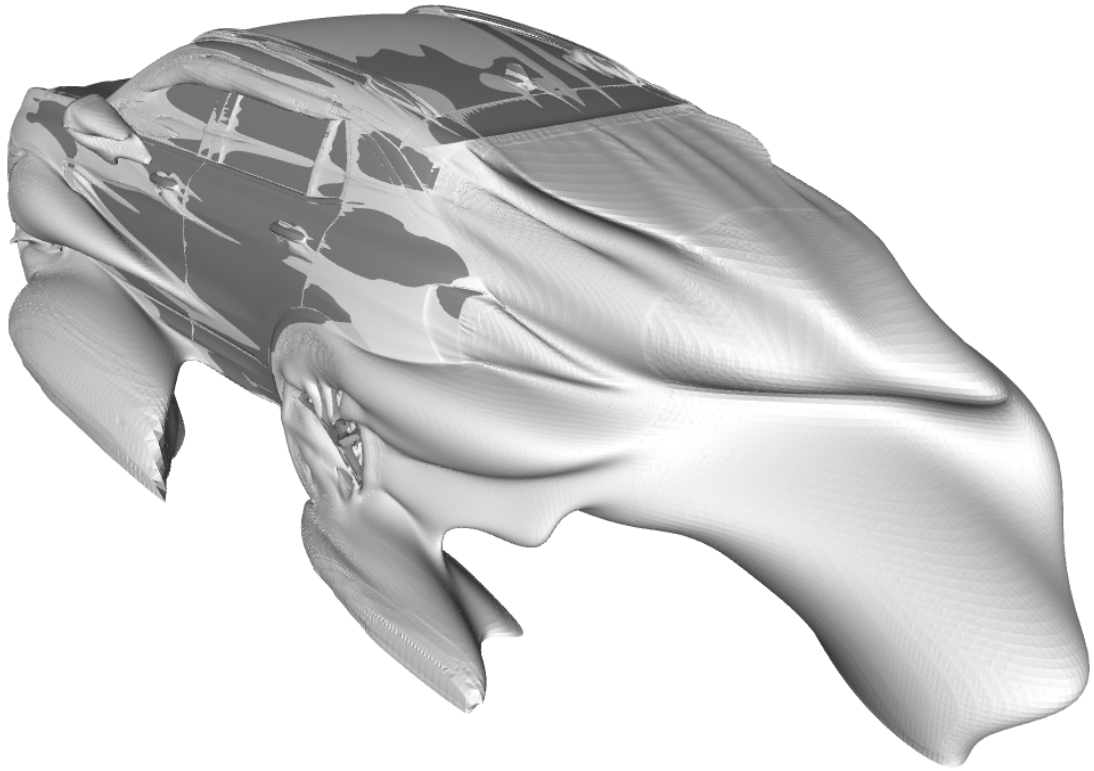
CHALMERS
UNIVERSITY OF TECHNOLOGY



Shape Optimisation of Vehicles for Cross-wind Stability Using Neural Networks

Engineering Mathematics and Computational Science

AMANDA SJÖLAND

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2020

Master's thesis 2020:34

Shape Optimisation of Vehicles for Crosswind Stability Using Neural Networks

AMANDA SJÖLAND

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
Gothenburg, Sweden 2020

Shape Optimisation of Vehicles for Crosswind Stability Using Neural Networks
AMANDA SJÖLAND

Supervisor: Samuel Gabriel, CEVT
Supervisor: Adam Brandt, CEVT
Examiner: Sinisa Krajnovic, Department of Mechanics and Maritime Sciences

Master's Thesis 2020:34
Department of Mechanics and Maritime Sciences
Division of Fluid Dynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Wake visualisation showing a surface of zero total pressure at crosswind.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2020

Shape Optimisation of Vehicles for Crosswind Stability Using Neural Networks
AMANDA SJÖLAND
Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

Recent advances in computational power and the availability of large labelled data sets have contributed to the progress and interest in deep learning. One product development area associated with vast amounts of data is aerodynamics and previous studies have successfully applied neural networks to predict aerodynamic coefficients. The aerodynamic development process for a passenger vehicle usually focuses on the aerodynamic drag force and the lift force, with air flow coming head on. Fewer efforts are spent on investigating how all aerodynamic forces and moments are affected by strong crosswinds. It is excepted that the aerodynamic lift and side force together with the yaw moment affect driving stability. Furthermore, driving stability issues are often discovered late in the development process and methods for assessing this issue virtually are desired. Thus, a method for investigating the optimal shape changes of a passenger vehicle to minimise crosswind sensitivity was developed. Simulated data was created by generating 120 CAD models, each with a unique configuration of five design parameters: amount of corner filler below the roof spoiler, length and height of roof spoiler and the width and length of the back-light. A CFD simulation was performed on each configuration with a flow-angle of 7.5° and a driving velocity of 100 km/h. Three neural networks were trained to predict the drag, rear lift force and the yaw moment, respectively. The effect of each design parameter was studied, as well as the optimal configuration for minimising each coefficient. A genetic algorithm was used to find the settings for minimising yaw-moment and rear lift force, while retaining other coefficients within set requirements. The final design was estimated to decrease the yaw moment by 4.15 %, but due to the small variation of yaw moment in the generated data set, the results are uncertain. The optimal design to minimise C_{LR} achieved a decrease of 0.0185 in rear lift.

Keywords: neural networks, CFD, stability, yaw moment, rear lift force, genetic algorithm.

Acknowledgements

I would like to express my deepest gratitude to my supervisors: Samuel Gabriel and Adam Brandt. Without Sam's belief in me, this thesis would never have been written. Completion of this thesis would also never have been possible without the incredible support from Adam. Thank you both so much!

I would also like to thank Sofia Ore for all the kind advice. I'm extremely grateful to Lili Hansson and Patrik Sondell for inviting me to join the wonderful CAE Energy team at CEVT. And thank you Niclas Dagson for keeping me around.

I am also grateful to Robert Moestam for his invaluable suggestions. My final thanks goes to my patient examiner: Sinisa Krajnovic.

Amanda Sjöland, Gothenburg, June 2020

Contents

1

Introduction

1.1 Background

The artificial neuron, a simplified version of its biological counterpart, was first described in the 1940's by McCulloch and Pitts [1]. By connecting these artificial neurons into a network, Rosenblatt found a novel way to process information [2]. Recent advances in computational power, as well as the availability of large labelled data sets, have contributed to the progress and interest in deep learning (DL). In the automotive industry, DL "is paving the way for the delivery of unprecedented automotive innovations that will disrupt the status quo and deliver an enriched user experience" [3]. What comes to mind for most are the advances in autonomous driving, but this type of technique is also expected to influence data-driven product development [3].

One product development area associated with vast amounts of data is aerodynamics. The aerodynamic development process for a passenger vehicle usually focuses on reducing the aerodynamic drag force and sustaining the lift forces at the front and rear axle within levels of requirement. Fewer efforts are spent on investigating how the aerodynamic forces and moments are affected by strong crosswinds. While driving during these conditions, issues with stability may arise which results in an unpleasant experience for the driver. Problems associated with driving stability are often discovered late in the development process, and more research is needed on how they can be prevented [4]. However, it is assumed that the aerodynamic yaw moment and rear lift force greatly affect the stability [4, 5]. By introducing DL techniques, effective shape optimisation can be performed to find design parameters related to the significant vehicle motions affecting the driving stability at crosswinds.

Shape optimisation with computational fluid dynamics (CFD) can be done by manually updating design parameters after each simulation is finished, but this is a time intensive and therefore costly procedure. These problems can be circumvented by creating a surrogate model, or meta-model, in the form of a deep neural network, which approximates the results from the CFD solver. Previous studies have successfully predicted aerodynamic coefficients using neural networks, with both numerical and visual input [6–9]. Wallach et al. [6] present four benefits to incorporating a neural network based meta-model in conceptual studies for aerodynamic performance of aircrafts, which apply equally to the automotive industry: possibility to interpolate

results for a larger population than traditional CFD, increased size of possible search space due to faster computational time, possibility to accumulate knowledge from an increasing number of project through adaptive learning algorithms and lastly, the ability to use multiple input sources, supplementing the CFD results with data from wind tunnel experiments or driving tests. The first two points, concerning interpolation and increased search space, provide the possibility of effectively finding the optimal set of input parameters for a desirable output.

While there are a number of previous studies proving that neural networks are a feasible method for calculating aerodynamic coefficients, like in aerodynamic development as a whole, the main focus of these studies has been on air flow meeting the vehicle head on [6, 8, 9]. Most research is focused on understanding how the aerodynamic drag is affected by design changes. Fewer studies have been conducted on how the design changes affect the aerodynamic yaw moment and rear lift force under crosswinds. By exploring less prominent domains where CFD and neural networks intersect, this project could shed light on design solutions to problems rarely discovered in the virtual development process, namely the sensation of instability when the vehicle is subjected to strong crosswinds.

