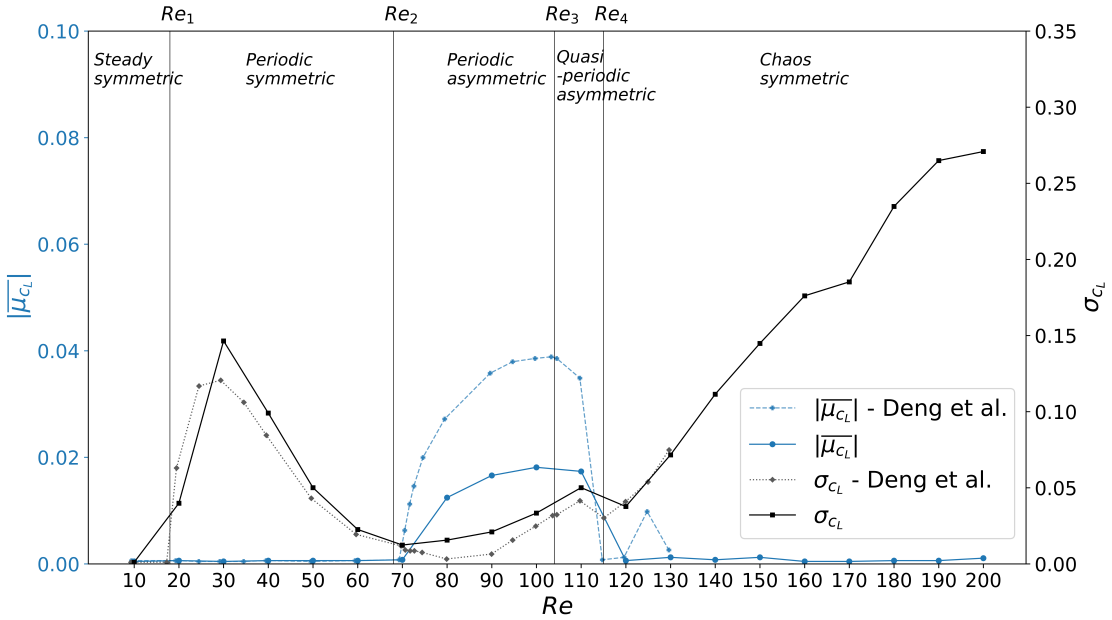

Figure 3.4: Flow Regime Analysis within a Reynolds number interval of 200. Simulation data was used from [7] and reference data originates from figure 4 in Deng et al. [8]

Until a first critical value of $Re_1 \simeq 18$ the flow remains stable with mean and standard deviation (STD) zero. Then C_L starts oscillating as STD is not zero anymore. Since $|\overline{\mu_{c_L}}| = 0$, it can be seen that the oscillation is symmetric around 0, forming a von Kármán vortex street in the wake of the top and bottom cylinder. At the second critial Reynolds number $Re_2 \simeq 68$, $|\overline{\mu_{c_L}}|$ starts rising which yields that the symmetry of the flow was broken. At this point in the gap between the upper and lower cylinder a jet arises which is slightly deflected to the bottom increasing the lift coefficient of the bottom cylinder and therefore being responsible for a state called asymmetric vortex shedding. This phenomena can be seen in figure 3.5b) where a contour plot of the magnitude of $\mathbf{u}$ is plotted. Reaching $Re_3 \simeq 104$ the jet between the two rear cylinders starts to oscillate in the deflected position until a last characterstic Reynolds number of $Re_4 \simeq 115$. At this point the flow is charcterized by random up- and downwards switches in the jet which is referred to as chaotic vortex shedding. The amplitude of the lift coefficient increases with higher Reynolds numbers but since the deflection of the jet is random, $|\overline{\mu_{c_L}}|$ is statistically 0 [10, 8, 9]. As different reference areas are used for coefficient estimation in the literature, $|\overline{\mu_{c_L}}|$ deviates from the reference data in the non-zero asymmetric and quasi-period asymmetric regime. Nontheless the data shows good agreement following all expected trends.

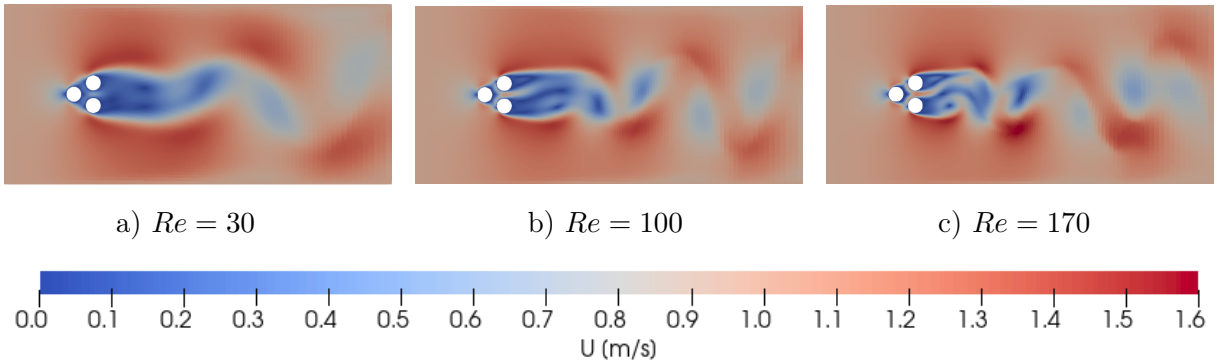

Figure 3.5: Velocity contour plot for different Reynolds numbers indicating a) symmetric vortex shedding, b) asymmetric vortex shedding and c) chaotic vortex shedding

Based on the previous analysis Reynolds numbers of 30, 100 and 170 have been chosen to be studied in this work to cover the 3 characteristic vortex shedding regimes.

3.3 Simulation Setup for Active Flow Control

To perform active flow control, the DRL agent needs to execute actions in the environment. The environment in this study is the CFD simulation and the action is the rotation of the 3 cylinders. To model the rotation of each cylinder in the CFD Simulation, a boundary condition is used to translate the rotation into a tangential velocity field that acts on the flow field. OpenFOAM offers a *rotatingWallVelocity* boundary condition which describes the rotation of an object in the flow. This boundary condition is modified towards sampling a rotational velocity $\boldsymbol{\omega}$ from the three action probability density functions (PDF) predicted by the agents policy network. The tangential velocity vector field $\boldsymbol{u}_{tan}$ is derived as

$$\boldsymbol{u}_{tan} = -\boldsymbol{\omega} \left(\boldsymbol{r}_f - \boldsymbol{r}_0 \right) \times \frac{\boldsymbol{\Omega}}{|\boldsymbol{\Omega}|} \tag{3.17}$$

where $\boldsymbol{\omega}$ is the sampled angular velocity, $\boldsymbol{r}_f$ is the vector of the cylinders surface cells centers, $\boldsymbol{r}_0$ is the vector of the corresponding cylinders center and $\boldsymbol{\Omega}$ is the vector specifying the axes of rotation. In this setup, each cylinder is rotated individually and each cylinder is allowed to rotate clockwise or counter-clockwise where counter-clockwise rotation corresponds to a positive

ω value as shown by the positive rotation angle ϕ in figure 3.6. In this example, ω_A and ω_B are rotating clockwise while ω_C is rotating counter-clockwise.

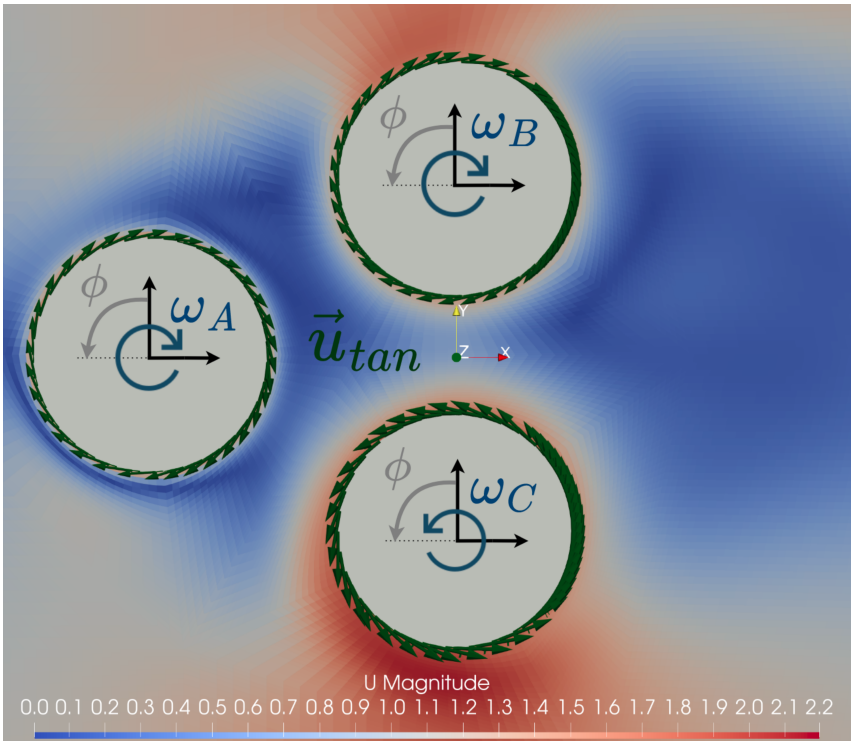

Figure 3.6: Modified Rotating Wall Velocity Boundary Condition

Updates of ω for each cylinder are processed with a linear transition function to avoid unphysical behaviour.

$$\Delta\omega = \frac{\Delta t\,(\omega_{t_c} - \omega_{t_c-1})}{n_{t,c}\,(t - t_{c-1})} \tag{3.18}$$

where ω_{t_c} is the new sampled rotational velocity, ω_{t_c-1} is the rotational velocity of the previous control updating time step, Δt is the simulation time step, $n_{t,c}$ is the number of time steps between a control update and t_{c-1} denotes the last time at which the angular velocity was computed. To prevent simulation instabilities due to very high tangential velocities, each rotational velocitiy is bounded by the action magnitude interval $\omega \in [\omega_{min}, \omega_{max}]$.

Chapter 4

Sensor Placement Methodologies

4.1 Unactuated Flow

In this chapter two methods are introduced to obtain optimal sensors for the DRL agent before learning the control law itself. The underlying goal is to investigate whether optimal sensors for the unactuated flow dynamics are still optimal for the actuated flow. Training a DRL agent is based on the requirement that the agent can observe states that are correlated with the parameters of the reward function. Therefore the spatial correlation between the flow field and the reward function parameters, in this case the lift- and drag coefficients of the 3 cylinders, is studied on the unactuated flow.
In a first approach, described in chapter 4.1.1, optimal sensors are placed corresponding to the spatial mode minima/maxima to reconstruct the flow field. This approach uses the assumption that a proper reconstruction of the whole flow field includes an accurate reconstruction of the pressure distribution at the cylinders surfaces which causes the lift- and drag forces on the cylinders. This assumption is evaluated by training the same random forest as in the second method with the optimal reconstruction sensors to predict the lift- and drag coefficients. In the second method, presented in chapter 4.1.2, a random forest regressor is trained to predict the lift- and drag coefficients from the flow state using data from CFD simulations. Afterwards two importance scores are computed generating a ranking of the impact of each flow input variable on the prediction accuracy.

Sensor Type Choice

Furthermore the type of the sensors needs to be chosen. In principle the flow state may be characterized by the pressure or the velocity. In most real engineering applications, e.g. cars or airplanes, sensors are restricted to the surface of the geometry and it is not possible to e.g. measure the flow in the wake [30]. Most realistic thereby might be pressure sensors where static pressure holes on the surface could be used. Since the goal of this study is not to conduct experiments afterwards, but to first of all investigate possible correlation in the whole flow field, sensors are also allowed to be placed in the wake. Nevertheless in preparation for future experiments, both methods can be easily adapted to only place sensors in position that may be measureable in reality. With the random forest method this can be done by only sampling sensors in a measureable area and the flow reconstruction method offers a way to introduce a cost factor to exclude undesired sensor positions.

Simulation Execution

Finally the unactuated simulations are carried out to obtain the pressure field and the coefficients used in the above mentioned approaches. The results are presented in figure 4.1 where the

aforementioned cases of vortex shedding were simulated for a dimensionless time of 1000. For visual interpretability a 1D gauss filter was applied to the lift coefficients of the chaotic vortex shedding, indicated by the index filter in figure 4.1c), to remove some noise induced by the chaotic fluctuations. The simulations start with a transient period in which the vortex shedding evolves. As soon as the vortex shedding is fully established, the coefficients and pressure values are splitted into training- and test time intervals for the random forest as shown by $t_{train,RF}$ and $t_{test,RF}$ in figure 4.1, respectively. For the pressure field reconstruction approach the simulation data is fully used starting at $t_{train,RF}$ where the von Kármán vortex street has fully formed. To compare the reconstruction results with the random forest approach, the prediction with the reconstruction sensors is also conducted on the $t_{test,RF}$ time interval. To validate the simulations, the resulting coefficients were compared with similar studies on the fluidic pinball where good agreement with the coefficients presented in Bieker et al. [3] was found.
Since the lift and drag forces on the cylinders result from the pressure distribution at the cylinders surfaces that leads to the vortex street formed in the wake, meaningful sensors have to be placed in the adjacent area of the cylinder surfaces or in the wake. This prior knowledge can be used to narrow down the investigated area of the mesh to a rectangular box, in the follwing called mask, spanned by $x^* = [-2.5, 10]$, $y^* = [-2, 2]$. Applying this mask reduces the total amount of mesh cells from 52950 to 27850, therefore substantially decreasing the computational effort to conduct the sensor placements.

