IX International Conference on Computational Methods in Marine Engineering
MARINE 2021
I.M. Viola and F. Brennan (Eds)

GEOMETRIC CONVOLUTIONAL NEURAL NETWORKS – A JOURNEY TO SURROGATE MODELLING OF MARITIME CFD

Asad Abbas[†], Ashkan Rafiee[†], Max Haase[†] and Andrew Malcolm[†]

[†]Technology Development Department, Austal
Perth, WA, Australia
{firstname}.{lastname}@austal.com
www.austal.com

Key words: Surrogate Modelling, Deep Learning, Geometrical Convolution, Computational Fluid Dynamics

Abstract. Computational Fluid Dynamics (CFD) has become an indispensable tool in the field of engineering design evaluation and optimisation. Existing numerical simulation methods are computationally expensive, memory demanding and time-consuming, thus limiting design space exploration and forbid generative design. In order to overcome these challenges, we propose a deep learning based surrogate modeling in-lieu of CFD simulations. Our proposed framework can predict flow fields (e.g pressure field) on the surface of the geometry as well as any overall scalar parameters (e.g drag force) given a three-dimensional shape input. It can also provide uncertainty quantification over predictions. Finally, we demonstrate that our proposed surrogate modelling does not require pre-processing of the input geometry and also outperforms state-of-the-art models in prediction accuracy. When comparing a dataset on aerodynamic drag of car geometries, we show that our model reduced the error standard deviation by a factor of ≈ 2.5 compared to a Gaussian Process-based surrogate model.

1 Introduction

Hydrodynamic hull form optimisation is of utmost importance during the vessel design to assure maximum fuel efficiency. Traditionally, hull form optimisations rely on domain experts to vary design parameters (such as hull length, shape, . . .) and evaluate their influence on given sets of objectives [1]. Such approaches are inherently very limiting in exploring the full design space and hence lower chance of finding the most optimum design. On the other hand, automated design optimisation is the process of allowing a computer algorithm to evolve an initial design to meet or exceed given sets of objectives. A notable example of a successful automated design optimisation is the 2006 NASA ST5 spacecraft antenna designed through evolutionary algorithms [2].

With advances in computing power and computational algorithms, hull form optimisation using evolutionary algorithms or other automatic optimisation algorithms has recently attracted many attentions [3, 4, 5, 6]. A comprehensive overview of geometry optimisation is presented by Harries [7]. However, the analysis of the flow field tends to be the most computationally intensive and time-consuming part of the optimisation process. This is due to complex non-linear hydrodynamics such as wave-making, transom stern ventilation,interaction effects at multihull ships and dynamic sinkage and trim of the vessel. In order to accurately capture such non-linear hydrodynamics, high fidelity Computational Fluid Dynamics (CFD) simulations are required. While CFD simulations can provide accurate performance predictions for a full-scale ship [8], they tend to be highly resource demanding. The drawback of computationally expensive objectives evaluations, hinders the applicability of using advance global optimisation algorithms for automatic design optimisation process.

A less computationally demanding alternative for objective function approximation is to use deep learning techniques. Deep learning architectures are highly suitable when dealing with high–dimensional data such as geometrical features. The use of deep learning based techniques to predict CFD simulations has recently attracted some attention. Guo *et. al* [9] used Convolutional Neural Network (CNN) architecture to predict steady laminar flow field around bluff bodies. The work is considered as pioneering the use of CNN for flow prediction but provides only qualitative comparison of the flow field and does not provide accurate comparison of drag and lift coefficients. Hennigh [10] proposed using U-Net based CNN architecture to predict the flow around bluff bodies. the proposed architecture was then used in a gradient based optimisation for reducing the drag to lift ratio of an airfoil. Umetani and Bickel [11] suggested the use of Gaussian process (GP) as surrogate model for CFD simulations. They have successfully used the approach for aerodynamic flow over car geometries. Their proposed approach showed excellent agreement with CFD simulations for both the pressure distribution over geometry surface and the resulting drag coefficients. Furthermore, they showed that through using GP surrogate model, their approach can also provide the uncertainty in the predicted drag coefficient. Baque *et. al* [12] proposed the use of Geodesic Convolutional Neural network as the surrogate model for CFD simulations. Similar to [11], they required PolyCube remeshing to pre-process the surface mesh. Raissi *et. al* [13] introduced the concept of physics–informed neural network to predict the lift and drag forces for a cylinder oscillating in a flow from scattered data in space and time. Lee and You [14] used Generative Adversial Networks along with convolutional neural networks to predict flow fields for unsteady flow over a circular cylinder in both space and time. In order to satisfy physical constraints, they introduced physics–informed terms in the loss function of networks. Sekar *et. al* [15] used a combination of convolutional neural network and classical Multilayer Perceptron (MLP) neural network to predict incompressible steady laminar flow field over airfoils. They used CNN architecture to parameterise the airfoil geometry while using the MLP network to predict the drag and lift coefficients. A comprehensive summary of applications of deep learning for fluid dynamics problems can be found at of [16].

