School of Computation, Information and Technology

Technical University of Munich

Bachelor's Thesis in Informatics

Upsampling Computational Fluid Dynamics Simulation with Convolutional Neural Networks via preCICE

Egor Rakcheev

School of Computation, Information and Technology

Technical University of Munich

Bachelor's Thesis in Informatics

Upsampling Computational Fluid Dynamics Simulation with Convolutional Neural Networks via preCICE

Author:	Egor Rakcheev
Supervisor:	Dr. rer. nat. Felix Dietrich
Advisor:	Gerasimos Chourdakis, M.Sc.
Submission Date:	June 15th, 2023

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

June 15th, 2023 Egor Rakcheev

Abstract

Engineering design benefits from computational fluid dynamics simulations since they give detailed information on a system's or product's performance without the requirement to build and test physical prototypes. However, running them requires substantial computational power. Super-resolution methods have shown promising results in upsampling data from low-resolution simulations; however, less effort has been made in coupling these technologies efficiently. In this work, a system for coupling the SRCNN model with a two-dimensional fluid simulation using the preCICE coupling library is proposed. Results of experiments have found that this method creates more accurate upsampled simulations than regular interpolation methods without deep learning methods, although the limitations of higher scale factors and more complex meshes show a substantial decrease in accuracy. The coupling approach indicates the possibility to not only apply super resolution but also combine other simulation-improving techniques arbitrarily and rapidly.

Contents

1 Introduction

Recent years have seen a sharp increase in the amount of research being done on deep neural networks, which has even helped fields like generative neural networks gain widespread attention due to innovations like OpenAI's ChatGPT and other technological developments. Questions about which practical sectors this technology may help and enhance are raised as research on deep learning-based AI continues to spread into many application fields. Upsampling images to a greater resolution using convolutional neural networks is one application called image super-resolution. Although this technology is mostly applied to regular images taken by camera, drawn, or otherwise created to be observed by human eyes, it poses the question if it would be possible to use such techniques to upsample data from simulations beyond the limitations of regular interpolation methods. It has already been shown that it is feasible to do so with cosmological simulations [13], and further viable for fluid simulations as well. Some research has already shown promising results on training generative adversarial networks with temporal coherence [19] and physics informed networks [12] to perform super resolution on fluid simulations, however this research neglects the coupling possibilities between solvers and artificial neural networks.

This study aims to determine if upsampling of fluid simulations is possible using convolutional neural networks by coupling them to low-resolution simulations using preCICE and comparing upsampled simulations to conventionally calculated high-resolution simulations. A successful application of this method would reveal the possibility of saving on computation resources while performing detailed fluid simulation upsampling in parallel to the low resolution calculations, as super resolution requires much less time than calculating a full-scale simulation at the same resolution.

As mentioned previously, to help in this endeavor and possibly allow for real-time simulation and super-resolution combination, the coupling library preCICE is used. Although originally applied to couple various simulation solvers in space or time, it might be possible to treat the super resolution process as a kind of solver that performs the upsampling. This coupling would also show the capabilities of utilizing preCICE not only for solver-to-solver communication but for the connection of different kinds of technology with simulations as well.

To achieve the set goals, the first chapter 2 includes a brief introduction and summary of fluid simulations, super resolution technology using convolutional neural networks, and the preCICE library. Afterwards, the proposed theories are explained, set up, and finally tested through computational experiments in chapter chapter 3. The results of these experiments are concluded in chapter chapter 4.