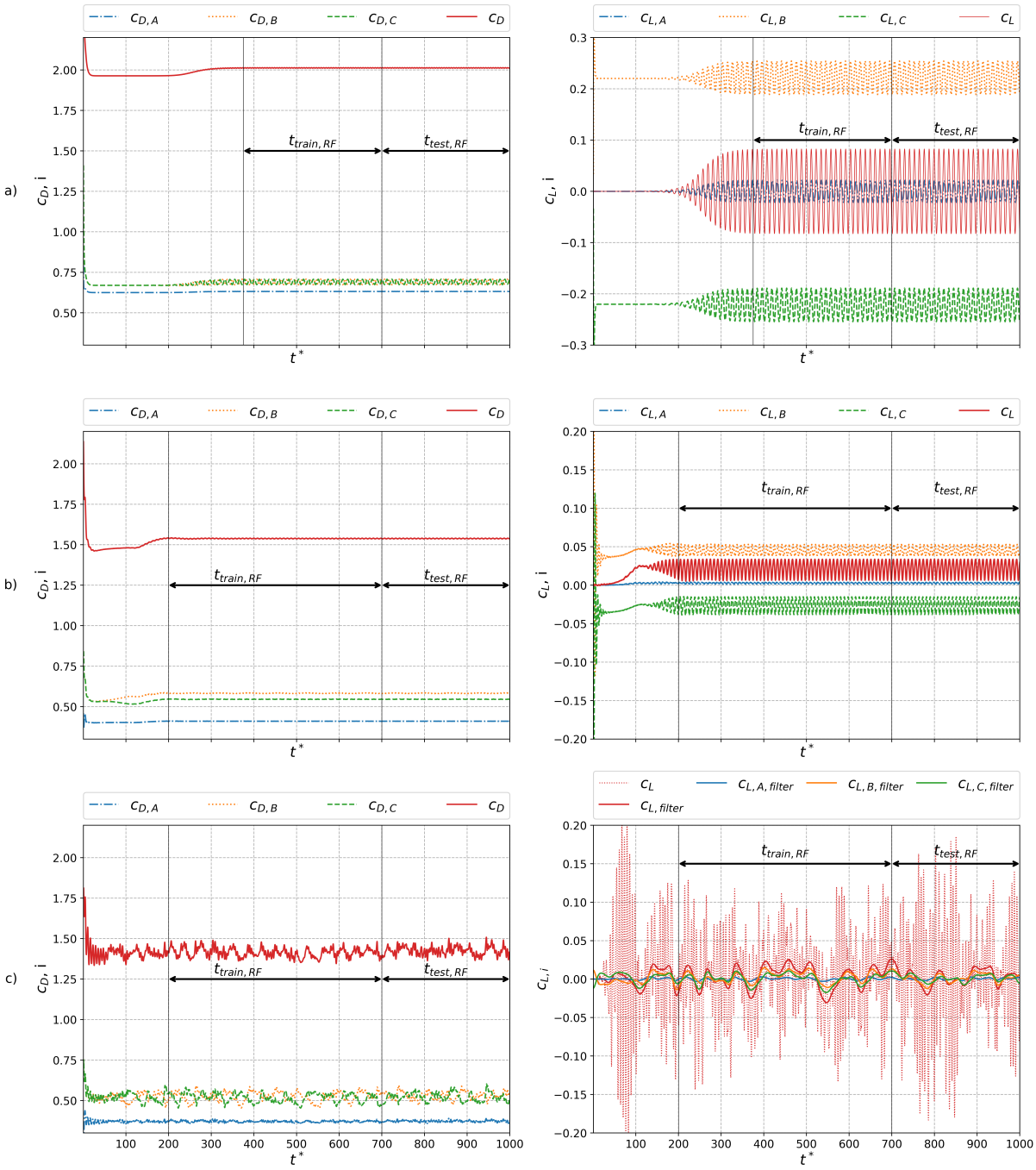

Figure 4.1: Coefficients for a dimensionless simulation time t^* of 1000 for a) Symmetric Vortex Shedding, b) Asymmetric Vortex Shedding and c) Chaotic Vortex Shedding

4.1.1 Sparse Sensor Placement via QR column pivoting

This sections presents the methodology published in the article Data-Driven Sparse Sensor Placement for Reconstruction [17]. It was implemented and made publicly available in the PySensors python package, which is used in this study [31].

The aforementioned approach determines pointwise measurements to reconstruct a high-dimensional signal from a lower-dimensional representation. Therefore training data is used to generate a low-rank POD basis, utilizing the SVD, which lays the foundation of this method. With this basis, which is specifially adjusted to the training data signal, optimal sensors for reconstruction can be found, as presented in the following.

The unknown, to be reconstructed signal consists of a time series of a measureable property with n spatial points which n this study is the fluctuating component of the pressure p'. It may be described as a rank r truncated linear combination of the POD modes where the truncation is chosen such that r modes describe the majority of the energy/variance associated with the pressure fluctuations. Equation 4.1 describes the coordinate form of the POD with j being the coordinates index.

$$p'_j = \sum_{k=1}^{r} \Psi_{jk} a_k \tag{4.1}$$

The main challenge of generating optimal sensors for reconstruction is to estbalish a measurement matrix $\mathbf{C} \in \mathbb{R}^{p \times n}$ which intends to reduce the initial n sensors to $p \ll n$. $\boldsymbol{C}$ is thereby structured as shown in equation 4.2 where $\boldsymbol{e_j}$ are basis vectors with unit entry at index j and zeros elswhere.

$$\mathbf{C} = \begin{bmatrix} \mathbf{e}_{\chi_1} & \mathbf{e}_{\chi_2} & \dots & \mathbf{e}_{\chi_p} \end{bmatrix}^T \tag{4.2}$$

By multiplying the measurement matrix C with $\boldsymbol{p'}$, a subset of p observations, denoted as $\mathbf{o}$, is created.

$$\mathbf{o} = \mathbf{C}\mathbf{p}' = \begin{bmatrix} p'_{\chi_1} & p'_{\chi_2} & \dots & p'_{\chi_p} \end{bmatrix}^T \tag{4.3}$$

Formally, $\chi = \{\chi_1, \dots, \chi_p\} \subset \{1, \dots, n\}$ which means that a set of p observations out of all n spatial points is derived. Upon inserting equation 4.1 into equation 4.3, the linear system depicted in equation 4.4 is received.

$$o_i = \sum_{j=1}^{n} C_{ij} p'_j = \sum_{j=1}^{n} C_{ij} \sum_{k=1}^{r} \Psi_{jk} a_k \tag{4.4}$$

C_{ij} is thereby the coordinate form of $\mathbf{C}$. To effectively compute the reconstruction $\hat{\mathbf{p}}'$ of the full state, the temporal mode coefficients $\hat{\mathbf{a}}$ have to be determined where equation 4.4 is solved for a and inserted into equation 4.1.

$$\hat{\mathbf{p}}' = \boldsymbol{\Psi}_r \hat{\mathbf{a}}, \text{ where } \hat{\mathbf{a}} = \boldsymbol{\Theta}^{-1}\mathbf{o} = \left(\mathbf{C}\Psi_r\right)^{-1}\mathbf{o} \tag{4.5}$$

Equation 4.5 therefore shows that the inverse of the matrix $\boldsymbol{\Theta}$ has to be derived. Ultimately the sensor placement seeks for rows of $\boldsymbol{\Psi_r}$ corresponding to sensors in the high-dimensional state space, that optimally condition the inversion of the matrix $\boldsymbol{\Theta}$. Optimal conditioning of a matrix is expressed with the condition number which measures the sensitivity of matrix multiplication

or inversion to errors in the input. It is computed as the ratio of the maximum and minimum singular values of the matrix where higher values correspond to worse performance and a value of 1 is the optimum. Therefore the condition number of the matrix $\boldsymbol{\Theta}$ has to be bounded by explicitly choosing the rows of the measurement matrix $\boldsymbol{C}$ such that the product of the p singular values of $\boldsymbol{\Theta}$ is maximized.

$$\chi_\star = \underset{\chi,|\chi|=p}{\operatorname{argmax}} \prod_i \lambda_i \left(\boldsymbol{\Theta}\right) \tag{4.6}$$

A direct computation of the optimization problem above is computationally infeasible which is why an approximation method utilizing a matrix QR factorization with column pivoting of $\boldsymbol{\Psi}_r^T$ is used. Hereby the QR factorization decomposes the matrix $\boldsymbol{\Psi_r}$ into a unitary matrix $\boldsymbol{Q}$, an upper triangular matrix $\boldsymbol{R}$ and a column permutation matrix $\boldsymbol{C}$.

$$\boldsymbol{\Psi}_r^T \mathbf{C}^T = \mathbf{QR} \tag{4.7}$$

The pivoting procedure applied to the measurement matrix provides an approximation to the optimization problem in 4.6 and therefore finally yields χ and thus the p optimal sensors.

Using the training data produced by the priorly executed CFD simulations, the above described approach is carried out for all 3 vortex shedding regimes to generate optimal reconstruction sensors tailored to the individual flow dynamics. The resulting sensor placements are presented in chapter 4.1.3 and are compared with the sensors chosen by the Random Forest Importance scores, which is outlined in the next section.

4.1.2 Random Forest Importance Scores

In this chapter the methodology used to compute importance scores for the input variables of a Random Forest Regressor is presented. The scores quantify the impact of the input variables on the prediction of the random forrest, therefore providing a ranking to determine and keep the most important ones. Within the framework of random forests, two different variable importance scores can be computed based on the architecture of its decision trees [4].

Mean Decrease in Impurity

The Mean Decrease in Impurity (MDI) characterizes the importance of input variable X_j based on all splits in the forests trees using X_j as the splitting variable. The term impurity is used in the literature as the generalization of the metric to evaluate the splitting of a node where in this study the mean squared error (MSE) is used. MDI searches for all trees in the forest where X_j is used as a splitting variable and then averages the decrease in impurity for all found splits. The decrease in impurity Δi on a split s dividing node t into a left node t_L and a right node t_R is

$$\Delta i(s,t) = i(t) - p_L i\left(t_L\right) - p_R i\left(t_R\right) \tag{4.8}$$

where $i(t)$ is the impurity before the split and p_L and p_R are the fractions (probabilities) of all tree building observations reaching the left and right node, respectively. The final impurity of input variable X_j is then estimated as the product of the probability of reaching node t times the impurity decrease resulting from split s_t, averaged over all φ_m trees (for $m = 1, ..., M$) using X_j as a split.

$$\mathrm{I}_{\mathrm{MDI}}\left(X_j\right) = \frac{1}{M}\sum_{m=1}^{M}\sum_{t\in\varphi_m} 1\left(j_t = j\right)\left[p(t)\Delta i\left(s_t, t\right)\right] \tag{4.9}$$

By weighting the impurity decrease with the probability of samples reaching node t, splits located at the top of the tree are considered more important [16, 29].

Permutation Importance

Contrary the Permutation Importance (PI) of input variable X_j is characterized by the overall prediction accuracy of the random forest. Therefore in the literature it is also called Mean Decrease in Accuracy (MDA). Within this importance ranking score the relationship between input variable X_j and the response value is broken by randomly permuting the N observations of X_j. Since time series data is used in this study this refers to randomly shuffling the temporal evolution of the pressure value X_j. After shuffling the observations, the random forest is retrained with the corrupted dataset and the PI score of X_j is computed as the difference in the predictive accuracy resulting from the permutation

$$\mathrm{I}_{\mathrm{P}}\left(X_j\right) = \left(d_i - \hat{d}_i\right)^2 - \frac{1}{n_{perm}}\sum_{k=1}^{n_{perm}}\left(d_i - \tilde{d}_{i,k,j}\right)^2 \tag{4.10}$$

where $\hat{d}_i$ is the predictive accuracy without permutation and $\tilde{d}_{i,k,j}$ is the predictive accuracy of the k-th permutation iteration where the observations of input variable X_j were randomly shuffled. Shuffling is repeated n_{perm} times to decrease the variance in the prediction inherited by the random permutation of the observations. However this requires n_{perm} times shuffling, retraining and repredicting the response values for every input value X_j. Therefore the computational effort also scales with n_{perm}. Consequently the PI is much more costly to evaluate than the MDI since MDI comes as a byproduct of the training process. An advantage of the PI is that it can not only be computed on the training dataset, but also on the test dataset. This prevents overestimating the importance of input values in the training dataset due to possible overfitting. Overfitting means that the model might erroneously focus on certain input values not representing the true underlying relationship between input- and response value to increase the predictive accuracy on the training data [16].

Sampling Approach

Computing the MDI and PI as described above on the masked mesh region containing 27850 input pressure values might result into non meaningful and biased importances. To tackle the problem of the high amount of input values, a Latin Hypercube Sampling (LHS) approach combined with a clustering algorithm was utilized to conduct sensor placements on a reduced input variable space.

Since the mesh yields a very high spatial resolution of the input variables, two problems at computing the importance scores occur. The first problem is that due to the spatial correlation in the input variables, certain input values might be given a low importance because adjacent, nearly identical variables deliver the same information. This form of competition is contraproductive and should be adressed to larger areas within the mesh to find the regions where sensors are optimally placed. The second problem is that permuting the observations of only one input variable out of 27850 will likely have little to no effect on the overall prediction.
Therefore it can be concluded that the dimensionality of the input variable space needs to be significantly reduced to compute valid optimal sensors. At the same time the input variable space should still reflect a homogeneous distribution of the whole mask. This requirement is

achieved by using LHS to create a new reduced, but homogeneous input variable space with the dimension of $n_{samples}$. To compensate slight differences in the sampling it is repeated n_{repeat} times yielding a fair chance to cover the majority of all possible sensor locations. During each sampling iteration only the n_{keep} most important input variables are kept and saved. Combining the kept sensor positions results in pointclouds forming at the optimal sensor locations due to the differences in the sampling iterations. The pointclouds can be substituted by a sensor each with a clustering algorithm. Clustering is a machine learning technique used to group n datapoints into a predefined number of k clusters. So called cluster centroids μ_j are initialized and updated intending to minimize a distance criteria called inertia. The inertia computes the sum of the squared distance of each datapoint to its cluster centroid, as depicted in equation 4.11 [2].

$$\sum_{i=0}^{n} \min_{\mu_j \in C} \left(\|x_i - \mu_j\|^2 \right) \tag{4.11}$$

Eventually the converged centroids are considered as the optimal sensors. Evaluating the MSE of the RF retrained with the optimal sensors concludes this method. In this study the K-means ++ clustering algorithm is used [2]. The sampling approach is summarized in figure 4.2.

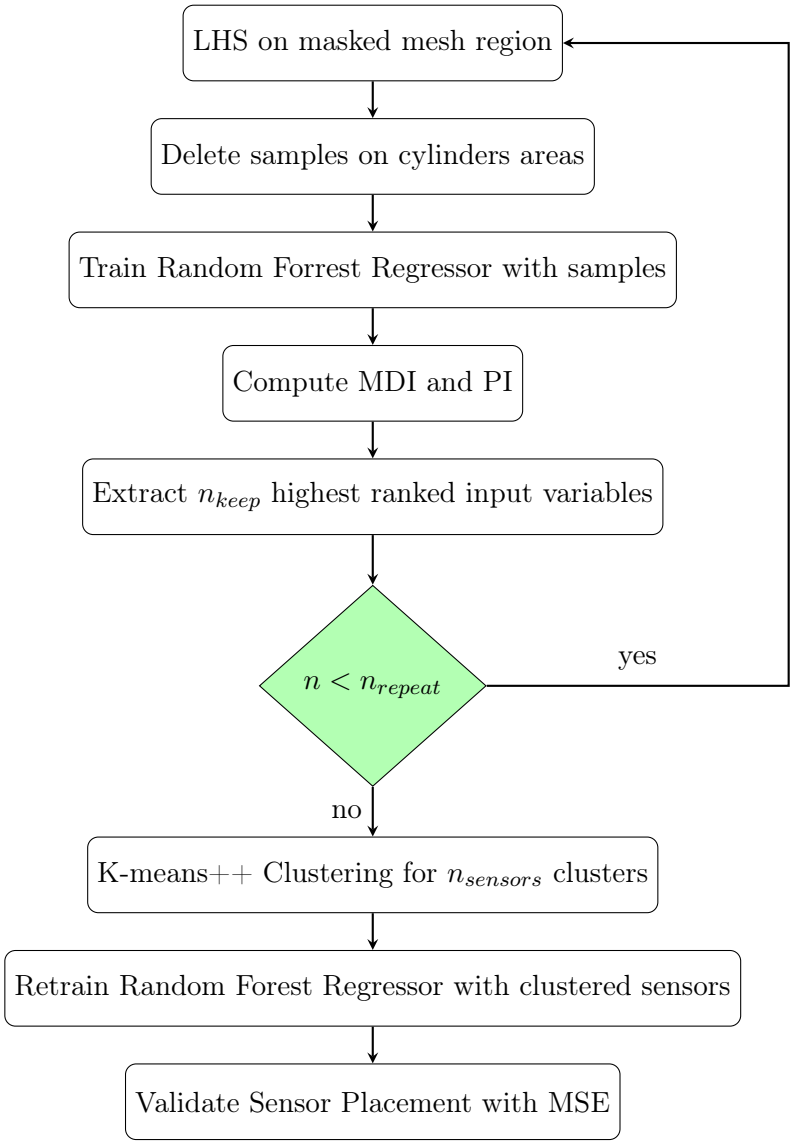

Figure 4.2: Random forest sensor placement methodology

Hyperparameter Discussion

This sections presents a brief introduction of the hyperparameters used in the previously described methodology. To compute representative importance scores it needs to be proven that the random forest yields valid predictions without overfitting the data. Goal of the hyperparameter search therefore is to find the least number of splits necessary to provide a sufficient prediction of the coefficients. In the second part of this discussion the influence of the sampling parameters on the sensor placement is investigated.