Except for the work of and Umetani and Bickel [11] and Baque *et. al* [12], all the work in literature are limited to overall flow prediction in the computational domain through the use of CNN architectures. Although, such simulation can be informative in providing understanding of the flow field, they are not very useful in design process and optimisation where the input into surrogate model would need to be the 3D surface mesh representation of the geometry and the design engineers are interested on flow fields such as pressure or wall shear stress over the geometry surface in addition to the scalar quantities (*e.g* drag and lift coefficient,

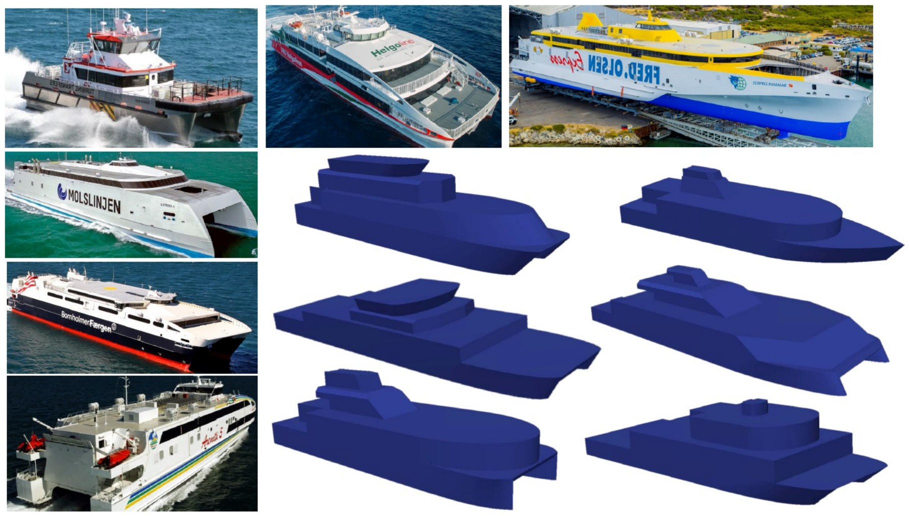


Figure 1: Examples of superstructures of existing ships and randomly generated geometries for training the surrogate model.

hydrodynamics resistance, or sinkage and trim of the vessel). Furthermore, the surrogate model should be capable of learning influence of fine geometrical features (such as surface curvatures, sharp corner, ...). To overcome this limitation, we propose a surrogate model architecture based on Geometrical Convolutional Neural Network (GCNN). The proposed GCNN architecture can learn the geometrical features from the 3D surface mesh input directly and predicts both flow fields and scalar quantities. Furthermore, we illustrate how the uncertainty can be built into the network to also provide uncertainty of scalar predictions.

To investigate the capabilities of a surrogate model for maritime fluid dynamics, the aerodynamic flow around ship superstructures was considered for this study. When compared to transient hydrodynamic flows around ship hulls, solutions for aerodynamic flows can be obtained in steady-state simulations that are more resource-efficient and ship superstructures provide a greater geometric variability which aids the development of an appropriate surrogate model.

2 Datasets

2.1 Ship Dataset

We generated 700 random ship geometries representing typical hull and superstructures shapes for high speed vessels (representing geometric features of the current Austal fleet) for training and testing of surrogate models. Each individual ship consists of an 80m long hull with three decks of random height, beam and length. Length–to–beam and length–to–height ratios were chosen to reflect typical values for those craft. For each geometry, the hull was chosen from 5 basic hull shapes and each deck from 8 different basic deck shapes. Figure 1 shows some examples of existing ship superstructures and of randomly generated geometries. The flow around each ship in head wind was simulated using the OpenFOAM's steady-state solver *simpleFoam* [17] with the semi-implicit method for pressure linked equations. The computational domain consisted of $\approx$ 700k cells and the ship structure had refinements along the feature edges.

2.2 Car Dataset

Car dataset was prepared by Umetani and Bickel [11] using shapes taken from the 'car' category of ShapeNet [18]. It contains 889 pairs of 3D input shapes, and output CFD simulation data consisting of drag coefficient (C_d) and pressure fields values.