1.2 Objective and aim

The main aim of this thesis is to combine the fields of deep learning and aerodynamics to develop a method for investigating the optimal shape changes of a passenger vehicle to minimise crosswind sensitivity. The objectives are to:

- Generate simulated training data
- Create and optimise a neural network
- Find the optimal design configuration
- Investigate how the design affect aerodynamic yaw moment and rear lift force

1.3 Demarcations

In the generation of the simulated data, only steady-state aerodynamic simulations will be run since transient crosswind simulations are too time consuming within the scope of this thesis. Furthermore, the number of iterations is limited to 3000 runs for each CFD simulation, to save time. Finally, the CAD model of the vehicle has been simplified from the original version with closed front and wake regions on the under-carriage blocked off by surfaces. Both these measures are done to reduce volume cell count and increase simulation speed.

2

Theory

The thesis uses concepts from experimental design, fluid mechanics, computational fluid dynamics, artificial neural networks and optimisation algorithms to create a method for finding the optimal shape improvements.

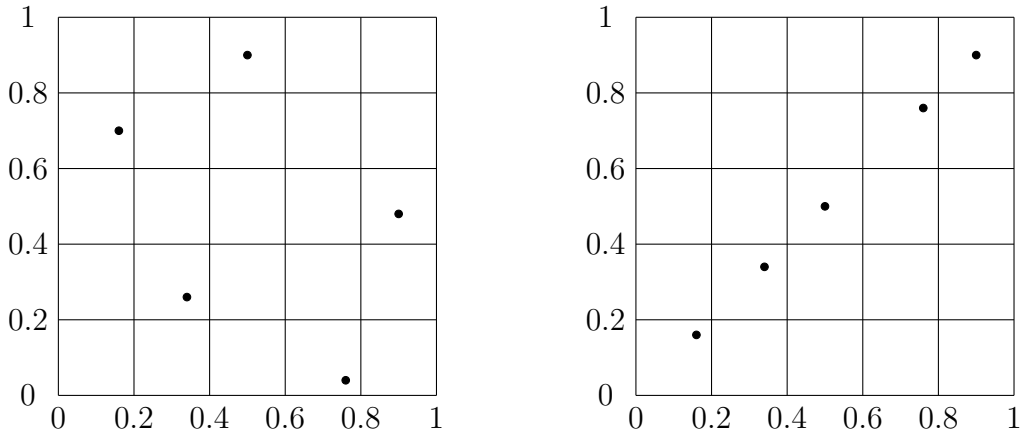
2.1 Design of Experiments

The design of the computer experiments determines how accurately and efficiently the effects of the multiple design parameters, as well as the effects of their interactions, can be predicted. In a computer experiment, the sampling method has to account for the deterministic output of the simulations and how time-consuming computational fluid dynamic simulations can be [10]. Furthermore, one would like designs for computer experiments to be space-filling when prediction accuracy over the entire experimental region is of primary interest, meaning that the generated designs should spread values evenly over the ranges of the parameters [10].

2.1.1 Latin Hypercube Sampling

Latin Hypercube Sampling (LHS) was presented as a method to use for selecting input values for simulations [11]. Figure 2.1a shows a simple case with a unit square $[0, 1]^2$ as the experimental region and consisting of five points. Each axis is divided into five rows or columns, and the five samples are selected so that each row and column is represented once. However, it is not guaranteed that the design is space-filling over the entire experimental region, as shown in Figure 2.1b.

By maximising the minimal distance between points, no two points will be too close to each other which ensures a spread over the experimental region. This is known as maximin design [10].



(a) Space-filling design. **(b)** Non-space-filling design.

Figure 2.1: Latin Hypercube Design with five points with a unit square $[0, 1]^2$ as the experimental region.

2.2 Fluid mechanics

The forces that oppose the forward motion of a passenger vehicle at constant speed on a flat road come primarily from the aerodynamic drag and the rolling resistance of the wheels [12]. Aerodynamic drag is defined as

$$F_D = \frac{1}{2}\rho V^2 A C_D$$

and becomes the dominant force when the speed of the vehicle exceeds 65-80 km/h [12]. C_D is the drag coefficient, A is the projected frontal area of the vehicle, ρ is the density of air and V is the speed of the vehicle relative to the air. The lift and side forces are similarly defined as

$$F_L = \frac{1}{2}\rho V^2 A C_L \quad \text{and} \quad F_S = \frac{1}{2}\rho V^2 A C_S$$

where C_L and C_S are the lift and side force coefficients respectively. When evaluating the stability of the vehicle, both yaw and pitch moments have to be considered. Pitch, roll and yaw moment are defined as

$$M_P = \frac{1}{2}\rho V^2 A C_{PM} L\,, \quad M_R = \frac{1}{2}\rho V^2 A C_{RM} L \quad \text{and} \quad M_Y = \frac{1}{2}\rho V^2 A C_{YM} L$$

respectively, where L is the wheel base of the vehicle and C_{PM}, C_{RM} and C_{YM} are the pitch, roll and yaw moment coefficients respectively.

2.3 Computational Fluid Dynamics

Computational Fluid Dynamics is a branch of fluid mechanics where numerical analysis is used to solve and analyse problems involving the flow of fluids. The domain

is discretised into finite volumes and the governing partial differential equations, concerning the conservation of mass and momentum, are solved for each volume. The governing equations are also known as the Navier-Stokes equations. Time averaging these equations and assuming the mean flow is steady results in the Reynolds Average Navier-Stokes equations (RANS). The continuity equation is written as

$$\frac{\partial \bar{v}_i}{\partial x_i} = 0$$

and the momentum equation as

$$\rho \frac{\partial \bar{v}_i \bar{v}_j}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right) - \rho\overline{v_i' v_j'}\right).$$

The final term in the momentum equations is called the Reynolds stress tensor and represents the turbulence of the flow. This term needs to be modelled in order to close the system of equations. The governing equations can then be solved for the computational domain, and any forces acting on a body in the flow can be calculated from the pressures and friction on the nodes adjacent to the body.

2.3.1 Turbulence model

The Boussinesq assumption introduces a turbulent viscosity, ν_t, to model the Reynolds stresses in RANS equations. The Reynolds stresses are expressed as

$$-\overline{v_i' v_j'} = \nu_t\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right) - \frac{1}{3}\delta_{ij}\overline{v_k' v_k'}$$

which means that the RANS momentum equation is rewritten as

$$\rho \frac{\partial \bar{v}_i \bar{v}_j}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left((\mu_t + \mu)\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right)\right).$$

In the realisable $k-\epsilon$ turbulence model, the turbulent viscosity is modeled by solving two partial differential equations representing the transportation of turbulent kinetic energy, k, and the dissipation of turbulent kinetic energy, ϵ. The modelled equation for k reads

$$\bar{v}_j \frac{\partial k}{\partial x_j} = \nu_t\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right)\frac{\partial \bar{v}_i}{\partial x_j} - \epsilon + \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]$$

and the modelled equation for ϵ is

$$\bar{v}_j \frac{\partial \epsilon}{\partial x_j} = \frac{\epsilon}{k} c_{\epsilon 1} \nu_t\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right)\frac{\partial \bar{v}_i}{\partial x_j} - c_{\epsilon 2}\frac{\epsilon^2}{k} + \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial \epsilon}{\partial x_j}\right].$$

The turbulent viscosity is computed with the formula

$$\mu_t = c_\mu \rho \frac{k^2}{\epsilon}$$

where c_μ is not a constant. This formulation prevents unphysical behaviour.

2.3.2 Convergence

To successfully obtain accurate simulation results, the numerical results must have achieved convergence. For convergence to be obtained, the residuals, i.e. the measurement of conservation of the flow properties, have to be small and the quantities of interest must have reached stable values [13]. The limit for when the residuals are deemed small enough is usually set as a numerical value a key residual has to go below, or when a set number of iterations has been reached. This means that it is not certain that the results have converged if the number of iterations are too few. This can be evaluated by studying the fluctuations, for example by calculating the moving average numerical value but always starting from a set iteration number. This technique would show if the average result was still increasing at the iteration limit.

2.4 Artificial Neural Networks

Neurons are the fundamental units of computations in the mammalian brain. These neurons form networks in the brain to process data and the data-processing capacity of the networks can be improved by "establishing reconnections between neurons". These biological neural networks have inspired the creation of artificial adaptations where the fundamental principle of reconnection between computational nodes is realised. An artificial neural network can learn to recognise patterns in a set of data, known as a training set, and later discern these patterns in a new data set.