Random Forest Hyperparameters

The random forest is implemented using the Random Forest Regressor class from the ML library scikit-learn [1]. The main hyperparameters to control the splitting process are the tree depth (maximum number of splits), the minimum number of samples per split and the minimum number of samples to end in a leaf node. It was found that as long as the depth of the trees is set to at least 3, the random forest is capable to provide a good prediction of the coefficients. Furthermore to give every input variable the chance to be chosen as an optimal sensor within the MDI, in every split all input variables may be considered as the splitting variable.

Sampling Hyperparameters

The sampling approach is based on 3 parameters, namely the number of sampling iterations n_{repeat}, the total number of input variables to sample in each iteration $n_{samples}$ and the fraction of the most important input variables to keep in each iteration n_{keep}. Figure 4.3 exemplary shows the influence of $n_{samples}$ and n_{keep} on the pointclouds obtained from the MDI in the chaotic vortex shedding case.

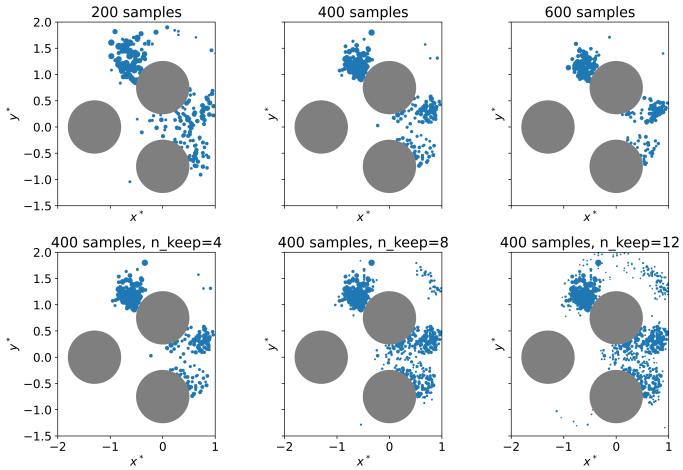

Figure 4.3: Influence of $n_{samples}$ and n_{keep} on the MDI pointclouds in the chaotic vortex shedding case

From the top row it can be seen that upon increasing $n_{samples}$ more clear pointclouds are received where starting at $n_{samples} = 400$ clearly separable clusters are achieved due to the more dense spatial resolution of the parameter space. A further increase almost doesn't change the pointclouds such that $n_{samples} = 400$ was chosen. In the bottom row the impact of n_{keep} at $n_{samples} = 400$ is displayed. At $n_{keep} = 4$ the cluster pointclouds consist of equally important ranked sensors and with a higher proportion of n_{keep} clearly less important sensors are added making the clusters less detachable thus only moving away the centroids from the most important regions. As a result $n_{keep} = 4$ was selected. The third and last parameter, the number of sampling iterations, controls the density of the pointclouds where $n_{repeat} = 50$ was found as

a good compromise between sufficiently developed pointclouds and computational effort. Table 4.1 summarizes the chosen hyperparameters

Parameter	n_{trees}	max. depth	max. features	max. samples	min. samples leaf	min. samples split	n_{repeat}	$n_{samples}$	n_{keep}	n_{perm}
Value	100	3	$n_{features}$	0.6	4	25	50	400	4	10

Table 4.1: Methdology hyperparameter

where the additional parameters max. samples denote the amount of bootstrap samples and max. features corresponds to the amount of input variables considered at each split.

4.1.3 Sensor Placements

In this chapter the resulting sensor placements obtained by the random forest importance scores and the reconstruction of the flow field are presented and compared for each vortex shedding regime. As choosing a rank r truncation of the modes is a key part for the QR pivoting method, the singular value spectrum is plotted for an increasing number of modes to analyze the energy associated with each mode. Subsequently the reconstruction error is plotted over an increasing number of sensors, the optimal number of sensors is chosen and the resulting sensors are visualized in contour plots of the dominant modes. In comparison the pointclouds of the MDI and PI as well as the optimal sensors are displayed and a correlation study is conducted. Finally the sensor placements are evaluated with the MSE of a Random Forest predicting the coefficients.

Symmetric Vortex Shedding

Figure 4.4 shows the logrithmic decay of the normalized singular values as well as the cumulative sum of the singular values in an interval of the first 20 modes. As the singular values are decaying very quickly the majority of the energy is stored in the first few modes. In this case the first mode is by far the dominant one as it already accounts for nearly 90% of the total energy. Therefore the symmetric vortex shedding is an appropriate case to establish a low rank POD basis for optimally reconstructing the pressure field.

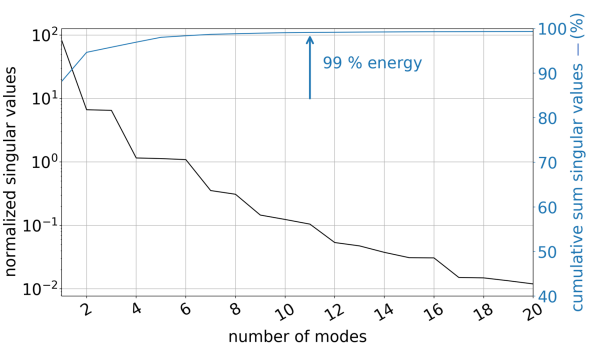

Figure 4.4: Modal singular value spectrum for Re=30

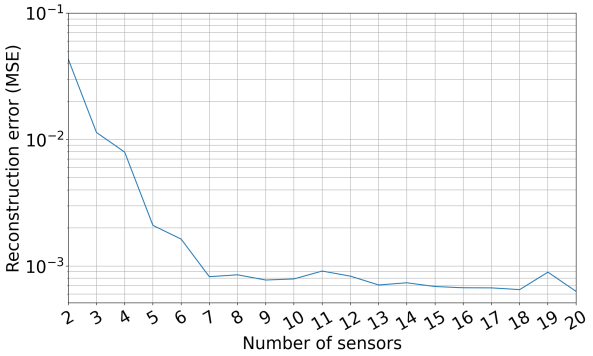

Figure 4.5: Reconstruction error (MSE) for Re=30 with QR pivot sensors

In figure 4.5 the reconstruction error is plotted over an interval of 20 sensors. It can be seen that the error saturates at $n_{sensors} = 7$ such that this marks the optimal number of sensors. This saturation is explained by the fact that there is hardly any increase in the cumulative sum of the singular values at $n_{modes} > 7$.

The 7 resulting sensors, ordered by their importance with 1 being the most important, are shown in contour plots of the first 4 modes in figure 4.6.

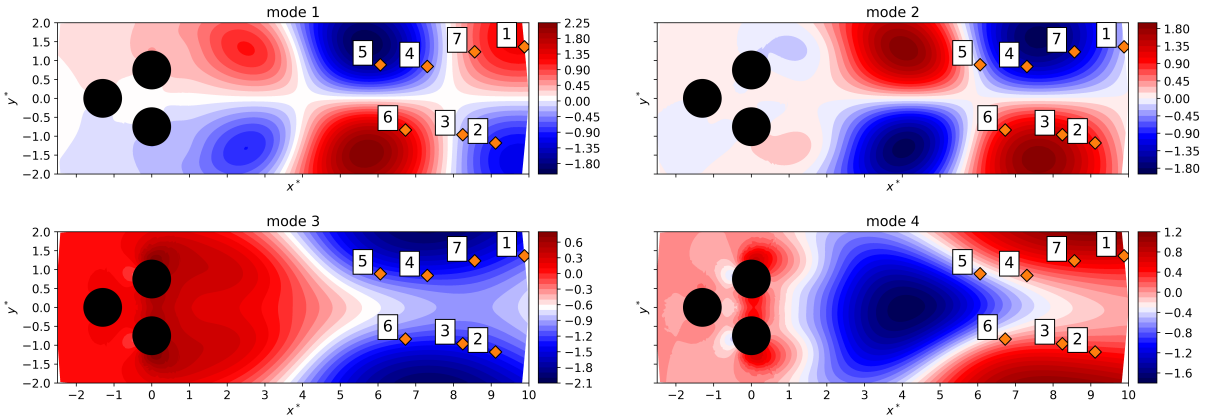

Figure 4.6: Sensor placement with QR pivoting for symmetric vortex shedding

The first and dominant mode reflects the oscillatory dynamics of the vortices propagated up and down in the wake as can be seen from the opposite signs of the modes according to the symmetry line. The sensor placement also reflects this symmetry and is mainly aligned to the minima/maxima of the dominant first mode.

Comparatively in figure 4.7 the pointclouds and optimal sensors for the MDI and PI are depicted in an instantaneous snapshot of the vorticity.

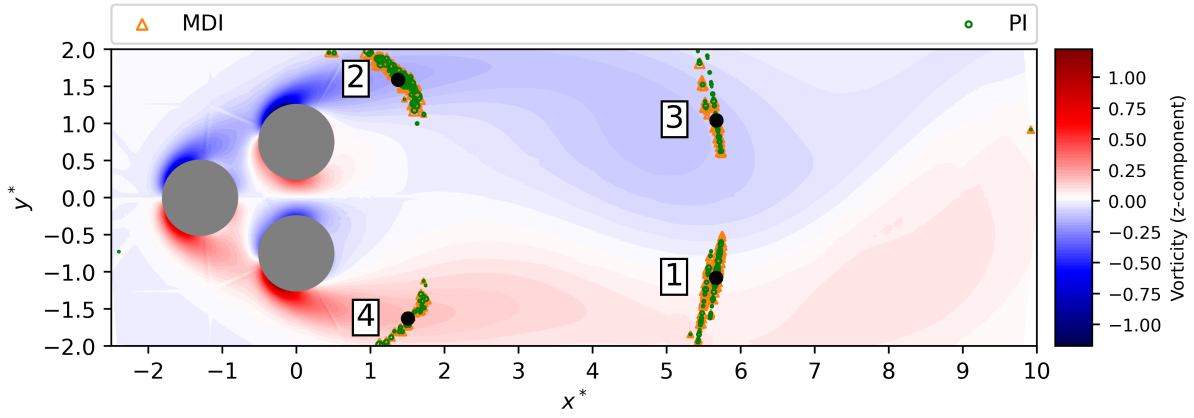

Figure 4.7: Pointcloud importances for symmetric vortex shedding

First of all 4 pointclouds occur from the MDI and PI which are nearly identical with the MDI building slightly denser clusters. Therefore the clustering was carried out using the MDI pointclouds. The sensor placement as well reflects the symmetry of the flow and interestingly very well matches the minima/maxima of the first dominant POD mode.

The choice of the optimal sensors is attributable to the splitting process of the random forests decision trees. In the splitting process the input variables with the highest correlation to the target values are favored. Since the observations at a split are propagated into a left and right node according to whether the splitting variable H_j is greater or less than the splitting value s, also the standard deviation is a relevant factor. In other words: if an input variable is nearly constant and uncorrelated to the target variables, it doesn't contain any valueable information for the prediction. These statements can be validated with figure 4.8 which displays a contourplot of the normalized standard deviation of the pressure σ_p and a linear correlation map between the total lift- and drag coefficient and the pressure probes.

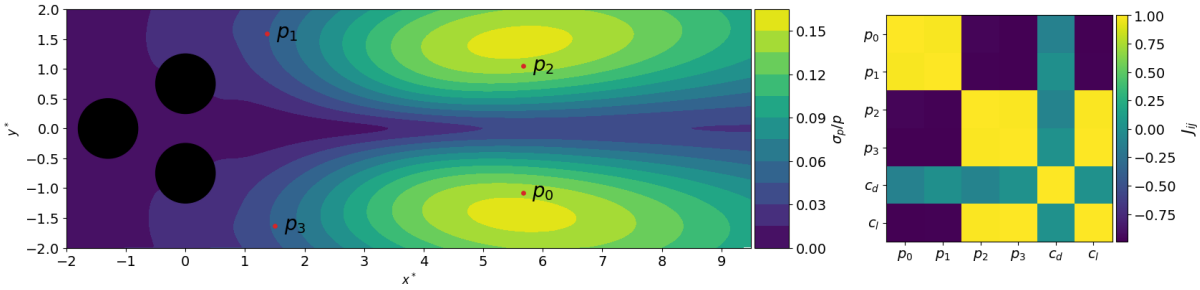

Figure 4.8: Standard deviation of the pressure field (left) and linear correlation matrix (right) for symmetric vortex shedding

The correlation map shows that the 4 probes are nearly perfectly correlated with the lift coefficient and the latter two probes are placed near the maximum of the standard deviation. The correlation with the drag coefficient is small where the probes denoted as p_1 and p_3 are nearly completely uncorrelated. This is the case because they are placed out of the recirculation area of the upper and lower cylinder accounting for the pressure drop resulting in the overall drag force.

To validate the sensor placements the MSE of the RF prediction in an interval of 10 sensors is shown in the upper row of figure 4.9.

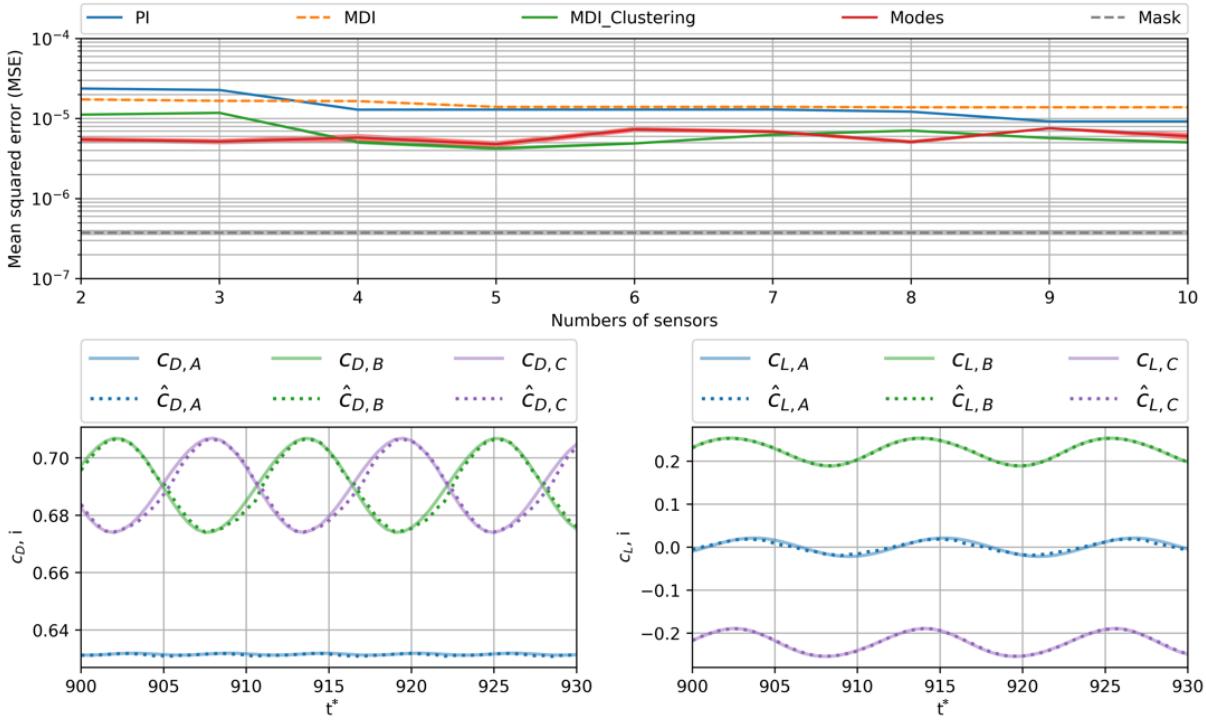

Figure 4.9: Accuracy of a random forest trained for 10 sensor placements on each method on Re=30

It can be seen that the clustered sensors and the reconstruction sensors, denoted as MDI_Clustering and Modes, respectively, yield reasonable predictions with an MSE in the order of magnitude of 10^{-5}. Using less than 4 clusters places the sensors inbetween the pointclouds which is not desireable. However since the the minima/maxima of the first dominant mode extend over large areas in the wake even with less than 4 clustered sensors reasonable amount of correlation is achieved leading to good predictions as well. Generally this shows that multiple sensor placements exist achieving sensible predictions. Using more than 2 reconstruction sensors nearly has no effect on

the prediction which indicates that 2 sensors already offer enough correlation to learn a valid prediction. The same holds if the highest ranked sensors of the PI and MDI are used indicated by the blue and orange-dashed curves. Predicting the coefficients with all input variables in the mask, denoted by the grey dashed curve in figure 4.9, states a prediction with roughly one order of magnitude less MSE. Nevertheless, as demonstrated in the bottom row of figure 4.9 for a dimensionless time interval of $t^* = [900, 930]$, the predictions are able to fully represent the periodic coefficients. The slight deviation in the drag coefficient might be explained by the fairly low correlation with the drag coefficient. Since p_0, p_1 and p_2, p_3 are nearly identically correlated, a further reduction to one sensor of each group could be made.