3 Methodology

Our goal was to train a regression model to predict flow fields vectors (e.g pressure field) and scalar parameters (e.g drag force) given a three-dimensional shape input. Let $\mathcal{D} = (\mathbf{M}_m, Z_m, \mathbf{Y}_m)$ be the labelled training dataset, where $m = 1, \ldots, N$, and is the m^{th} training sample, and N denotes total number of training samples. $\mathbf{M}_m$ represents three-dimensional shape defined by $\mathcal{V}, \mathcal{E}$ and $\mathcal{F}$ denoting sets of vertices, edges and faces. Moreover, $Z_m \in \mathbb{R}$ represents global scalar parameter and $\mathbf{Y}_m \in \mathbb{R}^N$, represents vector of local pressure fields obtained by running a CFD simulation. We trained a neural network regression model defined by a decision function $F_\mathbf{w}$, where $\mathbf{w}$ denotes its trainable parameters. The decision function $(F_\mathbf{w}(\mathbf{M}_m))$ was used to predict pressure fields $\hat{\mathbf{Y}}_m$ and global scalar drag $\hat{Z}_m$ given an input mesh $\mathbf{M}_m$. Let $\mathcal{L}(Z_m, \mathbf{Y}_m, F_\mathbf{w}(\mathbf{M}_m))$ represents the loss function which determines the cost between a given ground truth labels $(Z_m, \mathbf{Y}_m)$, and the estimated label $(\hat{Z}_m, \hat{\mathbf{Y}}_m)$ and is given by:

$$\mathcal{L}(\mathbf{w}) = \sum_{m=1}^{N} \left\| \hat{\mathbf{Y}}_m - \mathbf{Y}_m \right\|^2 + \lambda(\hat{Z}_m - Z_m)^2 \tag{1}$$

where λ is a scaling parameter that ensures that both terms have approximately the same magnitude.

3.1 Introducing Uncertainty

For the regression model, the output $F_\mathbf{w}$ is $\mathbb{R}$ and the probability $P(F_\mathbf{w}|\mathbf{M}_m, \mathbf{w})$ is a Gaussian distribution. Inputs $m \subset \mathcal{M}$ were mapped onto the parameters of a distribution on $F_\mathbf{w}$ by several successive layers of linear transformation defined by $\mathbf{w}$. The weights $\mathbf{w}$ of these layers can be learnt by maximum likelihood estimation (MLE): given a set of training examples $\mathcal{D} = (\mathbf{M}_m, Z_m, \mathbf{Y}_m)$, $\mathbf{w}^{\mathrm{MLE}}$ were given by:

$$\begin{aligned} \mathbf{w}^{\mathrm{MLE}} &= \arg\max_{\mathbf{w}} \log P(\mathcal{D} \mid \mathbf{w}) \\ &= \arg\max_{\mathbf{w}} \sum_i \log P(\mathbf{y}_i \mid \mathbf{x}_i, \mathbf{w}) \end{aligned}$$

With a unit Gaussian prior, this yields L2 regularisation — commonly referred to as weight decay. Overall, however, point estimates of the network weights were obtained, without any knowledge of the underlying uncertainty.

In order to introduce uncertainty into our prediction, weights $\mathbf{w}$ in a neural network were represented by probability distributions over possible values, rather than having a single point estimate. Therefore, it effectively meant training ensemble of an uncountable infinite number of neural networks, where weights $\mathbf{w}$ in each individual network are drawn from a shared, learnt probability distribution. Bayesian inference for neural network allows to capture epistemic uncertainty ("knowledge uncertainty") by estimating the posterior distribution of the weights given the training data.

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})} = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{\int P(\mathcal{D}|\mathbf{w})P(\mathbf{w})d\mathbf{w}}$$

The posterior $P(\mathbf{w}|\mathcal{D})$ is not tractable and thus the solution is subsequently obtained using variational approximation to the Bayesian posterior distribution on the weights $\mathbf{w}$ of a neural network proposed by Hinton and Camp [19] and Graves [20]. Variational learning finds the parameters θ of a distribution on the weights $q(\mathbf{w};\theta)$ that minimises the Kullback-Leibler (KL) divergence with the true Bayesian posterior on the weights.

$$\begin{aligned}\theta^\star &= \arg\min_\theta \mathrm{KL}[q(\mathbf{w} \mid \theta)||P(\mathbf{w} \mid \mathcal{D})] \\ &= \arg\min_\theta \int q(\mathbf{w} \mid \theta) \log \frac{q(\mathbf{w} \mid \theta)}{P(\mathbf{w})P(\mathcal{D} \mid \mathbf{w})} \mathrm{d}\mathbf{w} \\ &= \arg\min_\theta \mathrm{KL}[q(\mathbf{w} \mid \theta)\|P(\mathbf{w})] - \mathbb{E}_{q(\mathbf{w}|\theta)}[\log P(\mathcal{D} \mid \mathbf{w})]\end{aligned}$$

This resulting term is alternately referred to as (negative) variational free energy [21, 22, 23] or the expected lower bound [21, 24, 25]. For simplicity we can write it as:

$$\mathcal{F}(\mathcal{D},\theta) = \mathrm{KL}[q(\mathbf{w} \mid \theta)\|P(\mathbf{w})] - \mathbb{E}_{q(\mathbf{w}|\theta)}[\log P(\mathcal{D} \mid \mathbf{w})] \qquad (2)$$

Since the model evidence is constant, maximising $\mathcal{F}$ minimises the KL divergence. We used 'Bayes by Backprop' [26] algorithm to minimise compression cost in Equation 2