2 Fluid Simulations and Upsampling Background

2.1 Fluid Simulation and Finite Element Method

2.1.1 Navier-Stokes

An accurate method of mathematically describing the physical properties of a fluid are the Navier-Stokes equations. In the formulation of the equations, the fluid is treated as a continuous material, represented by functions of space and time. They only describe the effect of local properties in such a material and are not a detailed representation of singular physical particles. The equations are a set of coupled partial differential equations that describe the relations of pressure, temperature, and density of a moving incompressible fluid. They are split into two parts: conservation of mass and conservation of momentum. Conservation of mass ensures that no mass is created or destroyed and that the difference between the inflow and outflow of mass in any given system is zero. Mathematically, it is expressed as

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{2.1}$$

describing $\mathbf{u}$ as the fluid velocity and ρ fluid density. The second equation represents conservation of momentum and is based on Newton's second law of motion, adapted for fluids. It encompasses inertial, pressure, viscous, and external forces acting on the fluid and expresses an incompressible three-dimensional flow as

$$\rho(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}) = -\nabla p + \nabla \cdot (\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I}) + \mathbf{F} \tag{2.2}$$

with $\mathbf{p}$ being fluid pressure, μ the viscosity, and $\mathbf{F}$ external forces, which are applied to the fluid [17]. Although these partial differential equations are mainly used in this thesis, there are other PDEs often used to describe fundamental physical properties including:

- The heat equation that models the diffusion of heat through a given region.
- The wave equation, that describes waves, whether as sound or water mechanical waves, or electromagnetic waves like light.

- The Schrödinger's equation, which describes the wave function in a quantum mechanical system and is used to find the allowed energy levels of quantum mechanical systems.

- Maxwell's equations, which are governing the behaviour of electromagnetism, used to describe generation of electric and magnetic fields by charges, currents and changing fields.

2.1.2 Computational Fluid Dynamics

To compute a PDE in a system, like the pressure and flow of a chosen continuum numerically, the problem is generally approached using the Finite Element Method (FEM). It can be used in various applications for the computation of more complex fluid, heat, and structural problems that cannot be solved analytically. The main approach of FEM is to first discretize the continuum into finite elements, connected by nodes, that represent the original system as a discrete mesh.

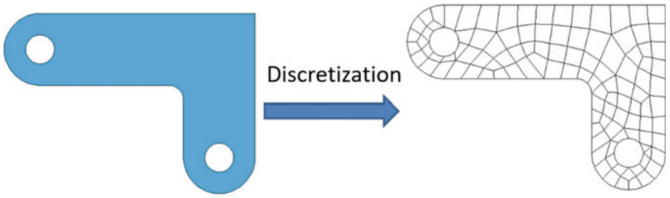

Figure 2.1: The discretization of a 2D object into a mesh [15].

There are two main discretization approaches for fluid simulations, namely the Eulerian and the Lagrangian approaches. The Eulerian approach discretizes the whole domain as a grid, on which, following the Navier-Stokes equations, the velocity values are gradually updated step-by-step. This approach enforces the incompressibility of fluids, preserving the volume of the fluid. It also allows for an accurate evaluation of spacial derivatives like gradient, divergence, and the Laplacian on account of the static grid points of the domain. An example of such a discretization of a 2-dimensional structure can be seen in Figure 2.1. The Lagrangian approach, in contrast, defines fluid volumes as a set of particles, or parcels, with stored physical quantities, or labels, and the simulation is forwarded by advecting these physical variables in accordance with the Navier-Stokes equations. Since the domain is not completely discretized as a static mesh in itself but rather only where the fluid exists, the particle calculations do not need to be performed where the fluid is absent, reducing memory usage and computational costs compared to the Eulerian approach. This, however, also makes ensuring fluid compressibility harder, as spacial derivatives need to be computed on irregularly moving particles, making the calculations more difficult and thus less precise [2].

After discretizing the system, the underlying PDEs are then approximated with a linear combination of basis functions at each discrete triangular area in the mesh. They are then combined into the initial boundary value problem, which consists of the partial differential equations themselves as well as the initial conditions and boundary conditions. Boundary conditions are the constraints in a domain on whose boundary a set of conditions is placed, and they can be set in multiple forms, whereas the initial values are conditions set in time-direction. The Dirichlet boundary condition prescribes set values for dependent variables within the system, while the Neumann boundary condition prescribes the derivatives of the dependent variables. These two are the most commonly encountered conditions in PDE solutions, together with the Robin boundary condition, which is a weighted combination of the Dirichlet and the Neumann conditions.