Asymmetric Vortex Shedding

In terms of the singular value spectrum the case of the asymmetric vortex shedding is very similar to the symmetric vortex shedding. Figure 4.11 shows a similar decay of the normalized singular values over an increasing amount of modes.

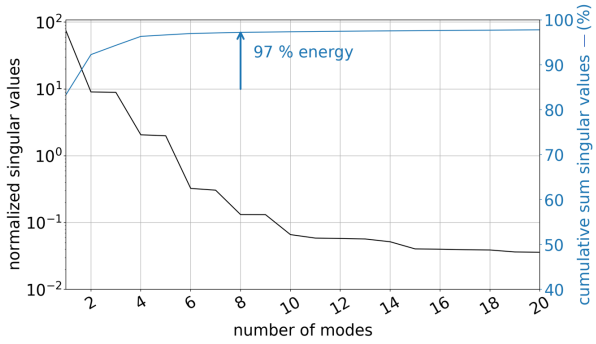

Figure 4.10: Modal singular value spectrum for Re=100

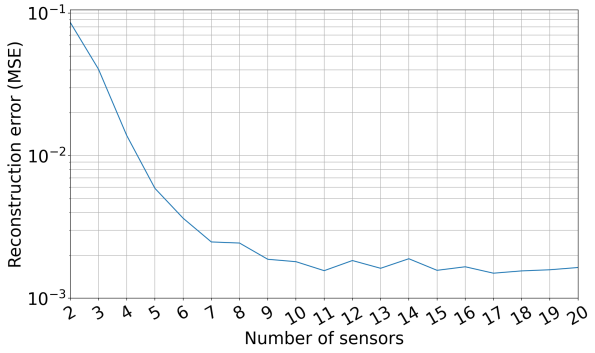

Figure 4.11: Reconstruction error (MSE) for Re=100 with QR pivot sensors

Here the energy associated with the first few modes saturates at 97% at $n_{modes} = 8$. Therefore also in the asymmetric case the flow is representable by a low rank POD basis to conduct sensor placements for optimally reconstructing the pressure field. The reconstruction error plotted in 4.11 decreases mainly until $n_{sensors} = 7$ which therefore highlights the optimal number of sensors.

In figure 4.12 the 7 optimal sensors are visualized in contour plots of the first 4 modes. The first and second mode are spatially pretty similar distributed to the modes in the symmetric vortex shedding case. Since the vortices are more propagated to the bottom, therefore breaking the symmetry of the flow, the dominant modes are located behind the lower cylinder in the wake. Likewise the symmetry of the sensor placement obtained in the symmetric case is also broken such that the optimal sensors are placed nearly in a line in the bottom half of the mask.

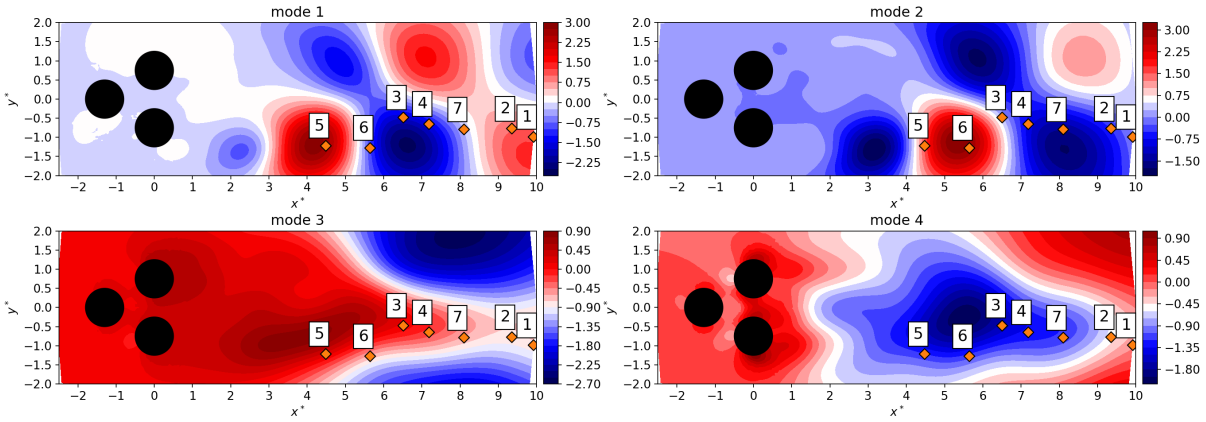

Figure 4.12: Sensor placement with QR pivoting for asymmetric vortex shedding

Contrary the pointclouds of the MDI and PI still show a near symmetric arrangement, as depicted in figure 4.13.

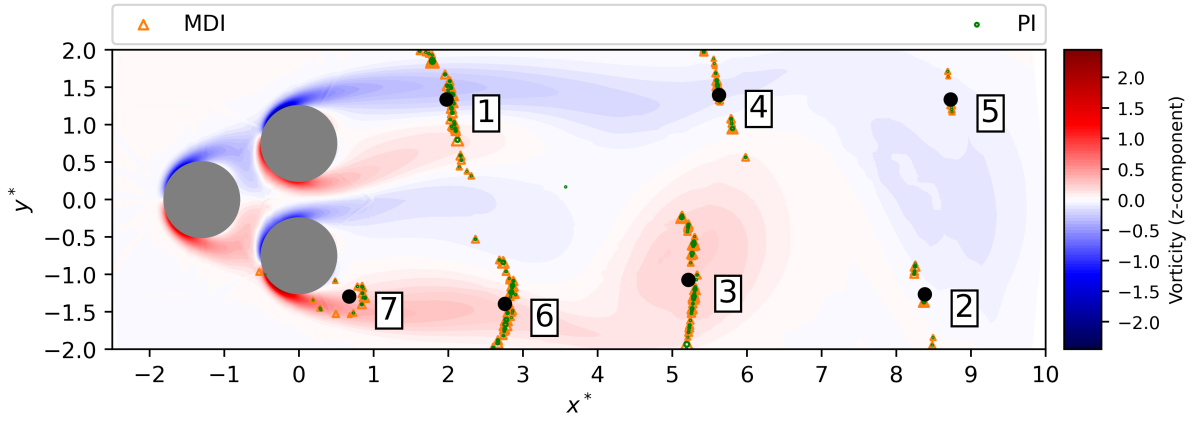

Figure 4.13: Pointcloud importances for asymmetric vortex shedding

As in the symmetric vortex shedding case, the clustering was carried out on the MDI pointclouds, resulting in 7 optimal sensors. Comparing the locations of the sensors to the first POD mode, the MDI sensors are still aligned to the spatial distribution of the minima/maxima but fit less well to them compared to the optimal sensors of the MDI in the symmetric case.

Figure 4.14 analyzes the sensor placement in terms of the standard deviation of the pressure field and the correlation of the sensors with the lift- and drag coefficients. All sensors except the first two, denoted as p_0 and p_6 are placed in the wake in an area of an increased standard deviation. Opposingly to the symmetric case all sensors are highly correlated with not only the lift- but also the drag coefficients where especially the drag coefficient shows near perfect correlation. The sensors p_0 and p_6 are placed in an area with a very low standard deviation but are at the same time nearly identically correlated as the other sensors. From this it can be derived that the driving criteria for chosing the most important sensors is the correlation to the target values and the standard deviation of the input variables plays only a subordinate role.

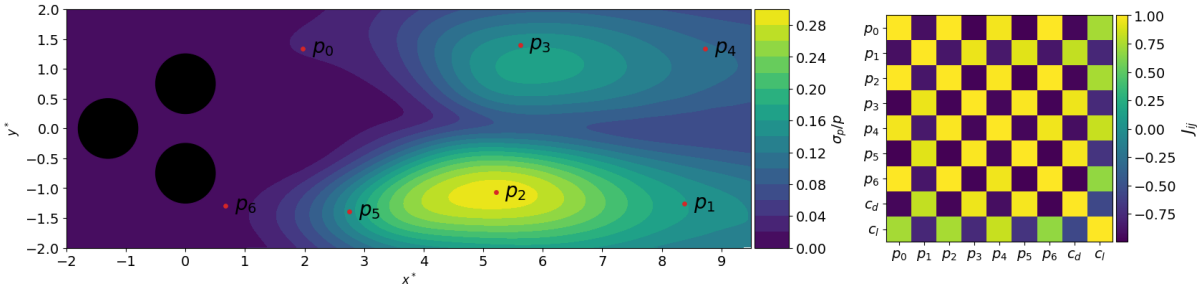

Figure 4.14: Standard deviation of the pressure field (left) and linear correlation matrix (right) for asymmetric vortex shedding

Due to not only the high correlation to the lift coefficient, but also to the drag coefficient in this case, the predictions are even better than in the symmetric case where for the clustered sensors a MSE in the order of 10^{-7} and for the modes of 10^{-6} is achieved. Therefore it can be concluded that useful sensor placements for the asymmetric vortex shedding have been obtained. Since the groups of p_0, p_2, p_4, p_6 and p_1, p_3, p_5 are nearly identically correlated, it may be feasible to further reduce the number of sensors to 2 by only keeping one of each group.

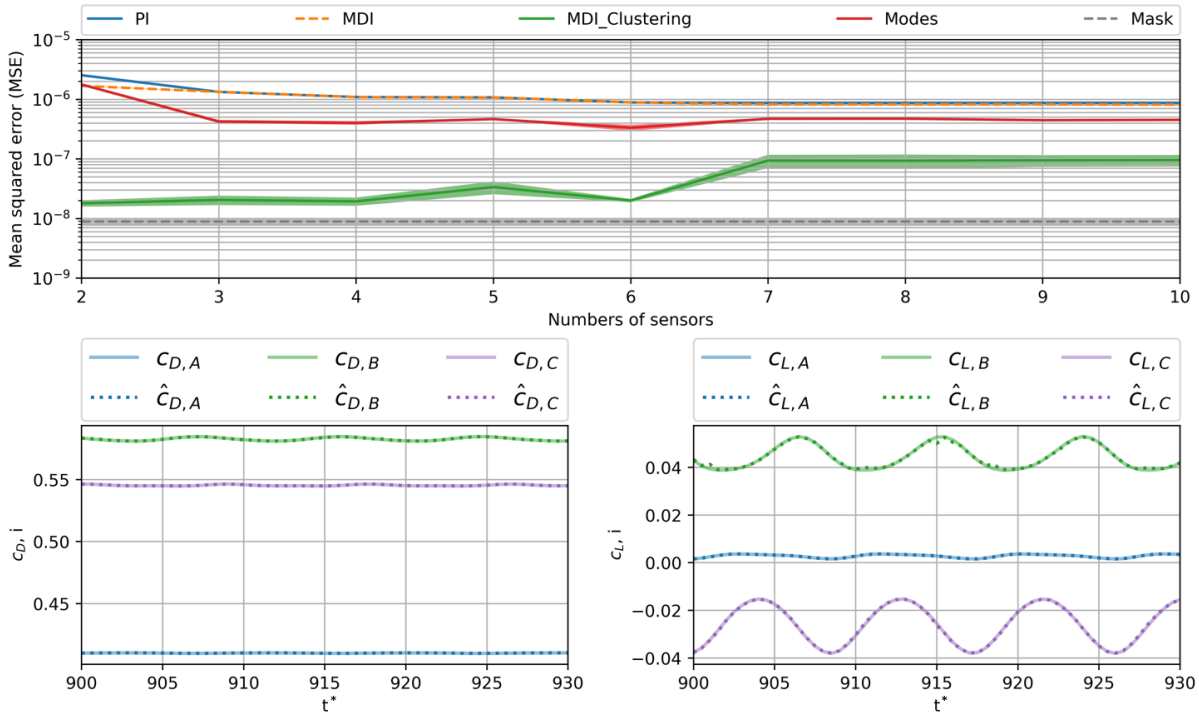

Figure 4.15: Accuracy of the random forest trained for 10 sensor placements on each method on Re=100

Chaotic Vortex Shedding

The increasing complexity of the vortex shedding dynamics due to the chaotic fluctuation of the jet between the top and bottom cylinder leads to extended scales of coherent structures in the wake. Therefore the singuar value spectrum is substantially changed, as can be seen from figure 4.16.

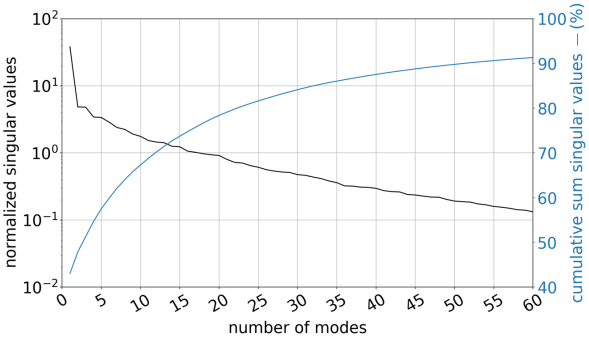

Figure 4.16: Modal singular value spectrum for Re=170

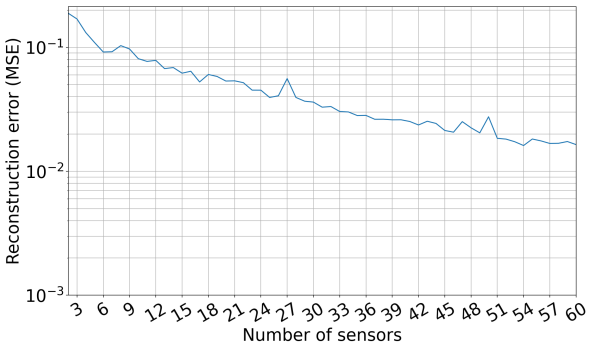

Figure 4.17: Reconstruction error (MSE) for Re=170 with QR pivot sensors

The logarithmic decay of the singular values for an increasing number of modes is much slower as observed in the cases before. Equivalently the energy of the flow is distributed over a much higher number of modes where 60 modes are required to cover at least 90% of the total energy. The diversification of the coherent structures leads to a higher dimensional problem which can not be transfered to a low rank POD basis. As a result also the reconstruction error displayed in figure 4.17 decreases only slightly over an increasing number of modes. Since the sensor placement methodology requires to place one sensor for each considered mode, an infeasible amount of sensors would have to be chosen. Therefore a sensor placement using QR pivoting dismissed over the chaotic vortex shedding.