3.2 Network Architecture

Our network architecture consists of two main building blocks namely the PointNet block and Geometric Convolution block as shown in the Figure 2. PointNet++ architecture was originally proposed by Qi *et. al* [27] and introduces a hierarchical structure allowing the network to capture features at different scales. This allows the network to efficiently encode global as well as local 3D shape features. However, one of the challenges that arise from the hierarchical structure is the resulting computational and memory cost, which inhibits its application to large point cloud dataset. To overcome this challenge we used a lightweight PointNet architecture with only two set abstraction levels. This allows the PointNet block to encode global features but limits its efficacy at encoding local structures. The second geometric convolution (GCNN) block overcomes this limitation and allows the network to learn local 3D shape features from extra information available in the input mesh. Compared with point cloud representations used in PointNet block, mesh-based representations also contain connectivity between neighbouring points, so they are more efficient for learning local regions on surfaces. Moreover, extra point-wise feature information extracted form input mesh by PointNet block allows GCNN to learn much more effectively on large meshes.

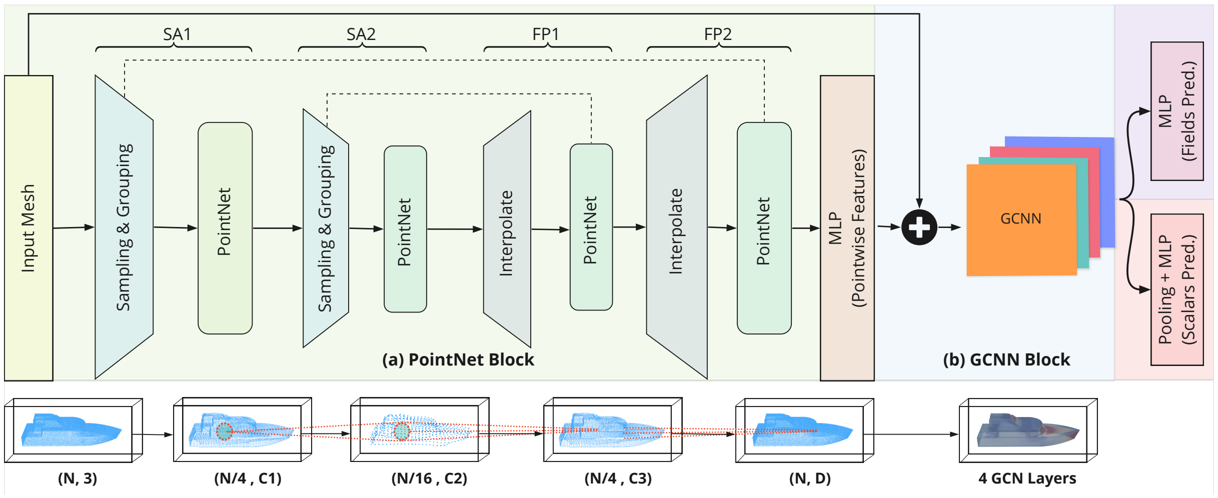


Figure 2: Proposed Network Architecture:

3.3 Experimentation

3.3.1 PointNet Block

The input to the PointNet block is set of 3D points $\{P_i \mid i = 1, \ldots, N\}$, where each point P_i is a vector of its (x, y, z) coordinate. These N points are fed into a hierarchical PointNet block which contains two Set Abstraction (SA) levels (SA1 and SA2) and two Feature Propagation (FP1 and FP2) levels. FP module uses conventional skip link concatenation, which is used for combining shallow features of the SA module with the corresponding deep features of the FP module. The first two set abstraction levels (SA1 and SA2) group input points into $N1 = 512$ and $N2 = 128$ local regions, respectively. In sampling and grouping phase we use farthest point sampling method to sample centroids of local regions and ball query to group points. The radius for ball query is set as 0.2 at the first set abstraction level and 0.4 at the second set abstraction level. Each local region contains $k = 64$ points. These two levels extract $C1 = 128$ and $C2 = 256$ dimensional features for each local region, respectively. The output of the last SA module is passed through successive FP modules. The FP module contains skip connections which helps the network to learn geometric features at different scales. Fully connected layers are used on top of the FP module to extract $D = 8$ dimensions feature vector for each point $P_i = (x, y, z, D)$.

3.3.2 Geometric Convolution Neural Network Block

The input to Geometric Convolution Neural Network (GCNN) block is Mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ and extra $D = 8$ dimensional feature channel extracted by PointNet block. GCNN block consists of a Spline-CNN architecture [28] with 4 convolutional layers: $\text{SConv}(1, 16) \rightarrow \text{SConv}(16, 32) \rightarrow 2 \times \text{SConv}(32, 64)$ with a kernel size of 5. Finally, to get field and scalar predictions, a regression head consisting of MLP layers (64, 32) and Max-pooling + MLP layer (32, 1) is used respectively. We used ReLU activation function after each SConv.