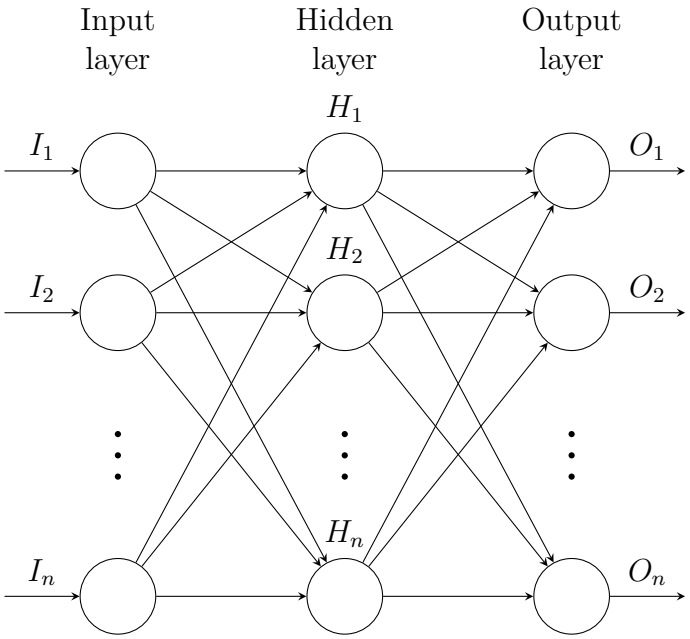


Figure 2.2: Schematics of a neural network with one hidden layer.

A network consists of an input layer, an output layer and a number of hidden layers between these two, as shown in Figure 2.2. In a feedforward network, the input

signals are propagated forward through the network where they are transformed by the weights and thresholds of each layer. The learning of the network is done by increasing och decreasing the strength of the signal between neurons through updates to the weights and thresholds, based on how far the output is from being correct. The difference between the output from the network and the actual output can be determined with the mean square error (MSE) as the loss function. This is calculated as

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - O_i)^2$$

where n is the number of outputs from the network, y_i is the actual value and O_i is the networks prediction. The decision to activate at all is done by an activation function, similar to the biological behaviour of neurons. For a function approximation network, the most common choices are the sigmoid or hyperbolic tangent curves. These curves estimates a probability of whether the neuron should activate or not, given the weights, threshold and current input.

It is important to note that while neural networks provide excellent results as universal function estimators or image classifiers, their performance can only be as good as the data they are trained on. This means that any bias or uncertainty present in the training set will be reflected in the model it produces [14].

2.4.1 Hyperparameters

For the network to be able to properly make predictions on data it has not been trained on, the hyperparameters need to be correctly tuned. The hyperparameters are the framework for the network which the network cannot learn by itself through training. These mainly include the number of layers, how many nodes each layer will contain, the learning rate, activation function and which optimiser to use. The layout of the network affects whether the network will be able to produce non-linear responses, but too many layers can lead to overfitting and a decrease in accuracy for new data. To prevent overfitting, different regularisation techniques can be employed.

2.4.1.1 Regularisation

As the number of neurons in a neural network increases, the risk of overfitting also rises. If the weights in the neurons are allowed to grow too large, the network will have larger responses too smaller changes, making the network less robust and more susceptible to noise. This can be prevented by introducing weight-decay schemes which reduces the weight during training, which makes the network better at generalisation. One common scheme is known as L_2 regularisation, which introduces an additional term when computing the energy function that determines how the weights should be updated.

2.4.1.2 Hyperband

There are many different methods for selecting the best hyperparameters, such as grid search and the superior random search. In 2018, the Hyperband method was conceived as an extension to the random search where adaptive resource allocation and early-stopping were implemented [15]. The Hyperband algorithm trains a large set of random configurations for only a few iterations; the configurations with the best performances are allowed to run longer. The schedule for how many iterations each network is allowed to run for is determined by two settings: the maximum number of iterations a network is allowed to train for and the fraction of configurations that are discarded in each round. This hyperparameter optimisation algorithm is formulated as a pure-exploration non-stochastic infinite-armed bandit problem where a predefined resource is allocated to randomly sampled configurations and has been shown to have an order-of-magnitude speedup over other popular optimisation schemes.

2.5 Genetic Algorithm

Inspired by the process of evolution, a genetic algorithm can be used to solve optimisation problems by introducing the concepts of generations, cross-breeding, mutation and survival of the fittest to the set of possible solutions. A set of suggested solutions is called a population, which can be initiated as random values. The performance of the individuals in the population is evaluated with a fitness function and each individual is given a fitness score. The fitness function reflects the optimisation problem at hand. The score is used to determine which solutions should survive and generate offsprings. The population of the next generation is created by combining the values of the most fit individuals of the previous generation. To introduce new possible values the individuals can have, mutation is implemented to randomly alter a number of values among the new population. The algorithm is run for a set number of generations, or until a satisfactory fitness score is reached.

3

Methods

The method for finding the optimal design changes can be split into three sections: the generation of the simulated data, building the neural network and finding the optimal shape. The workflow is depicted in Figure 3.1, where the main output from each section is highlighted.

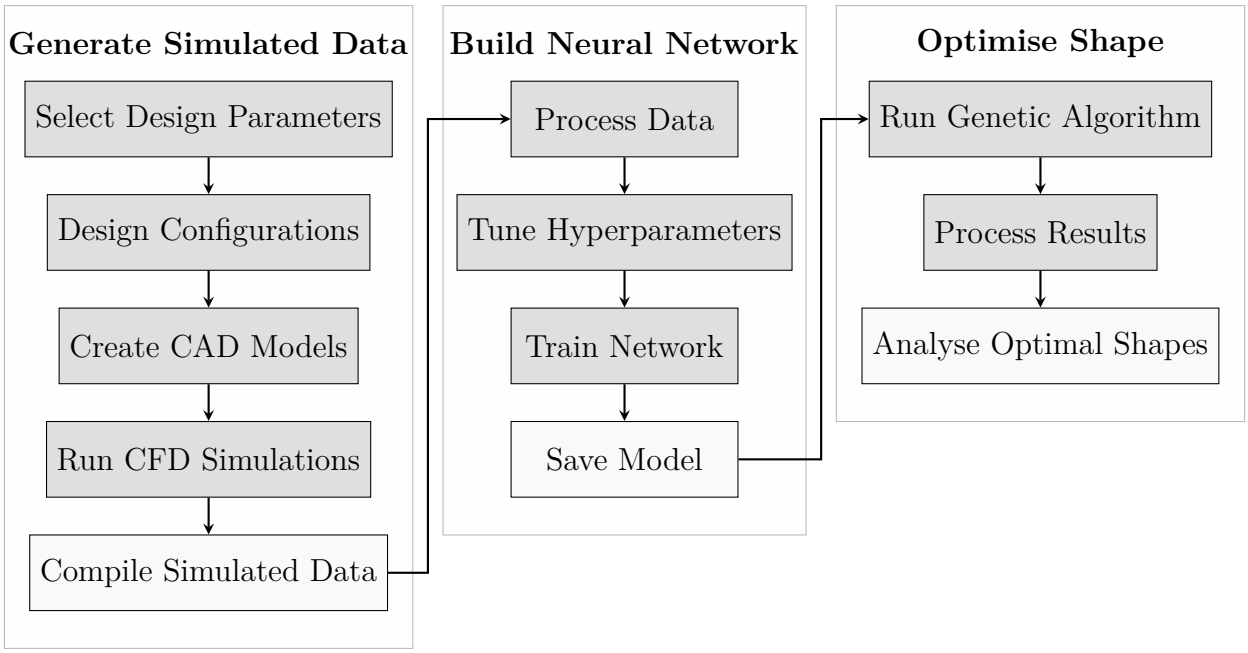


Figure 3.1: Flowchart of the method for finding the optimal design changes.

3.1 Generate Simulated Data

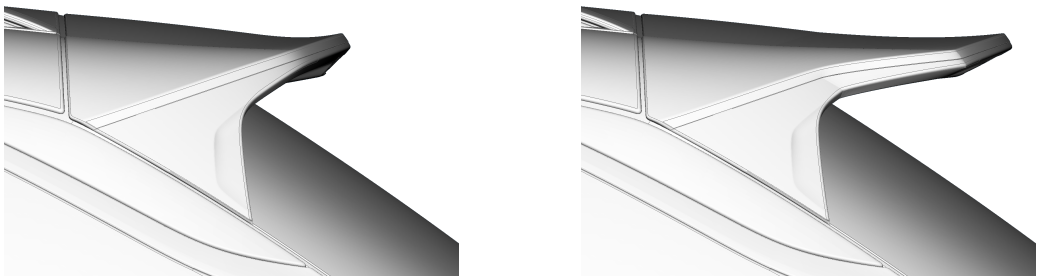
The method for producing the CFD simulation results followed the current workflow used by the aerodynamic group at CEVT, with additional steps to create a design space suitable for a simulated data set.