Despite the high dimensionality of the flow dynamics, the random forest is capeable to conduct an optimal sensor placement containing 3 dense pointcloud regions.

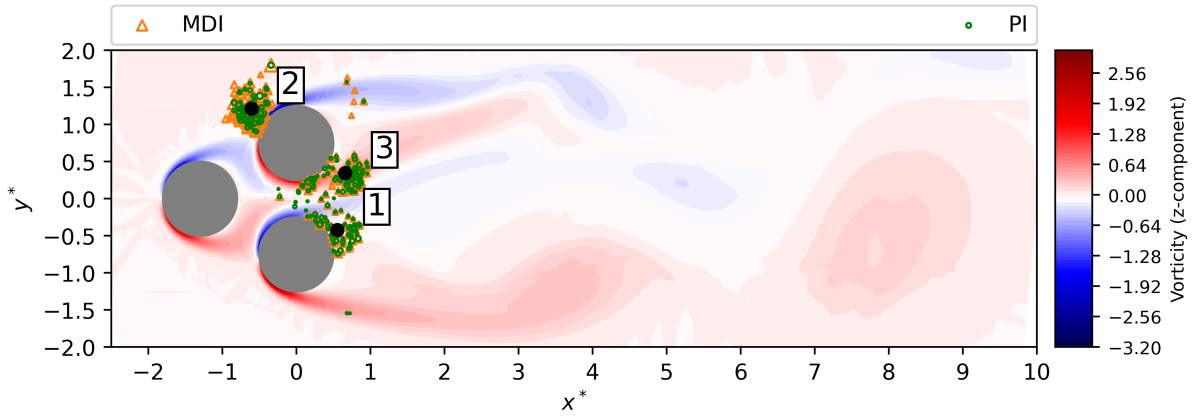

Figure 4.18: Pointcloud importances for chaotic vortex shedding

Figure 4.18 displays the 3 optimal sensors from the clustered MDI pointclouds. Sensor 1 and 3 are placed according to the random up- and downwards fluctuations of the jet arising between the upper and lower cylinder. Intuitively this is a clever sensor placement for active flow control since thereby the upper and lower cylinder can be rotated according to the instantaneous deflection of the jet. With this the vortex propagation and the flow separation behind the upper and lower cylinder could be controlled. Although the vast amount of the standard deviation of the pressure field is located downstream in the wake the sensors are clearly placed outside of this region. This can be explained by the fact that the vortices are chaotically propagated in the wake whereby no clear correlation to the lift- and drag coefficients can be found. However the linear correlation map shown in figure 4.19 indicates that clear correlation can be found at the locations where the sensors are placed. Sensor p_1 is highly negative correlated with the lift coefficient whereas p_0 and p_2 have a clear correlation with the drag coefficient.

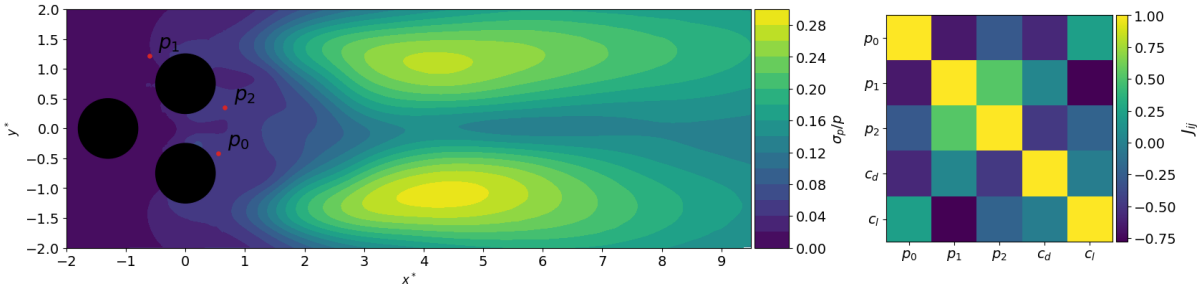

Figure 4.19: Standard deviation of the pressure field (left) and linear correlation matrix (right) for chaotic vortex shedding

Since the lift- and drag coefficients are highly non-linear in the chaotic vortex shedding case, evaluating the sensor placement with the prediction of a random forest seems reasonable. From the top row of figure 4.20 it can be seen that the the sensors obtained by the MDI clustering are providing near identical predictions compared with all input variables of the mask.

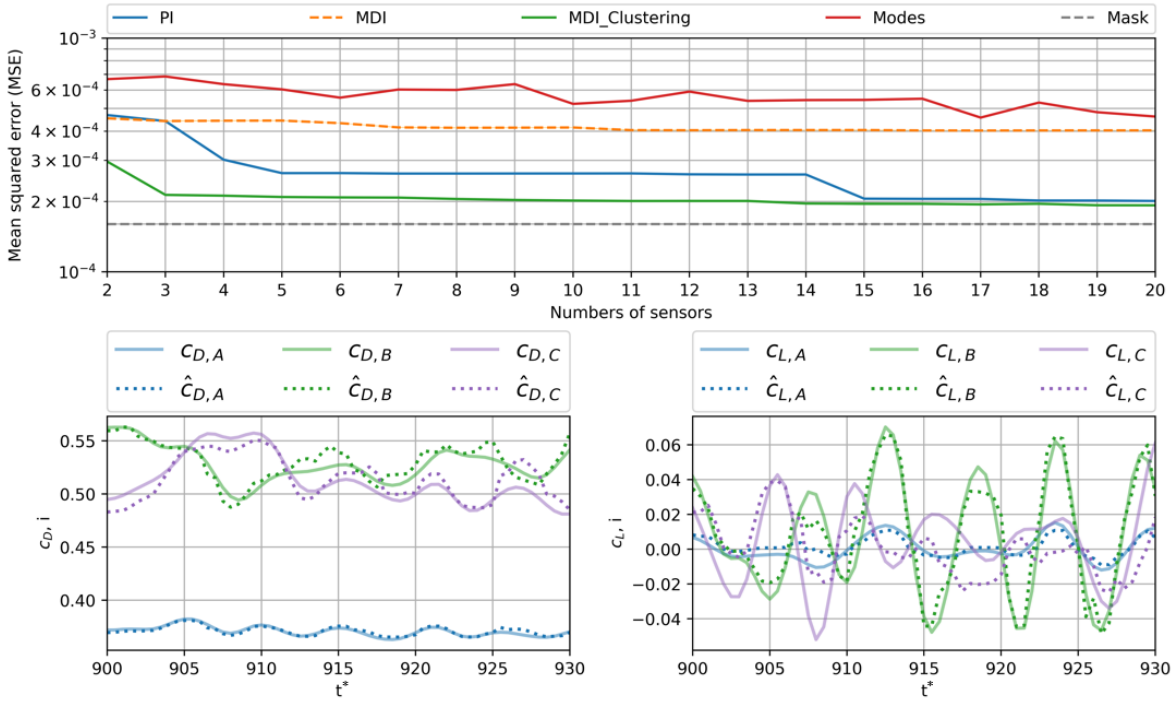

Figure 4.20: Accuracy of the random forest trained for 10 sensor placements on each method on Re=170

Due to the stochasticity induced by the chaotic fluctuations it is expectable that the predictions do not fully match the coefficients. However, as can be seen from the bottom row of figure 4.20 the predictions are capeable of representing all trends in the data with larger deviations observed in the lift coefficient of the bottom cylinder C.

4.2 Actuated Flow

4.2.1 Active Flow Control with Proximal Policy Optimization

The Proximal Policy Optimization (PPO) is a state-of-the-art DRL algorithm which was successfully used in the field of active flow control. PPO is a policy-based method which uses two neural networks, called the actor and critic to approximate the policy $\pi(s)$ and the state-value-function

$V(S)$, respectively. To collect multiple trajectories at the same time, PPO executes multiple environments in parallel storing the results in a reply buffer. After having the buffer filled, the full experience gathered is used to update the actor and critic network [27]. In this study the PPO implementation for AFC past a single cylinder from the drlfoam framework was extended to conduct AFC for the fluidic pinball. In order to produce comparable results, this setup mainly uses the same hyperparameter as in [18], but is not equivalent to it since it utilizes a different CFD solver and DRL framework.

Value and Policy Network

The value network approximates the value function in a nonlinear regression problem with a MSE loss function, parameterized with the weights η as depicted in equation 4.12

$$\eta_{k+1} = \arg\min_{\eta} \frac{1}{|\mathcal{D}_k|\, T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\eta\left(S_t\right) - \hat{R}_t \right)^2 \tag{4.12}$$

where $\mathcal{D}_k$ denotes the number of trajectories collected in the buffer. Updating the weights, indicated by η_{k+1}, is accomplished iteratively with a number of predefined epochs or until a stopping criterion based on the loss function is met. The value network has 2 hidden layers with 512 neurons each and the amount of input nodes depends on the chosen number of pressure probes. The policy network only differs in the output nodes where 6 nodes are used for the fluidic pinball. Each α_i and β_i thereby correspond to the parameters of a beta probability density function (PDF) to individually sample the rotational velocities of the three cylinders, as shown in figure 4.21.

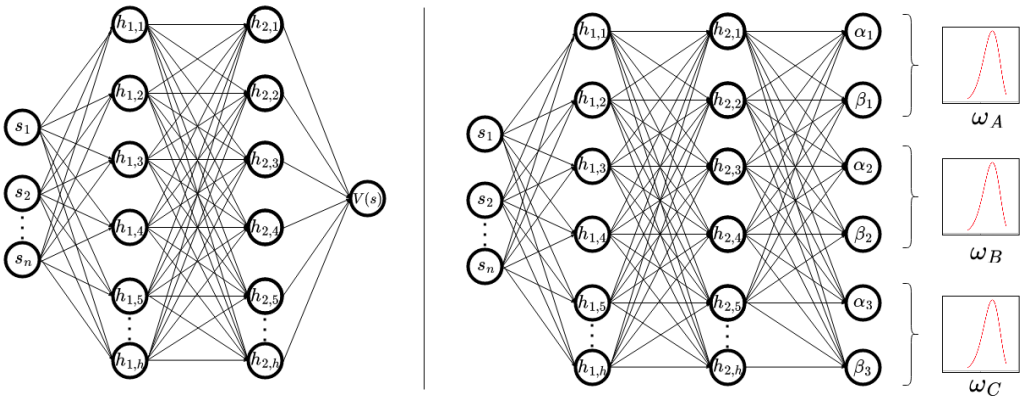

Figure 4.21: Value Network (left) and Policy Network (right)

In PPO the policy gradient is stabilized by updating a clipped policy loss function which means that too high changes of the policy are prevented. The update is based on the generalized advantage estimate (GAE) $G^{\pi_{\theta_k}}$ which is a measure of how much better it is to taken action A_t in state S_t. The update θ_{k+1} is limited with the clipping parameter ϵ [26]

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|\, T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min \left(\frac{\pi_\theta\left(A_t \mid S_t\right)}{\pi_{\theta_k}\left(A_t \mid S_t\right)} G^{\pi_{\theta_k}}\left(S_t, A_t\right), \quad g\left(\epsilon, G^{\pi_{\theta_k}}\left(S_t, A_t\right)\right) \right) \tag{4.13}$$

where

$$g(\epsilon, G) = \begin{cases} (1+\epsilon)G & G \geq 0 \\ (1-\epsilon)G & G < 0 \end{cases} \tag{4.14}$$

In this study ϵ is set to 0.2 [18]. π_{θ_k} denotes the policy from the previous update step. Since the action for the pinball is multidimensional, $\pi_\theta(A_t \mid S_t)$ is the product of the three beta PDFs for the individual cylinders. The beta distribution samples the action over a normalized interval of $[0, 1]$ which is then transformed to the action magnitude interval $[\omega_{min}, \omega_{max}]$. Further details on the PPO implementation and the beta distribution can be found in previous works [33, 11, 28].

During the training the agent executes the current policy to gather trajectories filling the buffer. In this study 10 parallel environments are run. Afterwards, the return and GAE are computed and the policy- and value gradients are backpropagated through the networks using gradient decent with the Adam optimizer. Upon testing the learned policy, the means of the 3 PDFs are sampled in the boundary condition.

Reward function

The reward function depicted in equation 4.15 is the same as applied in previous studies for AFC on the single cylinder.

$$R_t = \overline{c_{D,u}} - (c_{D,t} + \delta|c_{L,t}|) \tag{4.15}$$

The agent maximizes the reward by decreasing the drag coefficient. Moreover an increase in the lift coefficient is penalized where the coefficient δ was set to 0.2. The parameter $\overline{c_{D,u}}$ is the mean drag coefficient of the uncontrolled flow in the training interval normalizing the reward to the area of 0 at the beginning of the training.

Baseline sensor placement

To evaluate the performance of the optimal sensor placements a dense sensor configuration containing 476 sensors from [18] was used to conduct baseline trainings for every flow regime. The sensor locations were adapted to the mesh of this study and are shown in figure 4.22.

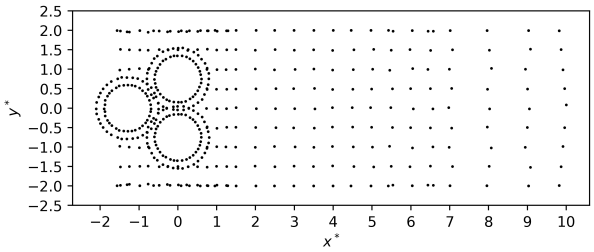

Figure 4.22: Baseline sensor placement adopted from [18]

Simulation parameters

To efficiently learn control laws, the parameters of the interaction between the agent and the CFD simulation have to be carefully set. In [18] the agent was successfully trained interacting on 8 full vortex shedding cycles during a trajectory. Furthermore the number of actuations per trajectory needs to be set. If the agent is allowed to interact too often with the environment, no learning takes place since the effect of a single action execution can not clearly be observed. For $Re = 100$ it was found that 80 interactions yields suitable results whereas for the chaotic vortex shedding case, due to a higher vortex shedding frequency, an increase to 160 interactions per

trajectory lead to a better performance [18]. Table 4.2 summarizes the simulation parameters for the 3 vortex shedding cases covered in this study where dt denotes the timestep of the simulation.