3.3.3 Hyperparameters

For training end-to-end network, we use Adam optimiser [29] with initial learning rate of 3×10^{-3} to optimise over ELBO (Eq. 2) with L_1 loss. Learning rate was reduced on reaching plateau by a factor of 0.1. For training, a batch size of 1 and early stopping with a patience of 20 epochs was used. In order to minimuse loss function (Eq. 1), we set $\lambda = 100$ to control the relative importance of the two terms, so that both terms have approximately the same magnitude. Moreover, to force the network to learn geometric features and reduce overfitting on scalar predictions, we dynamically adjusted λ during the training phase to give more weight to surface pressure prediction initially. For weight uncertainity we used prior $P(\mathbf{w})$ consisting of a scale mixture of two Gaussian densities. Both having zero mean, but differing variances:

$$P(\mathbf{w}) = \prod_j \pi \mathcal{N}\left(\mathbf{w}_j \mid 0, \sigma_1^2\right) + (1-\pi)\mathcal{N}\left(\mathbf{w}_j \mid 0, \sigma_2^2\right)$$

where $\mathbf{w}_j$ is the jth weight of the network, $\mathcal{N}(x|\mu, \sigma^2)$ is the Gaussian density evaluated at x with mean μ and variance σ_1^2 and σ_2^2 are the variances of the mixture components. For Bayes by Backprop, we averaged over 3 samples and considered scale mixture $\pi = 4$, prior $\sigma_1 = 1$ and prior $\sigma_2 = 4$ on the mixture prior distribution 1 and 2 respectively. We implemented our deep-learning algorithms using PyTorch and used Tesla V100 GPUs for training.

4 Results and Discussion

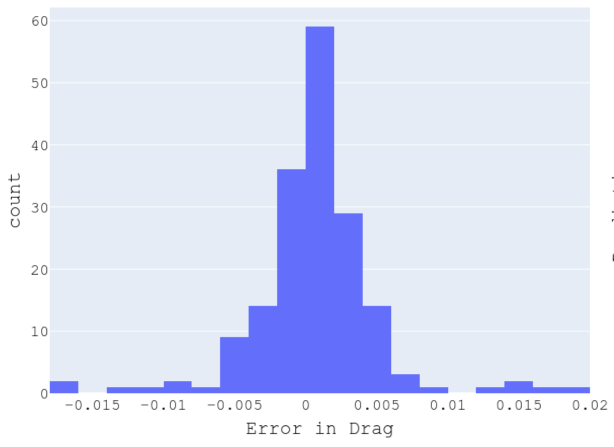
In this section, we evaluate our proposed approach for predicting pressure fields and associated scalar values for a given input shape. We quantify the performance of our proposed model for predicting pressure fields and scalars using Mean Absolute Error (MAE) and R^2 (coefficient of determination) regression score respectively.

4.1 Car Dataset

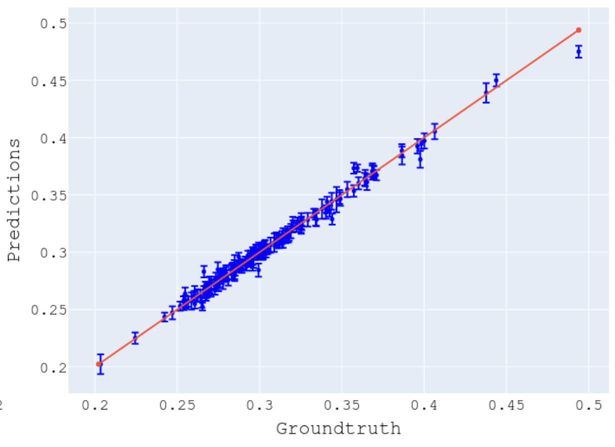
To generate our training and testing data, we split the car dataset in a $90 : 10$ ratio. We compare our results of drag coefficient prediction with a state-of-the-art PolyCube maps-based parametrisation approach presented by Umetani and Bickel [11]. Compared to their regression approach, our presented method achieves a higher accuracy in predicting drag coefficient $\mathcal{C}_d$ value with low uncertainty. As shown in Figure 3, standard deviation of the error is about ± 0.0046 compared to ± 0.012 value of previous state-of-the-art method. Our model achieves an excellent $R^2 > 0.98$ on both training and test dataset. Our model achieves a low MAE of 4.0 and 4.2 on the train and test datasets respectively. Figure 4 shows qualitative results for surface pressure prediction. It can be seen that our proposed model performs well in predicting pressure on the surface of the car.

4.2 Ship dataset

To generate our training and testing data, we split the ship dataset in a $90 : 10$ ratio. When Compared to the car dataset, the ship dataset is more complex and has varying number of vertices for different shapes. It has on average $70k$ vertices per ship, which is $\sim 20\times$ more than car dataset (3682 vertices for each car). Moreover, it contains force coefficients in both x and



(a) Distribution of the magnitude of errors



(b) R-squared plot with uncertainty

Figure 3: (a) Error distribution has $\mu = 0.0006$ and $\sigma = 0.004$. (b) R-squared for the proposed regression model. Error bars shows Epistemic uncertainty (i.e. uncertainty due to limited data and knowledge).