3.1.1 Design Parameters

Five design parameters were selected: the length and height of the roof spoiler trailing edge, the width and length of the back-light and the area covered on the side of the back spoiler sometimes referred to as corner filler. The maximum and minimum values of the parameters spans the design space and are defined as the distance from the original position in millimetres, see Table 3.1. These design changes were selected since they were thought to be important for stability. The vehicle is a personal vehicle that needs to respect some design requirements. Hence the different designs were set to lie within reasonable limits.

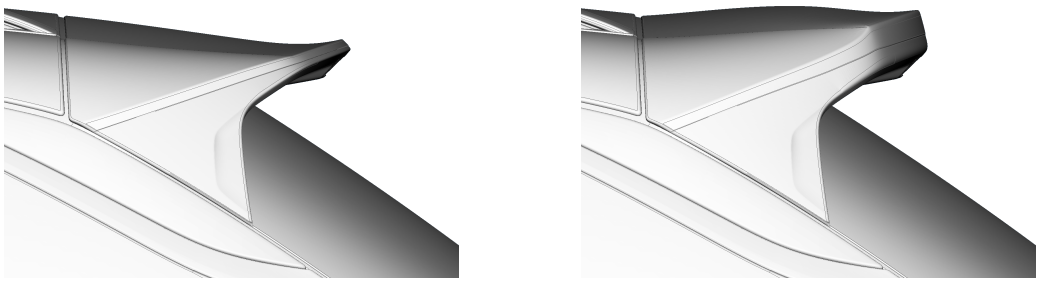
Table 3.1: Selected design parameters with maximum and minimum allowed changes from original position.

Design Parameter	Minimum Value [mm]	Maximum Value [mm]
Length of roof spoiler	0	100
Height of roof spoiler	-5	30
Length of back-light	-20	40
Width of back-light	-10	50
Corner filler	-160	160



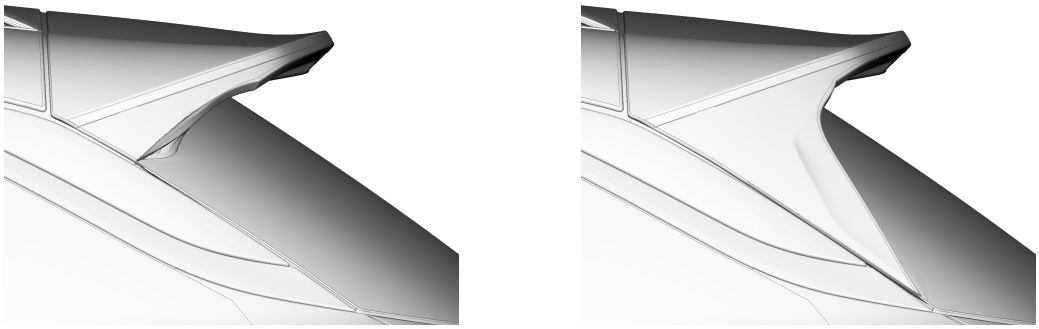
(a) Minimal setting. **(b)** Maximal setting.

Figure 3.2: Minimal and maximal change to length of roof spoiler.



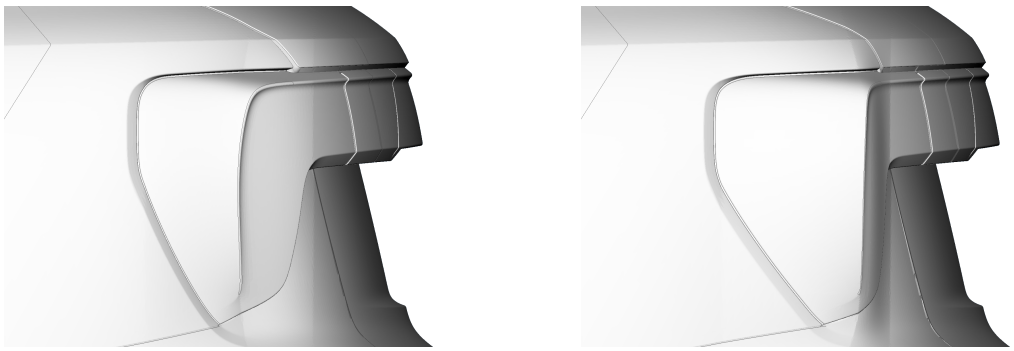
(a) Minimal setting. **(b)** Maximal setting.

Figure 3.3: Minimal and maximal change to height of roof spoiler.



(a) Minimal setting. **(b)** Maximal setting.

Figure 3.4: Minimal and maximal change to corner filler.



(a) Minimal setting. **(b)** Maximal setting.

Figure 3.5: Minimal and maximal change to length of back-light.

(a) Minimal setting. **(b)** Maximal setting.

Figure 3.6: Minimal and maximal change to width of back-light.

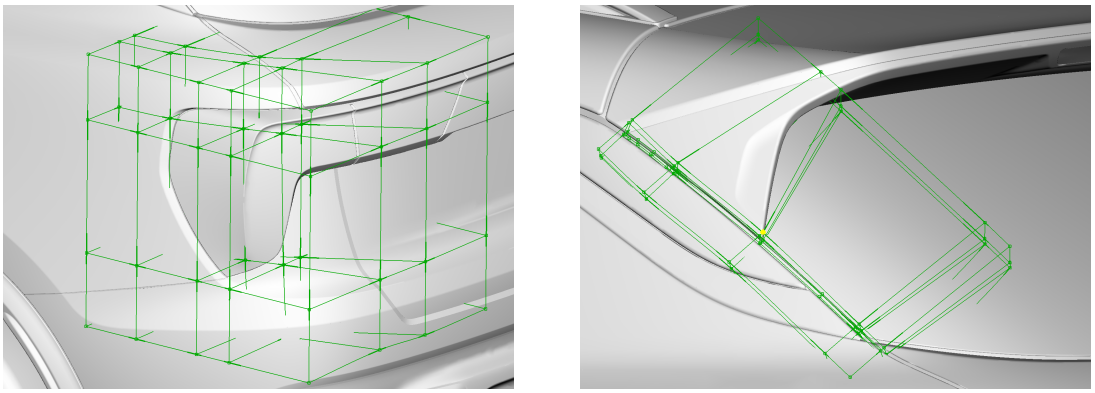
3.1.2 Design Configurations

The configurations of the design parameters were selected using latin hypercube sampling (LHS) with a maximin distance design to provide accuracy in the predictions. By designing the experiment with LHS, the number of simulations can be

kept at a minimum, while still filling the experimental region which is necessary for a training a network. Implementation of LHS was done in Python, where the input were the extreme values of each parameter and the number of designs. The output was a text-file with 120 combinations of the five parameters spaced out to fill the design space. The number of designs was set to 120 due to limitations in computational resources.

3.1.3 CAD Models

The base model of the car was the Lynk & Co 01 developed by CEVT. The model was slightly altered to decrease cell count and speed up simulation time. The modifications included sealing off less interesting fluid regions, basically wake regions between the under-body and the top hat and also blocking off the cooling flow. 120 different versions of this vehicle were created using the morphing and optimisation tools in Beta CAE's software Ansa. The optimisation tool was used to input the design parameters, which would use morphing to create the new version. The models were exported as surfaces, as part of the current CFD workflow at CEVT.



(a) Morphing box enclosing the back-light.

(b) Morphing box for adjusting the corner filler.

Figure 3.7: Placement and design of morphing boxes in green.

3.1.4 CFD Simulations and Data Compilation

The CFD simulations used a mesh created in Sharc's software Harpoon with settings similar to the regular side wind CFD analysis at CEVT following the sidewind method proposed by S. Gabriel [16], but adapted to fit the scope of the thesis. This was done to keep the focus on the deep learning, and not on the mesh or CFD in general. Furthermore, the setting of the simulations were selected to promote quick convergence. The turbulence model was the realisable $k - \epsilon$ model with a non-equilibrium wall function, run for 3000 iterations. The wind speed was set to 100 km/h in the $x-$direction and at a flow angle, φ, of 7.5°, with the coordinate system shown in Figure 3.8. The angle was chosen to be large, but still within reasonable limits of expected on-road crosswind conditions [4]. Figure 3.8 also depicts a representation of the computational domain, not to scale, which has inlets on two

sides and outlets on two sides. The inlets were both velocity inlets with the same magnitudes in the x- and y directions. Both outlets were defined as pressure outlets. The domain is larger on the two sides opposite the inlets as to not have the wake interfere with the boundaries. The simulations were run in ANSYS' software Fluent (version 19.0.0). To highlight the irregularity of wakes during side wind, Figure 3.9 shows the iso-surface where the total pressure is zero.