Re	dt [s]	$\overline{c_{D,u}}$	number of actuations per trajectory	duration per action	Simulation time per trajectory
30	0.005	2.01175	80	1.25	100
100	0.005	1.5368	80	0.875	70
170	0.0025	1.38945	160	0.4375	70

Table 4.2: Simulation parameter

4.2.2 Hyperparameter Study

In order to validate the implementation, a hyperparameter study was conducted with the goal to reproduce the results for $Re = 100$ in [18]. The remaining parameters that could not be adpoted from [18] are the action magnitude interval ω_{max} and the reward function which applies a moving average over the coefficients. In a first test, a training for $\omega_{max} = [-5, 5]$ and $\omega_{max} = [-1, 1]$ was carried out. Figure 4.23 a), b) and c) show the resulting rewards, c_D and c_L over a duration of 40 episodes. For $\omega_{max} = [-1, 1]$ nearly no learning takes place since the reward hardly rises and no serious drag reduction can be seen. However for $\omega_{max} = [-5, 5]$ a significant increase in the reward is achieved and $\overline{c_D} \approx 1$ after 40 episodes which corresponds to a mean drag reduction of approximately 35% whereas the reduction in [18] is 27.71%. At the same time an increase in the lift coefficient was discovered which is not occuring in [18] and which may be undesireable in real applications. As a result, in order to mitigate the lift increase, the lift penalization parameter in the reward function was doubled to $\delta = 0.4$. Moreover the number of actuations was increased to 140. A retraining hereby revealed that at the 20th episode, a similar drag reduction was obtained at a much lower $\overline{c_L}$ where also the standard deviation was decreased. In the further course of the training again a massive drop in the drag takes place where simultaneously the lift coefficient starts growing heavily. The maximum reward is reached at episode 33 where the mean drag reduction is 71.4%. To understand the change in the agents strategy, a validation simulation was conducted using the policy at the 20th and 33rd episode. Figure 4.23 d), e) and f) depict the action magnitudes, c_D and c_L over the duration of the validation simulation. Hereby subplot d) reveals that the rotation of the top and bottom cylinder, denoted as ω_B and ω_C, respectively, was roughly doubled from $[-2, 2]$ to $[-4, 4]$ between episode 20 and 33. It can therefore be deduced that the undesired rise in the lift coefficient is conditioned by a too high action magnitude which could be prevented by limiting ω_{max} to approximately $[-2, 2]$ in this case. Because of the limited parallelization capabilities and the duration of 4 days of computing time on a single training run for 40 episodes using the phoenix cluster of TU Braunschweig, the obtained baselines results were not repeated for a lower ω_{max} and $\omega_{max} = [-5, 5]$ was kept. Finally the hyperparameter analysis has shown that the setup was successfully implemented and was able to produce comparable results as presented in [18]. The comparison is summarized in table 4.3.

Case	$C_D \pm$ std	$C_L \pm$ std	C_D reduction
Holm - uncontrolled [18]	3.8407 ± 0.0029	-0.0523 ± 0.0294	0%
Holm - 80 actions [18]	2.7764 ± 0.0170	0.0011 ± 0.1607	27.71%
Holm - 160 actions [18]	2.8056 ± 0.0125	0.1136 ± 0.1271	26.95%
Uncontrolled	1.5381 ± 0.0013	0.0215 ± 0.0095	0%
20 episodes	1.0926 ± 0.0958	-0.0772 ± 0.1633	28.96%
33 episodes	0.2635 ± 0.0318	0.6833 ± 0.1741	82.86%

Table 4.3: Comparison of the coefficients for the baseline training

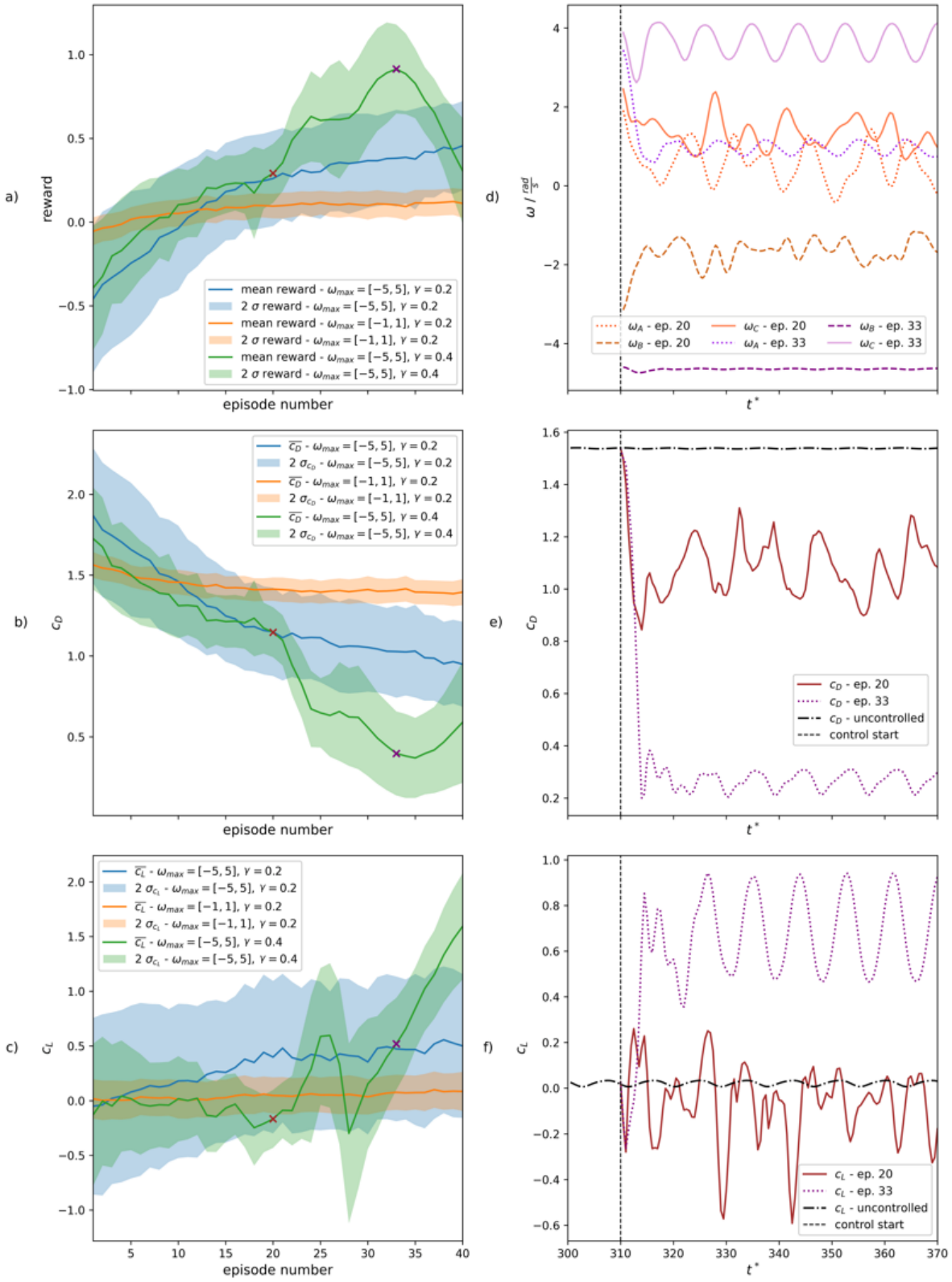

Figure 4.23: Hyperparameter study regarding the reward function parameters and the action magnitude interval for the asymmetric vortex shedding case at $Re = 100$

4.2.3 Attention-based optimal sensor selection

In this section the attention mechanism is introduced which provides a tool to directly integrate and optimize the sensor placement for AFC in the DRL loop. Attention refers to the task of adapting to the relative importance of inputs where the ultimate goal is to learn to only pay attention to a few important input variables and ignore irrelevant ones. To implement attention, a series of additional layers needs to be put in front of a neural network to reduce its input dimension. In the context of this study the chosen neural network is the policy network. Three primary steps are necessary to perform the attention mechanism. In the first step a score function is assigned to the input variables which learns to extract the inputs internal relationships. As the score function the hyperbolic tangent (Tanh) is applied to a linear layer which maps the input $s \in \mathbb{R}^n$ to the interval $[-1, 1]$ where $n = 476$ baseline sensors. Thereby the dimension of the input is compressed to $N_e = 20$ in order to cover for some redundancy in the input variables [37]. This is depicted in equation 4.16

$$\mathrm{e} = \mathrm{Tanh}\left(\mathrm{s}^T W_1 + b_1\right) \tag{4.16}$$

where the output of the layer is the vector $e \in \mathbb{R}^{N_e}$, $W_1 \in \mathbb{R}^{n \times N_e}$ are the weights and $b_1 \in \mathbb{R}^{N_e}$ is the bias. Afterwards the attention weight vector f is computed with another linear layer to generate an attention weight for all n input variables in s

$$\mathrm{f} = \left(\mathrm{e}^T W_2 + b_2\right) \tag{4.17}$$

where $f \in \mathbb{R}^n$, $W_2 \in \mathbb{R}^{N_e \times n}$ and $b_2 \in \mathbb{R}^n$. In the second step of the attention mechanism the attention weights are fed through the softmax function.

$$\kappa_i = \frac{\exp\left(f_i\right)}{\sum_{j=1}^{n} \exp\left(f_j\right)} \tag{4.18}$$

In this operation the attention weights are normalized to $0 \leq \kappa_i \leq 1$ creating a probability distribution such that $\sum_{i=1}^{n} \kappa_i = 1$. This ensures the objective of attention to ignore irrelevant variables by assigning a weight of zero to it and to only take into account a small number of variables with weights greater than 0. Computing the weighted average of the attention weights κ_i and the original input variables formulates the last step of attention resulting in the reduced input vector $\overline{s}$ for the policy network [24].

$$\overline{s} = \sum_{i=1}^{n} \kappa_i \cdot s_i \tag{4.19}$$

During the PPO training the attention weights for the 476 baseline sensors are learned from the updates of the policy network such that a subset of optimal sensors is found maximizing the agents reward. When the training is finished the final attention weights are extracted from the episode with the highest reward. Having all attention weights at hand, a subset of the highest weighted sensors is chosen as the optimal sensors. To finally validate the attention-based sensor placement, the training is repeated only considering the selected optimal sensors. In the following subsection the results of an attention training generating the attention weights for the asymmetric vortex shedding case at $Re = 100$ are presented. The final results after retraining with the optimal sensors are found in chapter 5.0.1.

PPO attention training

Figure 4.24 shows the reward for a PPO training over 40 episodes with $\omega_{max} = [-5, 5]$ and $\delta = 0.4$. The reward reaches a plateau starting at episode 21 where the maximum reward is achieved at episode 25 yielding the subset of optimal sensors. The alternating abrupt changes in the reward after the 25th episode may be caused by a slightly too high learning rate ($l_r = 3\text{e}{-4}$) of the policy update that causes a jumping back and forth in the area of the local optimum.

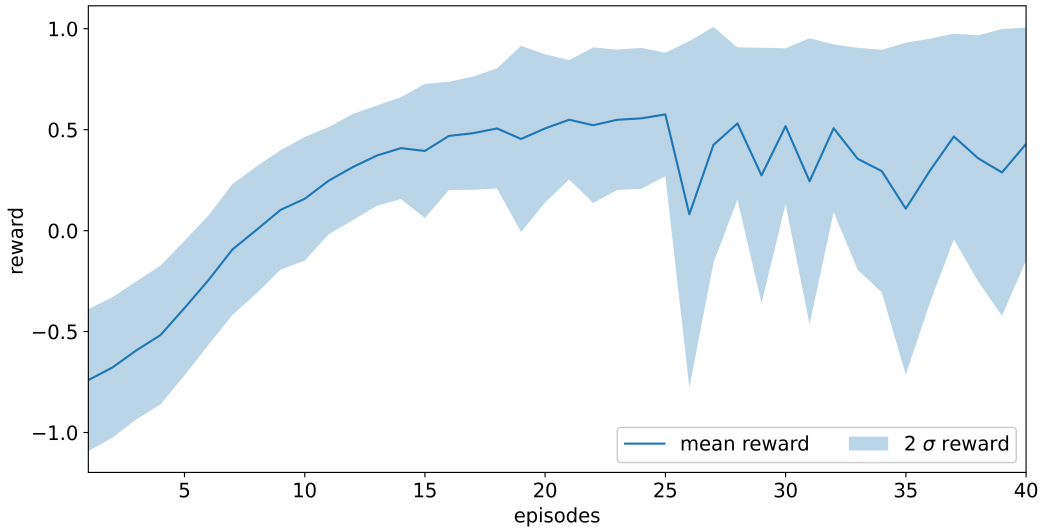

Figure 4.24: Averaged reward for an attention training over 40 episodes at Re=100

To analyze the attention weights learned during the training, a contourplot for the 476 baseline sensors is given in figure 4.25. The attention weights were averaged over all 10 trajectories gathered in episode 25, denoted here as $\overline{\kappa}$. As a result the obtained weights $\overline{\kappa_i}$ do not sum up to one here.

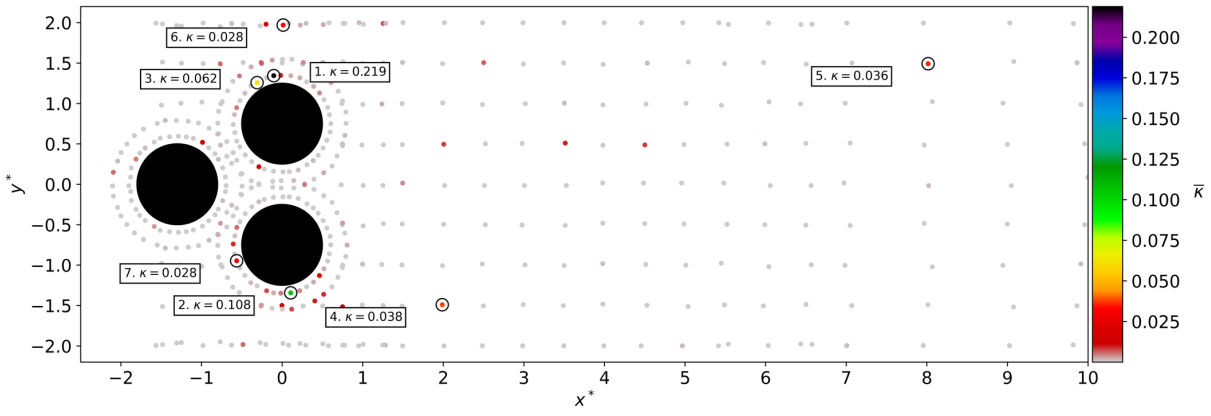

Figure 4.25: Attention weights averaged over all action executions in episode 25

To conduct a fair comparison to the sensor placements from the MDI and the reconstruction, the 7 most attended sensors are marked with the corresponding attention weight. The plot indicates that the agent mostly relies his actions on 4 regions indicated by the sensor numbers 1-5. Hereby the sensors placed directly at the upper surface of the top and lower surface of the bottom cylinder are the most important ones, followed by sensors 4 in the lower half of the wake and sensor 5 in the upper half of the wake. In comparison to the sensor placements of the unactuated flow, the most important sensors are much more aligned to the surfaces of the cylinders. Furthermore the

flow detachment area at the upper surface of the top cylinder and the the flow detachment area at the lower surface of the bottom cylinder are attended with multiple sensors. The distribution of the attention weights therefore also gives insights in the strategy of the agent which relies on counter-rotating the top and bottom cylinder to delay the flow detachment resulting in a drag reduction.

Chapter 5

Active Flow Control Results

This chapter presents the results for AFC of the three vortex shedding regimes and the previous obtained optimal sensor placements. Due to the long training time and the limited computational resources, the comparison of all sensor placement methodologies was restricted to the asymmetric vortex shedding case at $Re = 100$. For the symmetric and chaotic case only a training for the baseline sensor placement was executed.

5.0.1 Symmetric Vortex Shedding

For the symmetric vortex shedding the results of a baseline training for 32 episodes are shown in figure 5.1. The training duration was set to 40 episodes but crashed after the 32nd episode and was not continued due to time constraints.