To perform a finite element analysis, the following five steps are performed:

1. Prepare the model. This includes creating the geometry of the model, defining the properties of the material, choosing the actual initial boundary conditions and other conditions of the system, and finally discretizing the model into a mesh of discrete points.

2. Formulate the partial differential equation in its weak form, in which certain conditions can be relaxed, simplifying the solving process.

3. Set up the global problem as a set of linear equations for each element.

4. Solve the linear equations. To obtain the values of the unknowns in the equation, direct methods, like Gaussian elimination, or iterative methods, like the conjugate gradient method, can be used.

5. Apply post-processing, like obtaining visualizations for the result and check if the calculated values correspond to ones from the analytical solution if it is available, or otherwise check the correctness and plausibility of the output.

The FEM method can be used for various types of multiphysics problems involving heat transfer, fluid dynamics, and electric fields, with a main focus on the three types of problems:

1. Static problems, which usually exclude motion, are used as structural analyses of structures like bridges or buildings to locate stress high- and low-points.

2. Dynamic problems, which describe situations that change over time, like heat transfer or fluid flow.

3. Modal problems, which simulate vibrational effects on a system.

More specialized and advanced FE methods have also been introduced over the years, including Extended FEM to analyze systems with discontinuities like crack propagation and fractures and Generalized FEM, which combines regular FEM with meshless methods. [15]. Another possibility for computing numerical solutions for PDEs is the finite volume method. Contrary to FEM, this method splits the surface into finite volumes, as the name suggests, and evaluates average values for these volumes [16]. This paper, however, will mainly make use of FEM.

2.2 Super-Resolution

Upsampling of coarse-scale data is the process of increasing the count of observations or data points in a data set or signal. Super resolution is the process that attempts to upsample data while keeping certain features like structure, sharpness, texture preservation, and geometric invariance. It is often used to specifically increase image resolution.

2.2.1 Interpolation

Interpolation is one of the methods focusing on determining values between data points in a given data set. While simple methods like piecewise or linear interpolation are a fast way to estimate data, they often do not reflect the real underlying function, so more sophisticated methods, like splines, are used. Spline interpolation essentially utilizes polynomial interpolation, but instead of fitting a single high-degree polynomial to all values, multiple polynomials, or splines, are applied to sub-segments of the original value set.

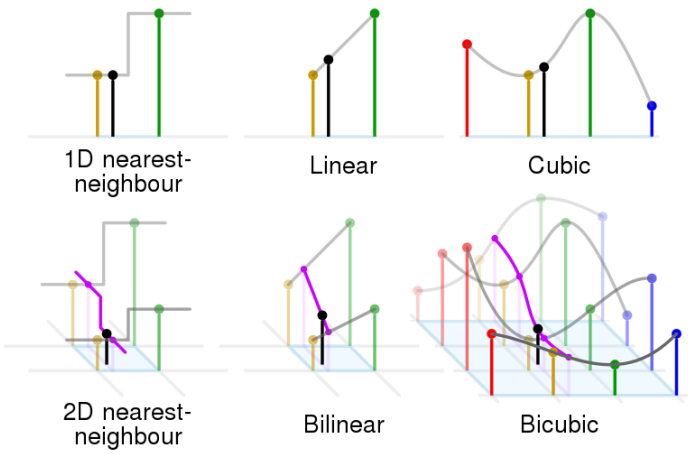

Figure 2.2: Types of 1D and 2D interpolation in comparison [4].

Bicubic interpolation is a method of using cubic spline interpolation in a two dimensional space, thus using 16 samples in a grid of 4x4 as the kernel shape, as seen in Figure 2.2

and iterating it over the image. The resulting image is smoother than bilinear or nearest neighbor interpolation, which only use two data points per dimension. Although this method can be used to attempt to super-sample a set of data points very quickly, results may still contain artifacts, appear coarse, and change the underlying structures of the data, like blurring sharp borders.