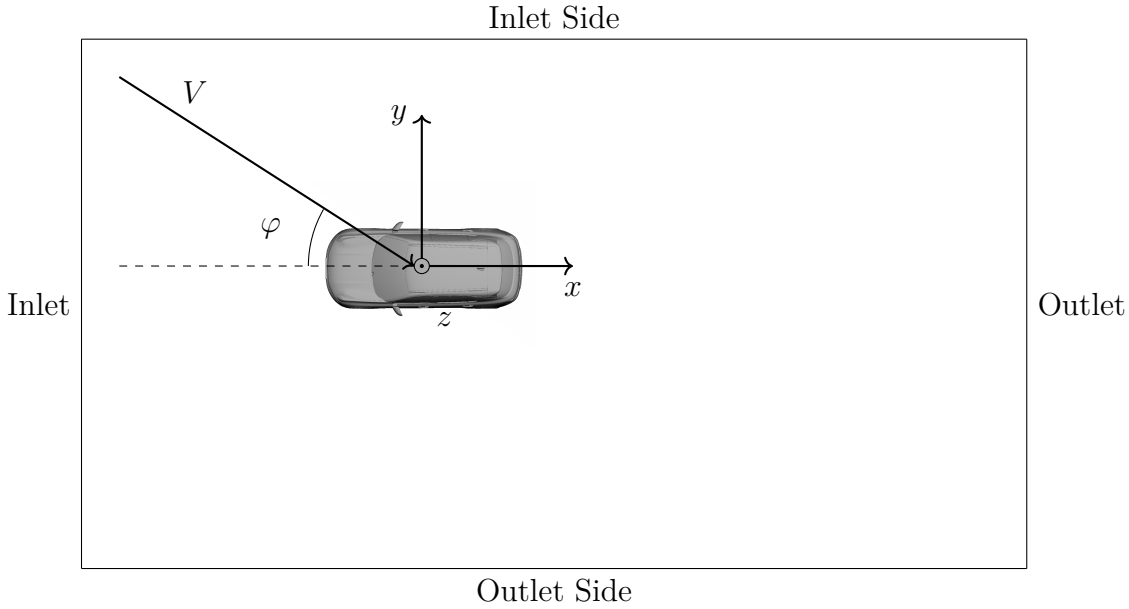


Figure 3.8: Definition of flow velocity and flow angle φ in a representation of the domain (not to scale).

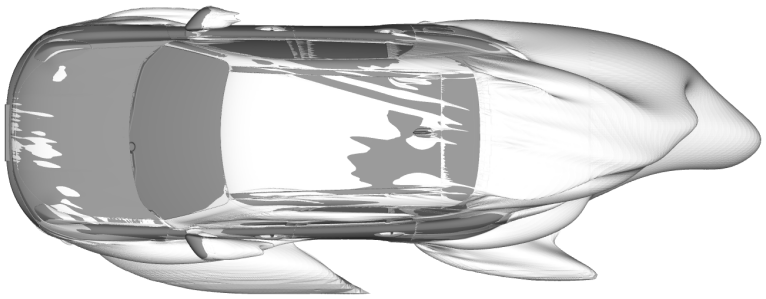


Figure 3.9: Wake regions from a simulation on the unmodified vehicle showing an iso-surface where the total pressure is zero.

The 120 CFD simulations were run on a Linux cluster with 432 nodes, where each run, including meshing, took less than four hours. To account for fluctuations, the

calculated forces and moments were averaged over the last 500 iterations. The data set was then compiled with the design parameters and corresponding results from the CFD simulations.

3.2 Neural Network

The neural networks act as surrogate models for the CFD solver in this optimisation process. With a limited set of simulated training data, the hyperparameters need to be precisely tuned for the network to be able to accurately predict the aerodynamic response to a novel set of design parameters. Three networks were constructed to each calculate an aerodynamic coefficient, specifically C_D, C_{LR} and C_{YM}. The networks were built in Python using PyTorch, an open source machine learning framework developed by Facebook's AI Research lab.

3.2.1 Process Data

To make the learning of the network more flexible and to keep the learning rate for the different input values similar, the values of the design parameters were normalised with respect to each parameter. Only three of the six aerodynamic coefficients were used as targets for the neural networks. The drag and rear lift, i.e. C_D and C_{LR} both have set limits the coefficients cannot exceed, which have to be considered when performing design changes. C_{LR} is also interesting for stability. The yaw moment, C_{YM}, was selected since it can act as a proxy for perceived instability during strong crosswinds [4]. This means that three networks, each with five inputs and one output, were created. From the original 120 simulations, 117 design were kept for the data set. The three removed cases had coefficients orders of magnitude from the rest of the numerical result, which indicates problems with convergence in the CFD simulation. From these 117 simulations, 80 % was selected for the training set and the remaining 20 % was the validation set.

3.2.2 Hyperparameters

To select the parameters the network is unable to learn by itself, the Hyperband algorithm was used. This algorithm was used to find a combination of parameters to minimise the loss in the validation set. Five parameters were evaluated: number of layer, number of neurons in each layer, activation function, learning rate and regularisation. This was implemented using the Hypersearch library which connects to the PyTorch. The input parameters for the Hyperband were set to 1 000 iterations to maximally train the network for and to discard 37.5 % of the configurations each round. Table 3.2 presents the search scope for each hyperparameter. A uniform distribution was used to sample from any hyperparameter expressed as an interval.

Table 3.2: Evaluated hyperparameters with corresponding search scope.

Hyperparameter	Tested
Number of layers	[1,4]
Neurons Hidden Layer	[5,200]
Activation Function	Sigmoid, Tanh
L_2 Regularisation Hidden Layer	[1e-8,1e-5]
Learning Rate	[1e-3,8e-3]

3.2.3 Train Network

The three forward-feeding fully connected networks were trained for 1 000 iterations using a RMS loss function. Between each layer was a L_2 regularisation layer to counteract the effects of overfitting. The training was integrated into the Hypersearch Python library.

3.2.4 Network Validation

To investigate the reliability of the surrogate model, the predictions on the validation set can be compared to the observed values. The difference between the models output and the target value is known as the residual (not to be confused with the residuals of a CFD simulation). By studying the residuals from the networks, the validity of the model can be assessed. To make residuals comparable for the different forces, they can be standardised by dividing each value with the standard deviation. It should be noted that it would be better to divide the data set into three sections: training, validation and test set, and perform these tests on the test set. However, due to the small number of data points, no test set was created.

3.3 Optimal Shape

With a complete surrogate model for the CFD solver in the form of three neural networks, the task was to find the optimal combination of design parameters to minimise the yaw moment, while maintaining the lift and drag forces within the limits. A genetic algorithm was used to search within the design space.

3.3.1 Genetic Algorithm

The algorithm was implemented in Python with the number of generations and the number of individuals in each generation as inputs. The output was the most fit individual, meaning the combination of inputs with the best fitness score. The fitness was calculated as the actual yaw-moment or rear lift force, which was set to a very large number if the drag and lift was not within the limits. The individuals with the lowest fitness score, defined as the actual yaw moment or rear lift intended to be minimised, were selected for crossbreeding. A fitness score combining C_{YM} and C_{LR} was also used to find the Pareto front.

A random population, each individual with five design parameters set between 0 and 1, was generated to initiate the algorithm. Each generation consisted of 1 000 individuals with 600 of them crossbreeding each iteration. Random mutation was added to 100 of the crossbred individuals for one of the five design parameters. The random mutation was done by randomly adding a value between 0 and 0.5 if the selected parameter was larger that 0.5, and subtract the same value if the parameter was smaller than 0.5. The algorithm was run for 10 000 generations.

4

Results and Discussion

With the aim of combining the fields of supervised learning and aerodynamics, a method was devised with the purpose of investigating design changes and their affect on crosswind sensitivity. This method, as well as the results it produces, are presented and discussed.

4.1 Neural Network Layout and Hyperparameters

Two of the three networks built ended up with two hidden layers, which could be argued is not deep learning, but it is sufficient to model any function. The third network, for predicting C_{YM}, had three layers. All networks had the sigmoid activation function as the best choice. Regularisation was also utilised in each layer. The number of layers, the learning rates as well as the final validation loss of the networks after training are all shown in Table 4.1. The validity of the networks is discussed in the next section.

Table 4.1: Evaluated hyperparameters with corresponding search scope.

Hyperparameter	C_D	C_{LR}	C_{YM}
Neurons Hidden Layer 1	16	85	46
Neurons Hidden Layer 2	67	44	27
Neurons Hidden Layer 3	-	-	74
Learning Rate	0.00568	0.00322	0.00649
Validation Loss	3.9085e-05	1.9635e-05	2.2865e-06

4.2 Neural Network Validity

The three networks are able to detect the trends in their respective data set.