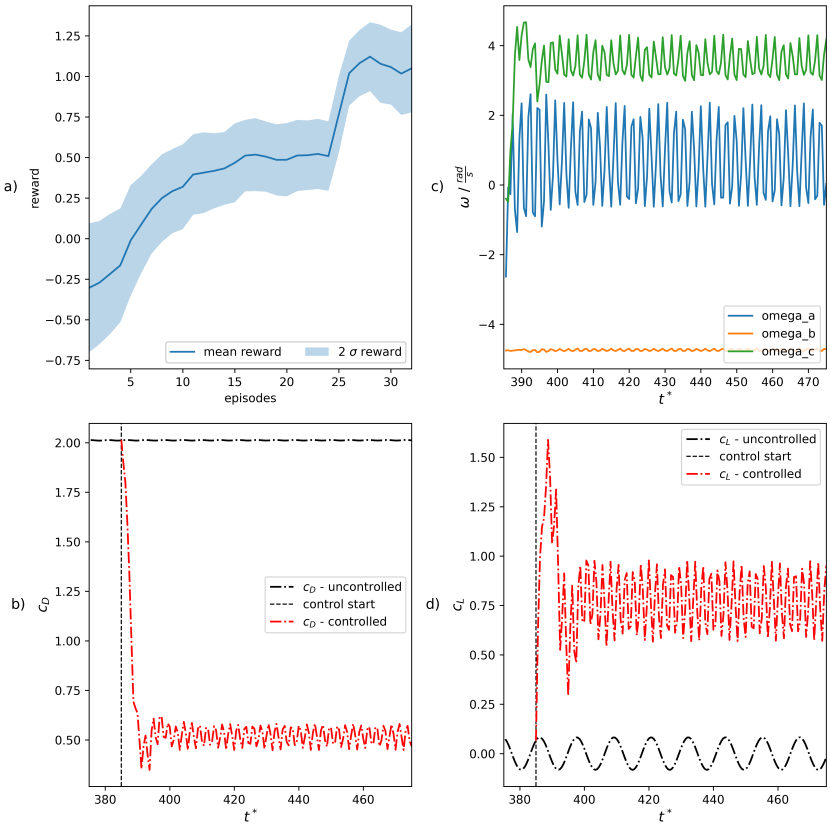

Figure 5.1: Results for active flow control on symmetric vortex shedding at $Re = 30$

It should be noted that the training already crashed on episode 24 as well which results in a reset of the optimizer also affecting the subsequent results. However the reward reaches a maximum of 1.1 at episode 28 whose policy was used to conduct a validation simulation starting the control at $t^* = 385$. Subplot c) hereby shows that the agent rotates the front cylinder periodically at a mean of approximately 1 and the top and bottom cylinder are counter-rotating with similar velocities. The achieved mean drag reduction of the control is 74.11% which comes with an increase in the mean lift coefficient to 0.77, as can be seen in subplots b) and d).

Case	$C_D \pm$ std	$C_L \pm$ std	C_D reduction
Uncontrolled	2.0123 ± 0.0010	-0.0020 ± 0.0582	0%
Baseline	0.5209 ± 0.0448	0.7728 ± 0.1420	$74,11\%$

Table 5.1: Resulting coefficients for a validation simulation with the optimal policies

Concludingly a plot of the norm of the velocity magnitude for the uncontrolled (left) and controlled case (right) is displayed to visualize the control strategy on the flow field. As a result of the counter-rotation of the top and bottom cylinder, the flow separation area at the upper and lower surface of the top and bottom cylinder, respectively, is significantly reduced leading to a shortened tapered form of the recirculation area. Therefore the pressure gradient between the front and back of the top and bottom cylinder is decreased, causing a reduction of the drag forces. At the same time the rotation of the top and bottom cylinder lead to an increase in the pressure gradient between each cylinders upper and lower surface. As a consequence a higher lift is estbalished.

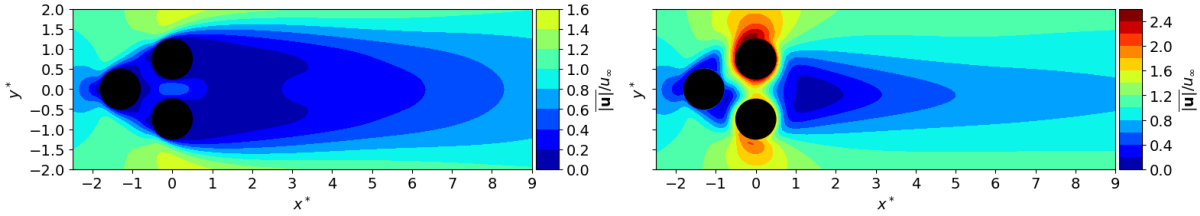

Figure 5.2: Uncontrolled velocity field (left) and baseline controlled velocity field (right)

5.0.2 Asymmetric Vortex Shedding

Figure 5.3 depicts in the upper row the training results for a duration of 40 episodes for the 7 optimal sensors of each method and for the 476 baseline sensors. From the very beginning the steepest increase in the reward is recorded by the attention approach which reaches a local optimum at only 18 episodes and a reward of approximately 0.89. The baseline case reaches a maximum in the reward of 0.92 after 32 epochs, which is nearly twice as long as the attention approach needs. Moreover, in contrast to the attention agent it is not able to find a plateau maintaining the maximum that was found. The methodologies using optimal sensors for the unactuated flow show a lower increase in the reward, both finishing at the same maximum of 0.62. Due to time constraints, only the training for the modes was continued where the reward ended up at a plateau of 0.7.

To validate the learned policies, validation simulations were conducted using the policy after 25 episodes for the attention approach and after 40 episodes for the MDI and the modes, as can be seen in the lower row of 5.3. The results are concluded in table 5.2.

The coefficients for the attention and the baseline approach demonstrate very similar periodic courses where the amplitudes of the coefficients for attention decrease during the validation simulation. For the baseline and the attention case, the mean drag was reduced by 82.86% and 80.98%, respectively, while MDI and the modes achieve reductions of 61.87% and 61.11%.

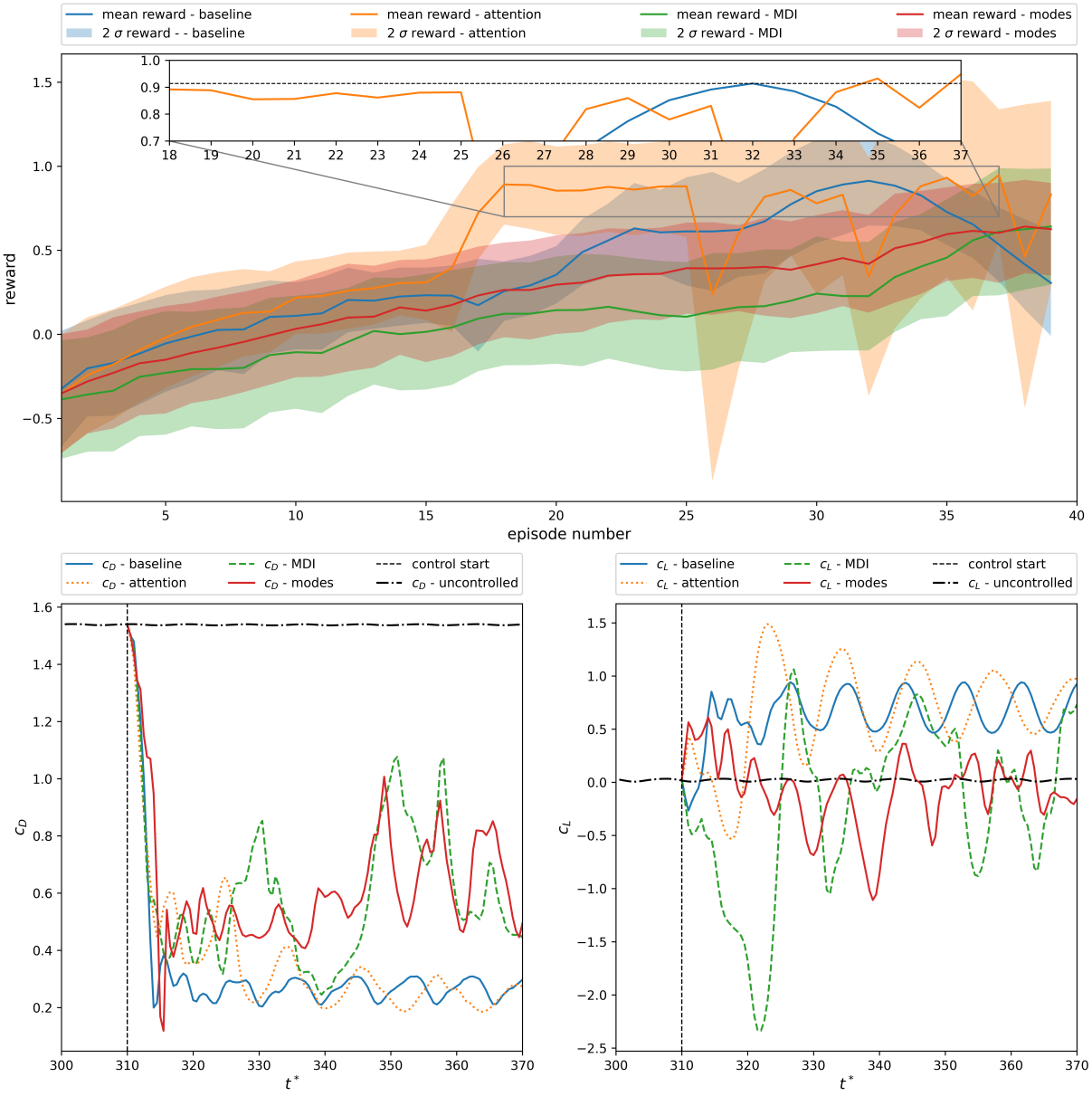

Figure 5.3: Results for active flow control with the sensor placements on asymmetric vortex shedding at $Re = 100$

Case	$C_D \pm$ std	$C_L \pm$ std	C_L reduction
Uncontrolled	1.5381 ± 0.0013	0.0215 ± 0.0095	0%
Baseline	0.2635 ± 0.0318	0.6833 ± 0.1741	82.86%
Attention	0.2925 ± 0.1013	0.7999 ± 0.3248	80.98%
MDI	0.5864 ± 0.2156	-0.1057 ± 0.7703	61.87%
Modes	0.5981 ± 0.1417	-0.1432 ± 0.3093	61.11%

Table 5.2: Resulting coefficients for a validation simulation with the optimal policies at $Re = 100$

To analyze the control laws of the agents, the action magnitudes applied for the validation simulations are plotted in figure 5.4.

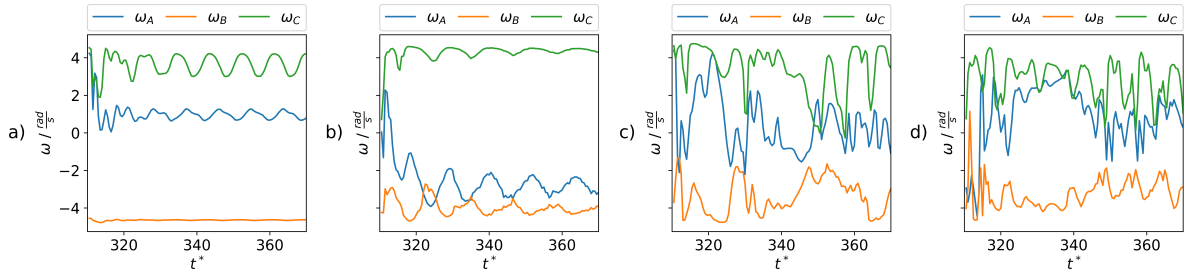

Figure 5.4: Action magnitudes of a validation simulation for: a) Baseline, b) Attention, c) MDI and d) Modes

The results indicate that similar policies were learned for the baseline, MDI and modes cases in which the top and bottom cylinder counter-rotate with on average equal rotational velocities ω_B and ω_C, respectively, and the front cylinder on average rotates with a rather small positive velocity. However the attention agent learns a different strategy where the front cylinders rotation ω_A is negative and oppositely changes with respect to the velocity of the upper cylinder.

The effect of the control on the flow is displayed in figure 5.5 with contourplots of the mean normalized velocity magnitude for the uncontrolled (left) and attention controlled (right) case.

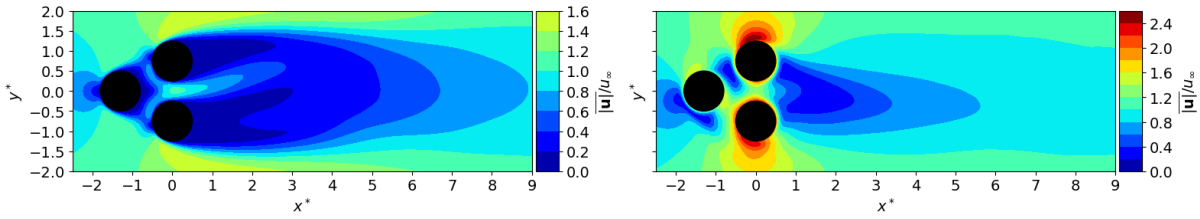

Figure 5.5: Uncontrolled velocity field (left) and attention controlled velocity field (right)

Due to the clockwise rotation of the upper cylinder and counter-clockwise rotation of the lower cylinder, the flow detachment behind the top and bottom cylinders is reduced. This again leads to a cone-shaped reduced recirculation area resulting in a drag reduction and a lift increase, as stated in the previous chapter.

5.0.3 Chaotic Vortex Shedding

Due to the increased complexity in the chaotic vortex flow dynamics, according to [18], training an agent from scratch does not lead to a stable and convergent policy. Therefore a pre-trained model on $Re = 100$ can be utilized as a starting point for training the chaotic vortex shedding case. In the field of ML the idea of benefiting from the knowledge gained in previous tasks is also referred to as transfer learning [40]. Applying the policy of the 476 baseline sensors for $Re = 100$ after 20 episodes to the chaotic regime is displayed in figure 5.6.

As can be seen in subplot a) from the very beginning the reward is already positive yielding that the policy from $Re = 100$ also achieves a drag reduction in the chaotic regime. In a matter of only 12 episodes the reward was doubled reaching a plateau at approximately 0.5 reward. A validation simulation was conducted with the policy gained after 15 episodes where subplot b) shows that contrary to the previous cases the front cylinder rotates alternately with positive and negative direction of rotation at a mean of 0. The mean drag reduction of 63.87% hereby comes at the cost of a periodic lift coefficient with a mean of -0.7135 and an amplitude of roughly 0.5.

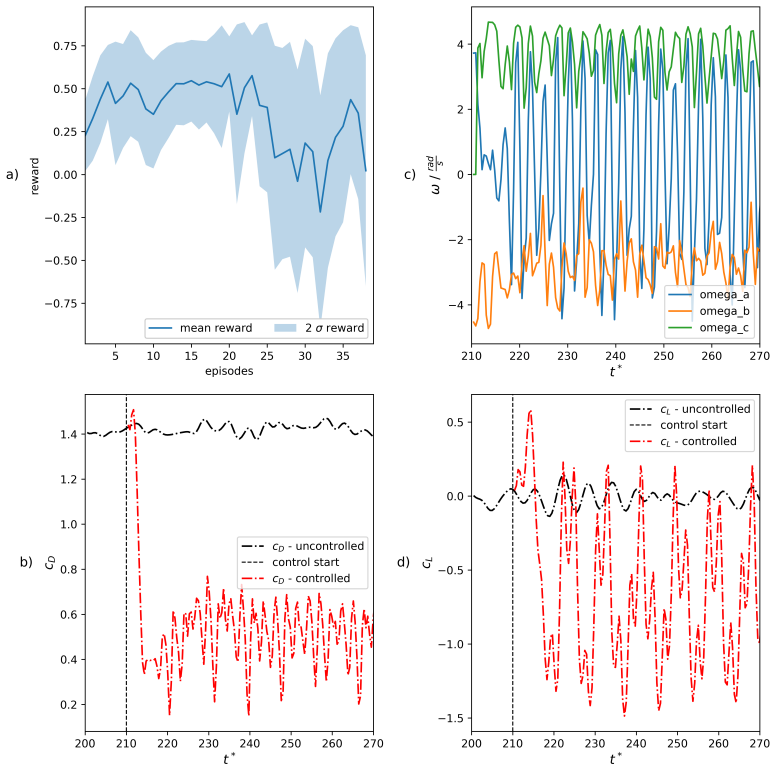

Figure 5.6: Results for active flow control on chaotic vortex shedding at $Re = 170$

Case	$C_D \pm$ std	$C_L \pm$ std	C_D reduction
Uncontrolled	1.4208 ± 0.0208	-0.0114 ± 0.0551	0%
Baseline	0.5133 ± 0.1334	-0.7135 ± 0.4694	63.87%

Table 5.3: Resulting coefficients for a validation simulation with the optimal policies at $Re = 170$

Finally the impact of the control on the velocity field is investigated in figure 5.7. Similar as observed before, the recirculation area is significantly decreased, even more than in the case of $Re = 100$, and forms a somewhat triangular shape behind the top and bottom cylinder. The control strategy also fully supresses the chaotic fluctuating jet between the top and bottom cylinder which is mitigated due to the alternating rotation of the front cylinder.