2.2.2 Neural Networks

To achieve the above-mentioned sharpness and detail for an image through super resolution, often neural networks are used in combination with, or without, regular interpolation. The first method to utilize deep learning for super resolution was the Super Resolution Convolutional Neural Network (SRCNN), which consists of convolutional layers to extract details of the original image. These convolutional layers use a set of filters with learnable parameters that iterate over an image or grid-like set of input neurons and compute a feature map. An example can be seen in figure 2.3

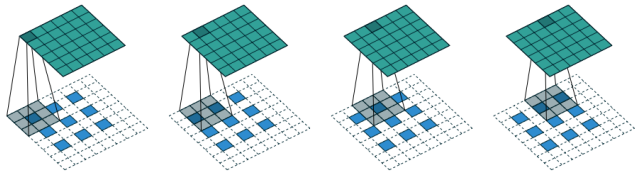

Figure 2.3: The workings of a convolutional layer [7].

Before running an image through the SRCNN model, the input is upsampled using bicubic interpolation to the desired output resolution as the pre-processing step. The complete structure of the model consists of three steps: patch extraction and representation, non-linear mapping, and reconstruction. A representation of the structure is shown in figure 2.4.

1. Patch extraction and representation creates overlapping patches from the interpolated low-resolution image, which are composed of feature maps.
2. Non-linear mapping projects each vector of the previous layer onto another high-dimensional vector, representing patches of the high-resolution image.
3. Reconstruction aggregates the high-resolution patches and reconstructs them back into the final high-resolution image.

During training, the loss between the high-resolution image and the output of the SRCNN is calculated using the mean square error [5]

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{X}_i - \mathbf{Y}_i)^2 \tag{2.3}$$

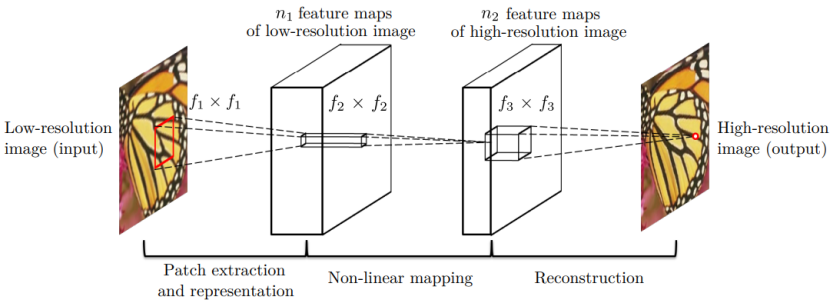

Figure 2.4: The architecture of the SRCNN model shows the three layers. [5]

.

where $\mathbf{Y}$ is the matrix of n predictions and $\mathbf{X}$ are the actual observed values. It should also be mentioned that for tasks that include image transformation besides super resolution, the perceptual loss function may be used. It calculates the MSE not only between in- and output images but also combines it with samples from the intermediate layers of the network, extracting features and representations of the image and minimizing the difference between them [8]. To evaluate the results on a higher than pixel-to-pixel level, the peak signal-to-noise ratio (PSNR) is often used as an image quality metric and calculated as

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right) \tag{2.4}$$

with R being the maximum of the input range, so for an image of pixel values as 8-bit unsigned integers, it would be 255. Although PSNR is used as the de facto metric in modern super-resolution research, it is considered a poor image quality metric for visual characteristics. This prompts the utilization of better visual quality metrics, like the structural similarity index (SSIM), which often accompanies PSNR in super-resolution quality measurement. SSIM is calculated pixel-wise for two images $\mathbf{X}$ and $\mathbf{Y}$ with the equation

$$SSIM(\mathbf{x}, \mathbf{y}) = l(\mathbf{x}, \mathbf{y}) c(\mathbf{x}, \mathbf{y}) s(\mathbf{x}, \mathbf{y}) \tag{2.5}$$

where the vectors $\mathbf{x}$ and $\mathbf{y}$ represent the k-pixel local neighbors of images $\mathbf{X}$ and $\mathbf{Y}$ and l represents luminance, c contrast, and s structural similarity of the pixel neighborhood, and these terms are defined as