4.2.1 Network for predicting C_DA

The most accurate predictions are generated by the network for C_D, though they are worse than results from previous studies [6]. However, that study was done on an aircraft that was not subjected to side wind. An aircraft is in most regions an aerodynamic body without big wake regions. This study on the other hand deals with a bluff body with large wake regions and diffuse areas of separation. To make things even harder, it deals with side wind. The complex wake regions are illustrated in Figure 3.9.

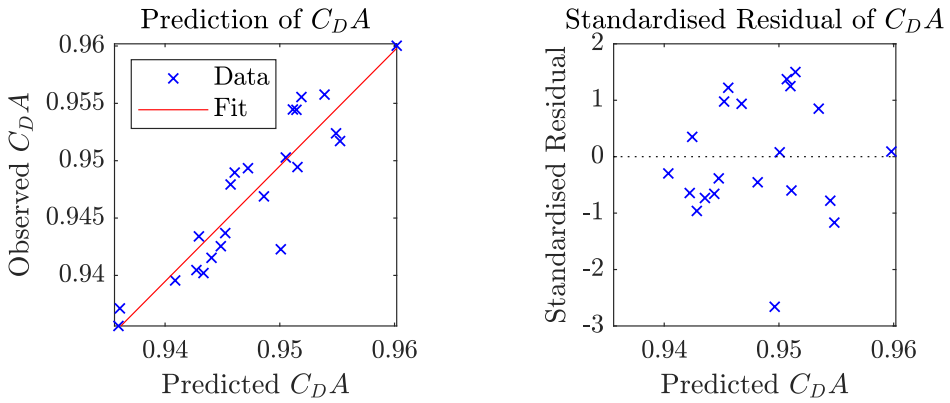


Figure 4.1: Evaluation of network performance on validation set for C_DA.

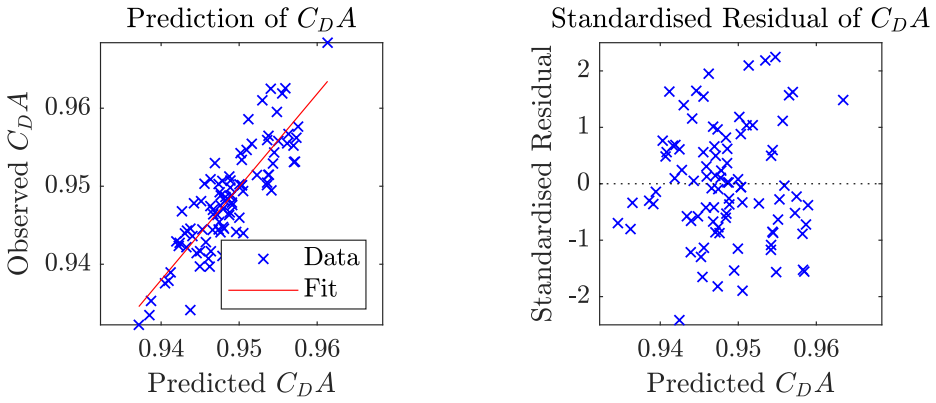


Figure 4.2: Evaluation of network performance on training set for C_DA.

The relationship between the observed and the predicted values are shown in Figure 4.1 with the linear regression plotted in red. The trends between the targets and the predicted values are very similar, and while a root mean square error of 0.00284 is still low, it is perhaps not useful for telling the exact value of C_DA. It could nonetheless be used for telling if a design change will increase, or decrease, the aerodynamic drag, which was its purpose. The residuals, shown in Figure 4.1, for the validation set show that there is a possible outlier, but otherwise the residuals are evenly distributed, indicating no prominent bias in the model. To ensure that the trends are not only present in the validation set, the prediction performance is compared to that of the training set, which is shown in Figure 4.2. The same trend is clearly present, but with a slight overestimation for lower values and underestimation

for larger values of C_DA, further establishing the model for C_DA as a reliable proxy for the CFD simulations. From Figure 4.2, the spread is also similar to that of the validation set.

4.2.2 Network for predicting C_{LR}

When calculating the rear lift forces, the predicted values, in comparison to the observed, show an underestimation at low target values and an overestimation at larger values. This means that there is a trend, but precise values will not be obtained. This behaviour is displayed for both the training and validation sets in Figures 4.3 and 4.4. It is also clear that the validation set contained fewer values of high C_{LR}. From the residuals, shown in Figures 4.3 and 4.4, no clear bias in the residuals are present. The root mean square error for the validation set is 0.00406.

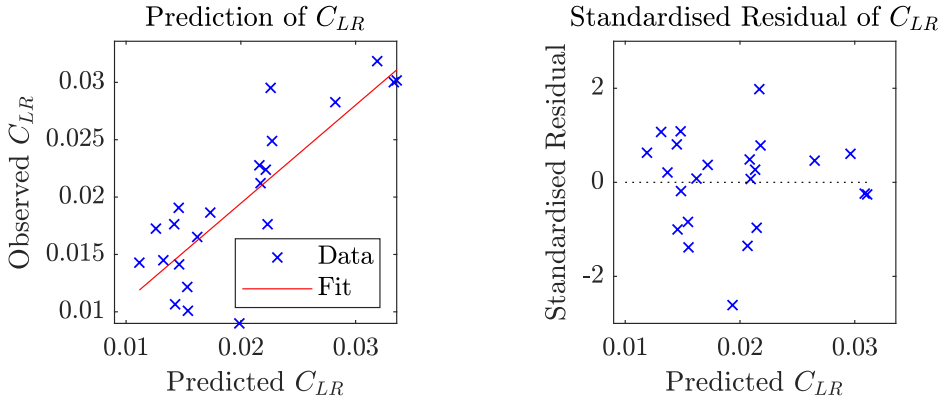


Figure 4.3: Evaluation of network performance on validation set for C_{LR}.

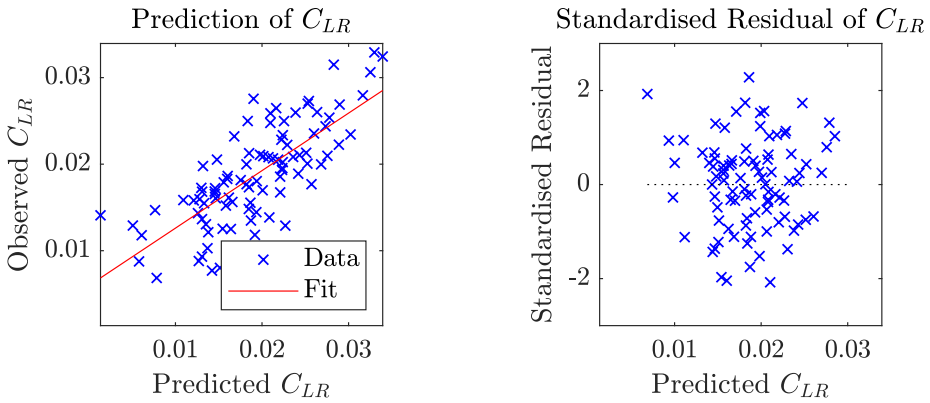


Figure 4.4: Evaluation of network performance on training set for C_{LR}.

4.2.3 Network for predicting C_{YM}

The poorest prediction results belong to an important aerodynamic coefficient for driving stability and, thus, for this thesis: C_{YM}. In the training set, there is a clear overestimation of low values, and an underestimation for larger values, as shown in

Figures 4.5 and 4.6. Since the total span for possible values of C_{YM} is very small, the design changes could be too small for the yaw moment to change significantly. This would put the changes at the same order of magnitude as the errors for C_{YM} in the CFD simulations, meaning the the neural network tries to predict noise. The standardised residuals also show a large spread, as seen in Figure 4.5 and 4.6. Since there is a small discernible trend in the predictions, the model will still be used for finding the optimal configuration of design parameters, but the value of the suggested solution should be questioned.

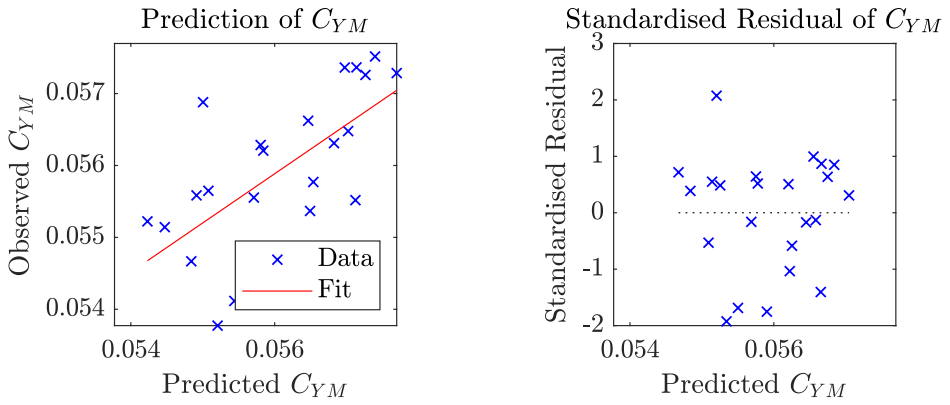


Figure 4.5: Evaluation of network performance on validation set for C_{YM}.