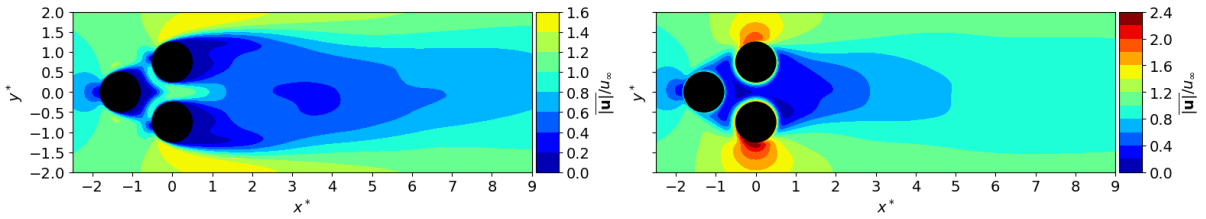

Figure 5.7: Uncontrolled velocity field (left) and controlled velocity field (right) for $Re = 170$

Chapter 6

Summary

Climate change and the associated shift towards sustainable solutions poses big challenges for aircraft manufacturers and the transport industry expecting them to develop more efficient technologies in order to reduce emissions. Innovative ideas are necessary to reach the major goal of zero emissions in the future. One way to achieve this objective is to use passive flow control by means of constructive changes, e.g. riblets in order to reduce the drag forces acting on an aircraft. Another option is active flow control which affects the flow around an object by adding external energy to it. Finding suitable control laws for AFC is mainly limited due to the high dimensionality and nonlinearity of the associated Navier Stokes equations. However the recent progress in data-driven approaches applying machine learning solutions has shed light into the field of AFC where deep reinforcement learning has proven great results facing high dimensional nonlinear state spaces. The first example of AFC with DRL was the 2D flow past a cylinder which is a fairly simple benchmark case yielding good results on reducing the drag and the lift oscillations induced by the so called von Kármán Vortex Street in the wake. Since the cylinder flow does not account for high dimensional real world flow scenarios, a more complex but still cheap to evaluate benchmark consisting of three individually rotateable cylinders in a triangular arrangement was introduced, the so called fluidic pinball. This setup is an ideal candidate to further progress research in the field of AFC with DRL. In order to cope for industrial implementation of DRL based AFC, flow sensors need to be compressed to a minimum to reduce hardware requirements and computational resources. Since control laws for AFC are trained with CFD simulations, optimal sensors can already be found before conducting experiments. Therefore in this thesis optimal sensor placement methodologies for AFC with DRL based on unactuated and actuated CFD simulations were studied on the fluidic pinball.

In a first step, a preliminary analysis regarding the flow dynamics of the fluidic pinball was conducted where three Reynolds numbers accounting for symmetric, asymmetric and chaotic vortex shedding were chosen to be covered in this study. Afterwards simulations were executed to generate training and test data for the two sensor placement approaches on the unactuated flow. To be able to already choose optimal sensors for AFC on the unactuated flow, spatial correlation between the flow field and the lift- and drag coefficients in the DRL reward function was investigated where pressure sensors have been chosen. In the first method optimal sensors are estimated to fully reconstruct the flow field utilizing proper orthogonal decomposition modes. The second approach uses a random forest regressor to predict the lift- and drag coefficients of all 3 cylinders. Hereby two importance scores, namely the mean decrease in impurity and the permutation importance, can be computed which quantiy the relevance of each sensor for the prediction. Since the considered region of the mesh to place the sensors in consists of 27850 mesh cells, computing the importance scores on such a high number of spatially dense placed sensors comes at the risk of assigning false importance values to individual sensors. This may happen due to identical correlation of adjacent sensors which therefore might assign less impor-

tance to a sensor because the same information can be provided by the ones surrounding it. Further the permutation importance breaks the relationship between the temporal evolution of a pressure sensor value and the associated lift and drag coefficients by random permutation of the data. Computing the difference in the prediction accuracy between the permuted dataset and the original dataset leads to the permutation importance score of a single sensor. Permuting the data of a single sensor out of 27850 will likely have not effect on the prediction, therefore demonstrating the necessity of reducing the amount of sensors for the random forest approach. This was accomplished by using latin-hypercube sampling which homogeneously samples the sensor placement space, effectively reducing the amount of input sensors for the random forest. Hereby it was found that upon repeating the sampling of 400 homogeneously placed sensors, computing its importance scores with the random forest and keeping the 4 most important sensors at each iteration, spatially clearly separable pointclouds are formed by the resulting sensors. These pointclouds are then summarized with a k-means++ clustering approach which places a centroid into each pointcloud accounting for the final optimal sensor position. To validate and compare the found optimal sensors for the two aforementioned methodologies, a random forest was retrained with the obtained optimal sensors of both approaches and the prediction accuracy was compared. For the symmetric vortex shedding case, the random forest found 4 optimal sensors and the reconstruction method 7. In the asymmetric vortex shedding case, both methods account for 7 optimal sensors while in the chaotic vortex shedding regime, only the random forest was able to find a suitable sensor placement containing 3 optimal sensors. Overall with each determined sensor placement the corresponding lift- and drag coefficients could be nearly exactly predicted where some deviation was found at the chaotic case which is expectable due to the stochasticity of the chaotic flow dynamics.
In the second part of this work, the PPO implementation for AFC on the single cylinder was adopted to the fluidic pinball case. Hereby the neurons in the hidden layers of the value and policy network were increased to 512 and the policy networks output was increased to 6 output neurons. This ensures that the parameters for 3 separate beta distributions are learned in order to sample a rotational velocity for each cylinder of the pinball. A baseline sensor placement of 476 sensors placed at the cylinders surfaces and in the wake was used to validate the AFC performance of each flow regime. A hyperparameter study regarding the action magnitude interval ω_{max} and the parameters of the reward function was implemented with the result that for rotational velocities greater than 3 rad/s a significant drag reduction was achieved that comes at the cost of an increased lift coefficient which may be undesireable in realistic scenarios. However due to time constraints the initial $\omega_{max} = [-5, 5]$ was kept. In a final step the attention mechanism was implemented which consists of a series of 2 additonal fully connected layers stacked in front of the policy network. Attention incorporates learning optimal sensors for the actuated flow directly in the DRL optimization process where the decisive point is the application of a softmax function that computes an attention weight for each of the 476 sensors. The attention weights, which in total sump up to 1, are multiplied with the state of each sensor which ensures that only a small portion of the sensors are attended by the agent and the majority is ignored, resulting in the optimal sensors for the actuated flow. Due to the long training duration of 4 days on a high performance cluster and the limited computational resources, a training comparing the 3 sensor placement methodologies was only executed for the asymmetric vortex shedding case at $Re = 100$ while for the symmetric and chaotic case only the baseline setup with 476 sensors was trained. To ensure a fair comparison with the each of the 7 optimal sensors for the unactuated flow, the top 7 sensors from the attention approach are used. The comparison of the 3 methodologies at $Re = 100$ revealed that the attention mechanism has achieved near identical results as the baseline case with 476 sensors. The mean drag reduction for attention is 80.98% and 82.86% for the baseline case while the mean decrease in impurity of the random forest achieves 61.87% and the POD modes 61.11%. At the same time the mean lift coefficient increased from 0.0215 to 0.7999, 0.6833, -0.1057 and -0.1432, respectively. It should be noted that the control of the attention agent results in a periodic lift- and drag coefficient with a decaying amplitude

over time which in the end reaches a near equal amplitude as the baseline case. This explains the differnce in the coefficients between the attention and the baseline case. Another interesting outcome is that despite near similar results, the attention and the baseline agent learn slightly different control strategies. All agents have in common that the top and bottom cylinder are counter rotated at near equal velocities. However the attention agent rotates the front cylinder in counter-clockwise direction with a mean of $\omega_A \approx -3$ rad/s and the baseline agent clockwise with $\omega_A \approx 1$ rad/s. As stated in [18], it could be confirmed that transfer learning is possible to train an agent for the chaotic vortex shedding case where a drag reduction of 63.87% was achieved at a lift coefficient of -0.7135.

To summarize, the results show that for all sensor placement methodologies a suitable control law was learned. The attention mechanism yields the best results, effectively reducing the 476 sensors to only 7 without losing performance. For future studies it has great potential to even improve the results.

Outlook

The results presented still have much room for improvement. As previously stated, the rise in the lift coefficient obtained by the control policies is due to a further increase in the rotational velocities of the cylinders. To mitigate this and also to stabilize the training, instead of exploring the necessary limit of the actuations, a better approach generalizing this problem could be to add an energy term to the reward function that somewhat penalizes high action magnitudes. To produce results faster, the phenomena of transfer learning could be applied and further studied to understand the effect of the amount of pretrained episodes on the final reward. To fairly compare the sensor placement methodologies, in this study the top 7 attention sensors were used for $Re = 100$. However sensors 6 and 7 are way less important regarding the attention weights and sensor 1 and 3 are nearly identical. Therefore a more sensible approach would be to use sensor 1,2,4 and 5, as depicted in figure 4.25. Also the hyperparameters of the attention mechanism could be changed to see if even better results can be achieved. Therefore the dimension $N_e = 20$ after the first layer could be varied. Moreover the attention weights can not only be multiplied with the initial states entering the attention layers, but also with the output of the second attention layer on which the softmax function is applied to compute the attention weights. Finally the results presented in this study should be recorded statistically by repeating the training since training neural networks is non-deterministic.

Bibliography

[1] Scikit learn - random forest regressor. `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html` [Accessed: 1.12.2022].

[2] David Arthur and Sergei Vassilvitskii. k-means++:the advantages of careful seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics*, 2007.

[3] Katharina Bieker, Sebastian Peitz, Steven L. Brunton, J. Nathan Kutz, and Michael Dellnitz. Deep model predictive control with online learning for complex physical systems, 2020.

[4] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 10 2001.

[5] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering*. 2019.

[6] Steven L. Brunton, Bernd R. Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52(1):477–508, 2020.

[7] Darshan Thummar. Fluidic pinball. `https://github.com/darshan315/fluidic_pinball` [Accessed: 29.09.2022].

[8] Nan Deng, Bernd R. Noack, Marek Morzynski, and Luc R. Pastur. Low-order model for successive bifurcations of the fluidic pinball. *Journal of Fluid Mechanics*, 884:1, 2020.

[9] Nan Deng, Bernd R. Noack, Marek Morzyński, and Luc R. Pastur. Cluster-based hierarchical network model of the fluidic pinball – cartographing transient and post-transient, multi-frequency, multi-attractor behaviour, 2022.

[10] Nan Deng, Luc R. Pastur, Marek Morzyński, and Bernd R. Noack. Route to chaos in the fluidic pinball, 2018,.

[11] Fabian Gabriel. Aktive Regelung einer Zylinderumströmung bei variierender Reynoldszahl durch bestärkendes Lernen, October 2021.

[12] Stefan Gössling and Andreas Humpe. The global scale, distribution and growth of aviation : Implications for climate change. *Global Environmental Change*, 65:1–12, 2020.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction: Second Edition*. Springer, 2017.

[14] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer US, New York, NY, 2021.

[15] Jose Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic mode decomposition: Data-driven modeling of complex systems*, volume 149 of *Other titles in applied mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, 2016.

[16] Gilles Louppe. Understanding random forests: From theory to practice, 28.07.2014.

[17] Krithika Manohar, Bingni W. Brunton, J. Nathan Kutz, and Steven L. Brunton. Data-driven sparse sensor placement for reconstruction, 2018.

[18] Marius Holm. *Using Deep Reinforcement Learning for Active Flow Control.* Master thesis, University of Oslo, Oslo, 2020.

[19] B. Noack and M. Morzyński. *The fluidic pinball - a toolkit for multiple-input multiple-output flow control: (version 1.0).* Technical report, Poznan University of Technology, Poznan, 2017.

[20] Romain Paris, Samir Beneddine, and Julien Dandois. Robust flow control and optimal sensor placement using deep reinforcement learning. *Journal of Fluid Mechanics*, 913, 2021.

[21] Prof. Dr.-Ing. Rolf Radespiel. *Grundlagen der Strömungsmechanik.* Vorlesungsmanusskript, TU Braunschweig, Braunschweig, 2016.

[22] Jean Rabault, Miroslav Kuchta, Atle Jensen, Ulysse Reglade, and Nicolas Cerardi. Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *Journal of Fluid Mechanics*, 865:281–302, 2019.

[23] Jean Rabault and Alexander Kuhnle. Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach. *Physics of Fluids*, 31(9):094105, sep 2019.

[24] E. Raff. *Inside Deep Learning: Math, Algorithms, Models.* Manning, 2022.

[25] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark computations of laminar flow around a cylinder, 1996.

[26] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2015.

[27] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[28] Erik Schulze. Model-based Reinforcement Learning for Accelerated Learning From CFD Simulations, March 2022.

[29] Erwan Scornet. Trees, forests, and impurity-based variable importance, 13.01.2020.

[30] Richard Semaan. Optimal sensor placement using machine learning, 2016.

[31] Brian M. de Silva, Krithika Manohar, Emily Clark, Bingni W. Brunton, Steven L. Brunton, and J. Nathan Kutz. Pysensors: A python package for sparse sensor placement, 20.02.2021.

[32] Richard S. Sutton and Andrew Barto. *Reinforcement learning: An introduction.* Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts and London, England, second edition edition, 2018.

[33] Darshan Thummar. Active flow control in simulations of fluid flows based on deep reinforcement learning, May 2021.

[34] Mikhail Tokarev, Egor Palkin, and Rustam Mullyadzhanov. Deep reinforcement learning control of cylinder flow using rotary oscillations at low reynolds number. *Energies*, 13(22), 2020.

[35] Jonathan Viquerat, Philippe Meliga, and Elie Hachem. A review on deep reinforcement learning for fluid mechanics: an update. 2021.

[36] P.R Viswanath. Aircraft viscous drag reduction using riblets. *Progress in Aerospace Sciences*, 38(6):571–600, 2002.

[37] Jiawen Wei, Fangyuan Wang, Wanxin Zeng, Wenwei Lin, and Ning Gui. An embedded feature selection framework for control, 2022.

[38] Andre Weiner. drlfoam: Deep reinforcement learning with openfoam. Github repository, 2022. `https://github.com/OFDataCommittee/drlfoam` [Accessed: 1.12.2022].

[39] Julien Weiss. A tutorial on the proper orthogonal decomposition. 2019.

[40] Zhuangdi Zhu, Kaixiang Lin, Anil K. Jain, and Jiayu Zhou. Transfer learning in deep reinforcement learning: A survey, 2020.

List of Figures

List of Tables