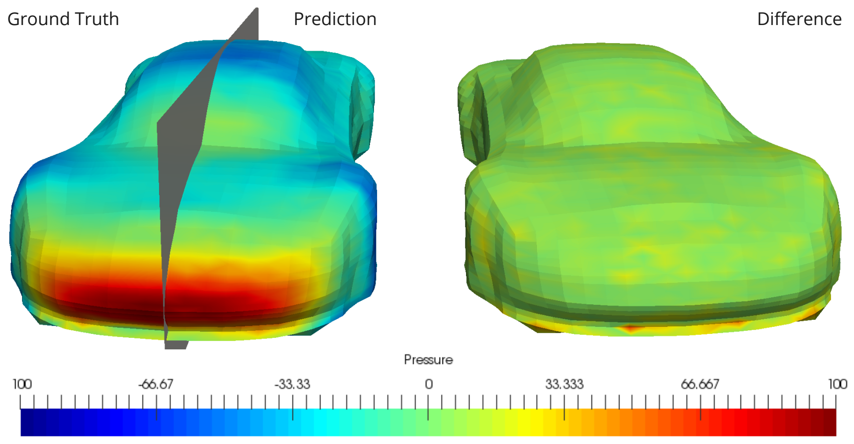


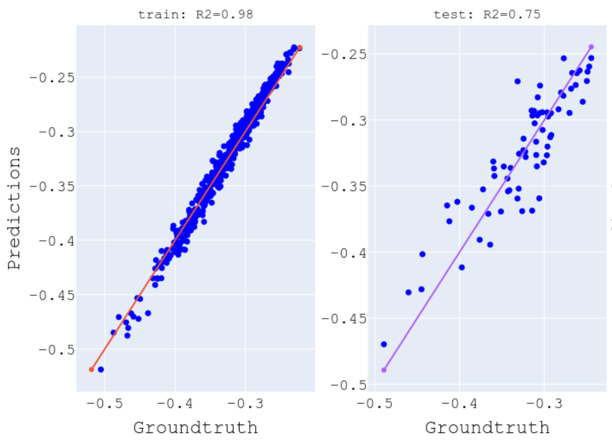
Figure 4: An example result from the car test data. (Left) Qualitative comparison of reference ground truth and predicted surface pressure values. (Right) Difference between ground truth and predicted surface pressure values.

y directions. Because of the size and complexity of ship dataset, we conduct an ablation study and compare the performance of our proposed model to a couple of baselines. This allows us to establish the efficacy of our proposed approach.

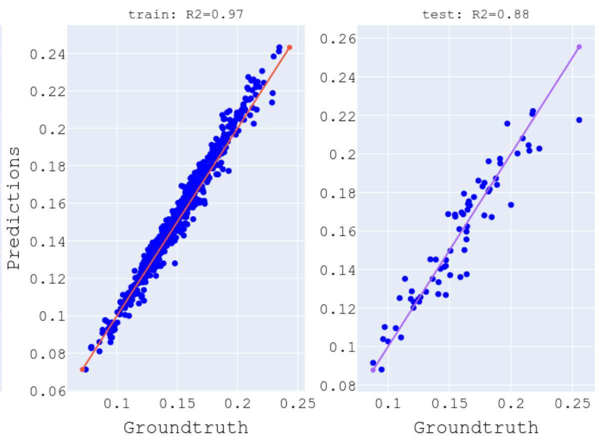
GCNN Model: We only used the GCNN block to learn surface pressure predictions.

PointNet Model: In this case, only the PointNet block is used to learn surface pressure predictions.

PointNet+GCNN: We train our proposed architecture and train it in an end-to-end manner to predict both surface pressure predictions and drag coefficients. Because of the memory limitations, we were unable to implement 'Bayes by Backprop' [26] and introduce uncertainty in our predictions for this application.



(a) R-square plot for x-drag



(b) R-square plot for y-drag

Figure 5: R-squared plots for the proposed regression model

Table 1: Comparison of MAE and MPE of different architectures for pressure field prediction on ship dataset. The present architecture is a combination of PointNet and GCNN.

Models	Train		Test		Training time	
	MAE	MPE	MAE	MPE	sec/epochs	epochs
GCNN	4.54	3.28	4.81	3.56	60	2000
PointNet	2.87	1.56	3.54	2.28	240	1000
present architecture	**1.27**	**0.56**	**2.74**	**1.37**	360	500

Table 1 shows comparative results for surface pressure prediction. Our proposed model achieves a substantially lower MAE and MPE (mean percentage error) for the surface pressure prediction. This confirms our hypothesis that learning extra point-wise features using PointNet block helped GCNN to converge. Figure 5 shows the R-squared score on the training and test dataset for x and y force coefficient. The model achieved a better R^2 score on the training dataset compared to the test dataset. We could reduce this overfitting by gathering more data in ranges where model has low confidence in predictions (i.e. having large error) or introducing strong regularisations during training (e.g. 'Bayes by Backprop' [26] or dropout [30]). Figure 6 shows a qualitative comparison of the ground truth pressure field and predicted pressure field. It can be seen that our proposed network learned to accurately predict the pressure field for this large complex input geometry as the difference between predicted and ground truth pressure field is small. Moreover, similar to Baque *et. al* [12], we observed that learning to predict surface pressure fields helped in improving scalar predictions for both, ship and car, datasets.