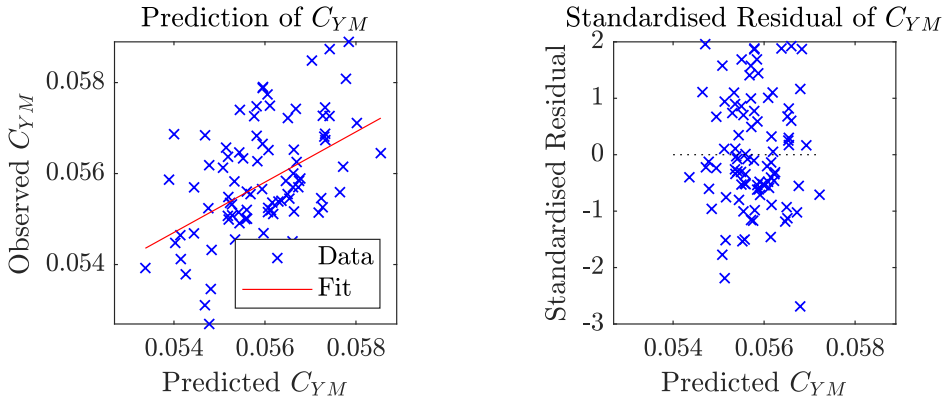


Figure 4.6: Evaluation of network performance on training set for C_{YM}.

4.3 Design Parameters

By evaluating how much each design parameter affect the three selected aerodynamic coefficients, guidelines for design changes can be created. The surrogate model is built from data generated with a 7.5° flow angle and a 100km/h head on wind speed, so generalisation to other setting should be done carefully. Most coefficients show a near linear relation to changes in the design parameters. While varying one parameter, the rest were kept at the baseline.

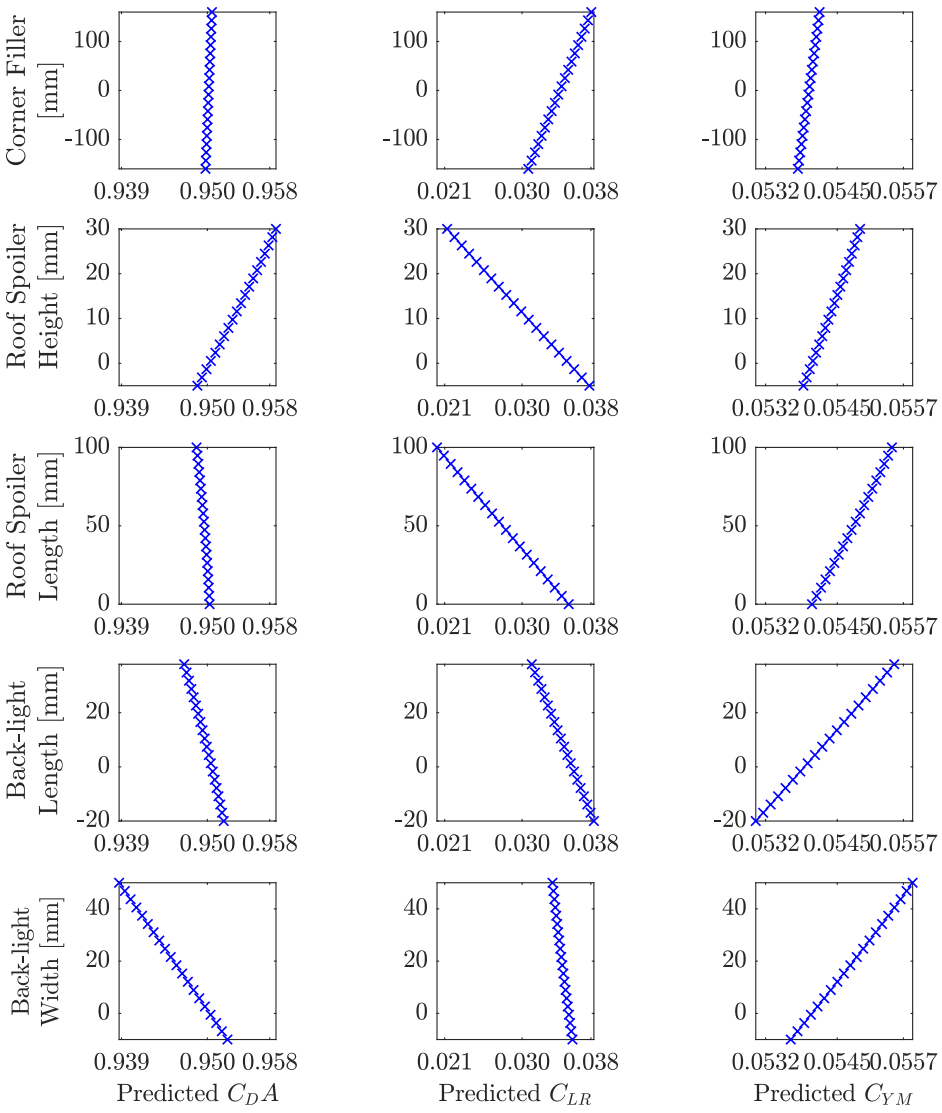


Figure 4.7: The relationships between the design parameters and each coefficient.

Studying the first column in Figure 4.7, the individual effects from the design parameters on C_DA can be evaluated. It is clear that the corner filler and length of the roof spoiler have the smallest effects. Increasing the length of the back-light decreases C_DA, but not by much. The two parameters with the largest effects are the height of the roof spoiler and the width of the back-light. The drag is increased with an increased height of the roof spoiler and the drag decreased when the back-light is made wider, possibly due to a clearer separation at the outer edge of the lamp.

In the middle column in Figure 4.7, the effects on rear lift force are found. Here the least effective parameter is the width of the back-light. The importance of the roof

spoiler, both in terms of length and height is evident. It is also evident that there has to be a trade-off for the height of the roof spoiler, since it greatly affects both C_DA and C_{LR}.

The final column shows how the yaw moment changes with the design parameters. Even with the most effective design parameter, namely the length of the back-light, little change in yaw moment is achieved. Interesting to note is that the yaw moment increases when all the design parameters are extended.

4.4 Optimal Design

Since there are distinguishable trends for the drag and rear lift, these networks could be used to indicate if a design change would increase or decrease C_D or C_{LR}. However, since the effects on C_{YM} from the selected design parameters were very small, results are less certain.

By investigating a grid of configurations of the design parameter, the combinations used to minimise the coefficients can be found. This accounts for the interaction of the design changes. It also highlights which parameters are important for each coefficient. The Pareto-front for minimising both C_{YM} and C_{LR} is shown in Figure 4.8. This hightlights the trade-off between these coefficients.

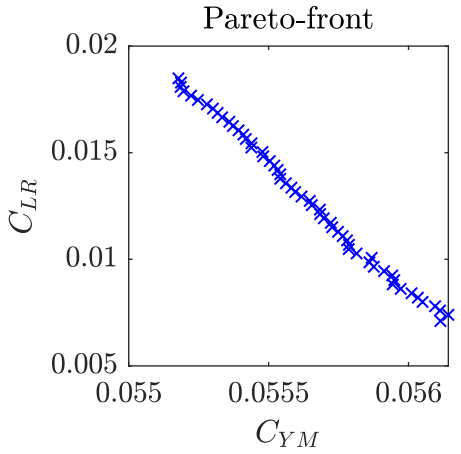


Figure 4.8: Pareto front showing the relationship between yaw moment and rear lift force.

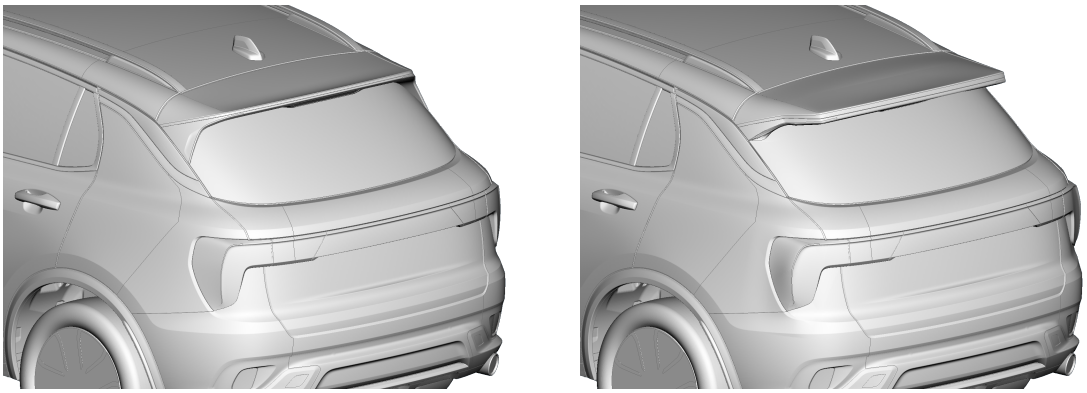
4.4.1 Minimising C_{LR}

Minimising the rear lift can be done by having a taller and longer roof spoiler, which was expected. The remaining three design parameters are less important for C_{LR}, but the general guidance is to decrease the corner filler and increase the length and width of the back-light. Using the genetic algorithm, the optimal configuration is shown in Table 4.2. This configuration accounts for the baseline values of both C_DA

and C_{YM}. The design parameters are in extreme positions for the length of the roof spoiler, as well as the corner filler. These settings results in a 0.0185 decrease in C_{LR}. The configuration is shown in Figure 4.9.

Table 4.2: Design parameters with optimal change from original position to minimise rear lift.

Design Parameter	Optimal Value [mm]
Length of roof spoiler	99.17
Height of roof spoiler	19.3
Length of back-light	-8.0
Width of back-light	23.9
Corner filler	-158.5



(a) Current design.

(b) Optimal design for minimising C_{LR}, while maintaining baseline levels of C_DA and C_{YM}.

Figure 4.9: Comparison between baseline design and the optimal design for minimising rear lift force.

4.4.2 Minimising C_{YM}

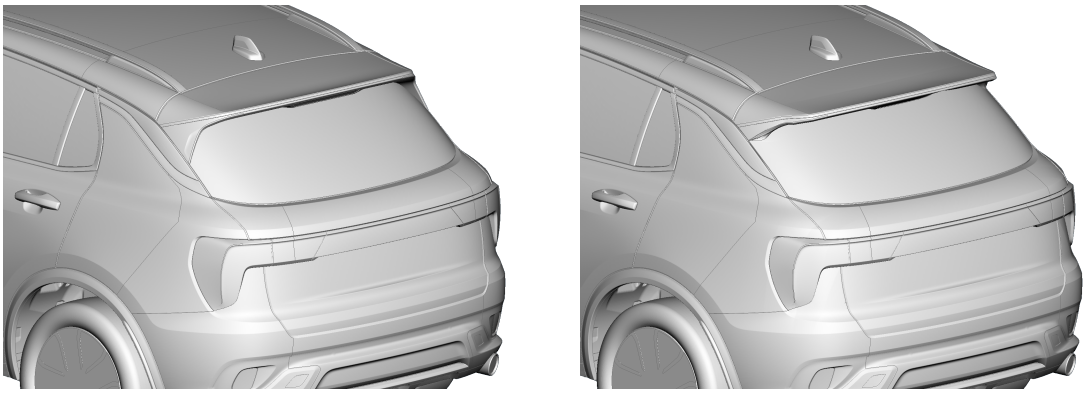
Reaching the lowest values of C_{YM} comes at the cost of both C_DA and C_{LR} and do not fulfil the requirements put on these coefficients. To minimise the yaw moment, the roof spoiler needs to be low and short. The back-light needs to have a short and narrow shape. Interesting to note is that the yaw moment seems to be independent from the corner filler parameter. This behaviour was also shown in Figure 4.7.

Using the genetic algorithm to search the design space, the suggested solution, to minimise the yaw moment while keeping the drag and rear lift forces below the baseline, is quite different from the one not considering the other coefficients. The exact suggestions are shown in Table 4.3, but the validity of these are questionable considering the poor performance of the neural network for predicting C_{YM}. The

solution presented by the genetic algorithm produces a 4.13% lower yaw moment. The optimal configuration is shown in Figure 4.10

Table 4.3: Design parameters with optimal change from original position to minimise yaw moment.

Design Parameter	Optimal Value [mm]
Length of roof spoiler	43.4
Height of roof spoiler	2.3
Length of back-light	-18.9
Width of back-light	6.8
Corner filler	-151.2



(a) Current design.

(b) Optimal design for minimising C_{YM}, while maintaining baseline levels of C_DA and C_{LR}.

Figure 4.10: Comparison between baseline design and the optimal design for minimising yaw moment.

4.4.3 Minimising C_DA

To minimise C_D, the surrogate model suggests maximally lowering the roof spoiler, and increasing its length, but the latter is of less importance. The same goes for the corner filler parameter, which has little effect on the drag. The back-light has a clear effect, and the design should extend the shape in both width and length. These results, when considering the interaction between all the parameters, correspond to the results from Figure 4.7, which only considers one design change at a time.

4.5 Ethics and Sustainability

The main concern while operating within the transport industry is the environmental impact a design change can have. By moving parts of the development process from physical tests to a virtual setting, possible errors could not only be detected earlier,

but also at a lower environmental cost. Furthermore, it turned out that the suggested best design to minimise the yaw moment was correlated to a larger drag force than the original design, which decrease the efficiency of the vehicle and, thus, impact the climate negatively. By putting a limit to these aerodynamic forces, a design can be determined which minimises yaw or rear lift force, but not at the cost of increasing drag or rear lift beyond what is acceptable.

5

Conclusion

This thesis combined the fields of deep learning and aerodynamics to investigate the difficult field of side wind simulations on a complex bluff body. The selected design parameters proved to have a small effect on the yaw moment at a 7.5° flow angle, which was reflected in the predictions from the neural network. Better prediction performances were received from the networks estimating the aerodynamic drag and rear lift forces, since the design changes had larger effects on these parameters. The genetic algorithm successfully identified the best configuration for minimising the predicted C_{LR} and C_{YM}. This thesis underscores the complexity of side wind simulations, and to fully investigate the stability of the car, transient fluid simulations need to be coupled to vehicle models, further adding complexity to the analysis.

For future work, the main area of interest would be to find design parameters associated with greater effects on the yaw moment. By investigating a larger set of design variables, and perhaps not only in the rear of the vehicle, more effective measures for improving the stability performance could be found. It would also be of interest to investigate how the size of the simulated data set affects the prediction performance. If a method could be devised for utilising existing simulations, without a rigid set of input variables, much could be learnt without the need to create new data. Lastly, this work has contributed to greater understanding of how to utilise neural networks as a key part of aerodynamic development. Most of the work, both for morphing the CAD and feeding the neural networks was automated by creating different scripts. The same methodology could easily be used for similar or other aerodynamic problems. It is believed that this way of working could be very beneficial within aerodynamic development of vehicles.

Bibliography

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[2] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[3] K. B. Singh and M. A. Arat, "Deep learning in the automotive industry: Recent advances and application examples," *arXiv preprint arXiv:1906.08834*, 2019.

[4] A. Brandt, S. Sebben, B. Jacobson, E. Preihs, and I. Johansson, "Quantitative high speed stability assessment of a sports utility vehicle and classification of wind gust profiles," tech. rep., SAE Technical Paper, 2020.

[5] J. Howell and G. Le Good, "The influence of aerodynamic lift on high speed stability," *SAE transactions*, pp. 1008–1015, 1999.

[6] R. Wallach, B. Mattos, R. Girardi, and M. Curvo, "Aerodynamic coefficient prediction of transport aircraft using neural network," in *44th AIAA Aerospace Sciences Meeting and Exhibit*, p. 658, 2006.

[7] K.-S. Song, S.-O. Kang, S.-O. Jun, H.-I. Park, J.-D. Kee, K.-H. Kim, and D.-H. Lee, "Aerodynamic design optimization of rear body shapes of a sedan for drag reduction," *International Journal of Automotive Technology*, vol. 13, no. 6, pp. 905–914, 2012.

[8] M. Larsson, P. De Raedt, M. Hedlund, *et al.*, *Aerodynamic identification using neural networks*. Linköping University Electronic Press, 1997.

[9] H. Chen, L. He, W. Qian, and S. Wang, "Multiple aerodynamic coefficient prediction of airfoils using a convolutional neural network," *Symmetry*, vol. 12, no. 4, p. 544, 2020.

[10] T. J. Santner, B. J. Williams, W. Notz, and B. J. Williams, *The design and*

analysis of computer experiments, vol. 1. Springer, 2003.

[11] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[12] R. H. Barnard, *Road vehicle aerodynamic design : an introduction*. MechAero, 2. ed. ed., 2001.

[13] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.

[14] T. Baer, "Understand, manage, and prevent algorithmic bias,"

[15] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.

[16] S. Gabriel, "Numerical simulation of vehicles under side wind load," Master's thesis, Royal Institute of Technology, 2004.