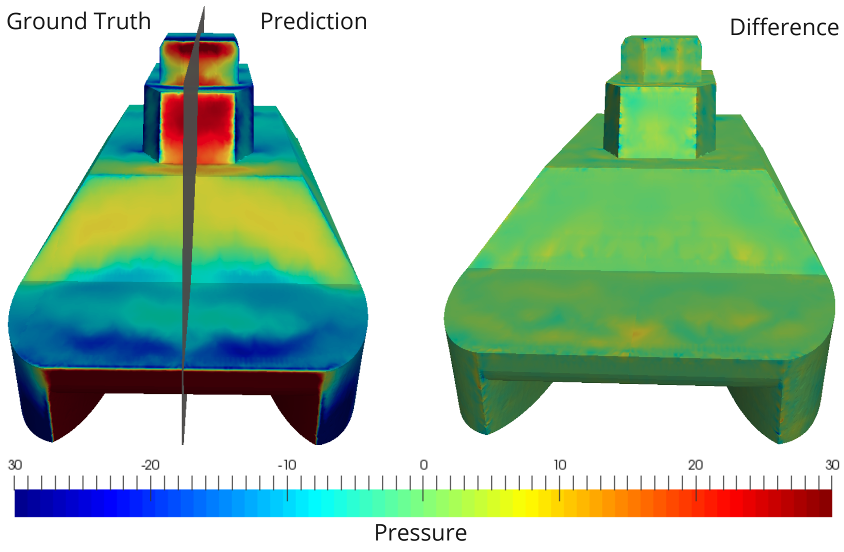


Figure 6: An example result from the ship test data. (Left) Qualitative comparison of reference ground truth and predicted surface pressure values. (Right) Difference between ground truth and predicted surface pressure values.

The error of the predicted scalar value when compared to the ground truth reached up to 4% and 6%, achieved for the y-force and x-force component, respectively. This compares to the uncertainties experienced in hydrodynamic ship scale-model testing that can vary from $2 - 6\%$ for fast monohull [31] or multihull vessels [32]. Hence the proposed method can be considered sufficiently accurate for maritime performance prediction purposes.

5 Conclusion

In this paper, we proposed a Geometric Convolutional Neural Network-based (GCNN) surrogate modelling approach to predict flow fields (e.g pressure field) on the surface of a geometry as well as scalars such as drag for a three-dimensional shape input. To use neural networks for surrogate modelling for flows around maritime structures, it is crucial to choose a data representation that encodes the geometrical properties of input data effectively. Our proposed approach benefited from both, mesh data (GCNN) and point cloud (PointNet) representations, to learn the underlying geometrical structure of the input data without introducing geometrical artefacts or compromising on vertex connectivity information. We also showed how to incorporate uncertainty estimations into predictions by using Bayesian learning.

We showed that our proposed surrogate model based on combining Pointnet and GCNN was able to predict the drag force on the car with a standard deviation of 0.0046, exceeding the accuracy of the state-of-the-art model having a standard deviation of 0.012. In case of wind resistance on ship, we showed that the proposed surrogate model achieved the lowest mean percentage error of 1.37 on the test dataset (compared to $2.28, 3.56$ for GCNN, PointNet).

The predicted values of the force acting on the ship deviated by $4-6\%$ when compared to the ground truth which correlates to the uncertainty of towing tank experiments. The prediction of force coefficients showed higher R^2 score on the training dataset when compared to the test dataset, indicating possibilities of over–fitting of the surrogate model. This is currently under investigation and improved results will be published in future work.

In future work we will focus on optimisation strategies using presented surrogate modeling. We aim to compare the optimised geometries using different surrogate models presented in Table 1 and ones derived from CFD simulations to quantify the accuracy required from surrogate models versus their computational cost when used for cost function evaluation in optimisation algorithms.

REFERENCES

[1] M. Haase, G. Davidson, S. Friezer, J. Binns, G. Thomas, and N. Bose, "Hydrodynamic hull form design space exploration of large medium-speed catamarans using full-scale CFD," *The International Journal of Maritime Engineering*, vol. 154, 2015.

[2] G. Hornby, A. Globus, D. Linden, and J. Lohn, "Automated antenna design with evolutionary algorithms," in *Space 2006*, 2006.

[3] M. Diez and D. Peri, "Global optimization algorithms for robust optimization in naval design," in 8th *Conference on Computer and IT Applications in the Maritime Industries*, 2009.

[4] ——, "Optimal hull-form design subject to epistemic uncertainty," in 10th *Conference on Computer and IT Applications in the Maritime Industries*, 2011.

[5] W. Luo and L. Lan, "Design optimization of the lines of the bulbous bow of a hull based on parametric modeling and computational fluid dynamics calculation," *Mathematical and Computational Applications*, vol. 22, 2017.

[6] R. Pellegrini, A. Serani, S. Ficini, R. Broglia, M. Diez, J. Wackers, and M. Visonneau, "Adapt, Adapt, Adapt: Recent Trends in Multi-fidelity Digital Modelling for Marine Engineering," in 19th *Conference on Computer and IT Applications in the Maritime Industries*, 2020.

[7] S. Harries, "Practical shape optimization using CFD: State–of–the–art in industry and selected trends," in 19th *Conference on Computer and IT Applications in the Maritime Industries*, 2020.

[8] M. Haase, K. Zurcher, G. Davidson, J. R. Binns, G. Thomas, and N. Bose, "Novel CFD-based full-scale resistance prediction for large medium-speed catamarans," *Ocean Engineering*, vol. 111, pp. 198–208, 2016.

[9] X. Guo, W. Li, and F. Iorio, "Convolutional neural networks for steady flow approximation," in 22nd *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, KDD '16*, 2016.

[10] O. Hennigh, “Lat-net: Compressing lattice boltzmann flow simulations using deep neural networks,” 2017.

[11] N. Umetani and B. Bickel, “Learning three-dimensional flow for interactive aerodynamic design,” *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–10, 2018.

[12] P. Baque, E. Remelli, F. Fleuret, and P. Fua, “Geodesic convolutional shape optimization,” in *International Conference on Machine Learning.* PMLR, 2018, pp. 472–481.

[13] M. Raissi, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis, “Deep learning of vortex-induced vibrations,” *Journal of Fluid Mechanics*, vol. 861, p. 119–137, Dec 2018. [Online]. Available: http://dx.doi.org/10.1017/jfm.2018.872

[14] S. Lee and D. You, “Data-driven prediction of unsteady flow over a circular cylinder using deep learning,” *Journal of Fluid Mechanics*, vol. 879, p. 217–254, Sep 2019. [Online]. Available: http://dx.doi.org/10.1017/jfm.2019.700

[15] V. Sekar, Q. Jiang, C. Shu, and B. C. Khoo, “Fast flow field prediction over airfoils using deep learning approach,” *Phys. Fluids*, vol. 31, 2019.

[16] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, “Machine learning for fluid mechanics,” *Annual Review of Fluid Mechanics*, vol. 52, no. 1, p. 477–508, Jan 2020. [Online]. Available: http://dx.doi.org/10.1146/annurev-fluid-010719-060214

[17] [Online]. Available: https://www.openfoam.com

[18] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, “Shapenet: An information-rich 3d model repository,” *arXiv preprint arXiv:1512.03012*, 2015.

[19] G. E. Hinton and D. Van Camp, “Keeping the neural networks simple by minimizing the description length of the weights,” in *Proceedings of the sixth annual conference on Computational learning theory*, 1993, pp. 5–13.

[20] A. Graves, “Practical variational inference for neural networks,” in *Advances in neural information processing systems.* Citeseer, 2011, pp. 2348–2356.

[21] R. M. Neal and G. E. Hinton, “A view of the em algorithm that justifies incremental, sparse, and other variants,” in *Learning in graphical models.* Springer, 1998, pp. 355–368.

[22] J. S. Yedidia, W. T. Freeman, Y. Weiss *et al.*, “Generalized belief propagation,” in *NIPS*, vol. 13, 2000, pp. 689–695.

[23] K. Friston, J. Mattout, N. Trujillo-Barreto, J. Ashburner, and W. Penny, “Variational free energy and the laplace approximation,” *Neuroimage*, vol. 34, no. 1, pp. 220–234, 2007.

[24] T. S. Jaakkola and M. I. Jordan, “Bayesian parameter estimation via variational methods,” *Statistics and Computing*, vol. 10, no. 1, pp. 25–37, 2000.

[25] L. K. Saul, T. Jaakkola, and M. I. Jordan, "Mean field theory for sigmoid belief networks," *Journal of artificial intelligence research*, vol. 4, pp. 61–76, 1996.

[26] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International Conference on Machine Learning.* PMLR, 2015, pp. 1613–1622.

[27] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the* 31st *International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 5105–5114.

[28] M. Fey, J. E. Lenssen, F. Weichert, and H. Müller, "Splinecnn: Fast geometric deep learning with continuous b-spline kernels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 869–877.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in 3rd *International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[31] J. Gorski, "The resistance committee – final report and recommendations to the 26th ITTC," in *Proceedings of the International Towing Tank Conference*, 2011.

[32] K. Zurcher, "Experimental waterjet investigations," Ph.D. dissertation, AMC,University of Tasmania, Australia, 2015.