$$
\begin{aligned}
l(\mathbf{x}, \mathbf{y}) &= \frac{2\mu_g\mu_d + c_1}{\mu_g^2 + \mu_d^2 + c_1} \\
c(\mathbf{x}, \mathbf{y}) &= \frac{2\sigma_g\sigma_d + c_2}{\sigma_g^2 + \sigma_d^2 + c_2} \\
s(\mathbf{x}, \mathbf{y}) &= \frac{2\sigma_{gd} + c_3}{\sigma_g\sigma_d + c_3}
\end{aligned}
$$

where μ_g,μ_d,σ_g,σ_d and σ_{gd} are local statistics and estimated from a neighborhood of 11 x 11 pixels. They are defined as

$$
\begin{aligned}
\mu_g &= \sum_{i=0}^{k-1} w_i g_i \\
\sigma_g &= \left(\sum_{i=0}^{k-1} w_i (g_i - \mu_g)^2 \right)^{\frac{1}{2}} \\
\sigma_{gd} &= \sum_{i=0}^{k-1} w_i (g_i - \mu_g)(d_i - \mu_d)
\end{aligned}
$$

where k is the length of the vectors $\mathbf{x}$ and $\mathbf{y}$ respectively, and $\mathbf{w}$ a circular-symmetric Gaussian window with a standard deviation of 1.5 samples and normalization to unit sum. This computes the SSIM pixel-wise and locally for parts of an image, and to acquire the SSIM for any image, the mean MSSIM is used. It is also just referred to as SSIM, as the local SSIM values serve no real purpose in the context of image comparison metrics [18].

Although the SRCNN has already achieved remarkable results, there have been many adaptations to improve quality and performance, like Very Deep Super Resolution (VDSR). As the name suggests, it uses more hidden convolutional layers with smaller 3x3 filters, extracting features from the residual of the interpolated input instead of the direct mapping [9]. Since the feature extraction of the SRCNN, or VDSR, occurs in the already interpolated image, other methods utilize post-upsampling to improve performance, as the lower initial input size decreases the required computational power. For example, the FSRCNN skips the pre-processing bicubic interpolation step and instead implements a deconvolutional layer at the end of the model to expand the image to the intended size [6]. Further approaches to SR include skip connections. The SRResnet, or further EDSR networks, involve residual blocks, which feed the input of the block through convolutional layers, ReLU layers (and for the SRResnet batch normalization), and then combine it together with the original input of the block, ”skipping” the layer and thus containing the original information as part of the output, as shown in figure 2.5 [14].

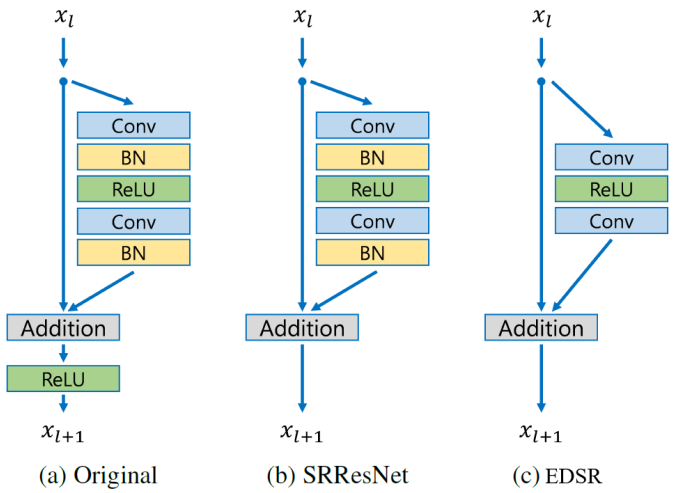

Figure 2.5: Residual blocks utilizing skip connections, with a) as the originally proposed residual block, b) a block in the SRResNet and c) being a block in the EDSR model [14].

To upscale images with even larger scale factors of 8x or more, the regular SRCNN and other variants might perform worse, as their architecture constrains the feasible quality of such a task and a method of increasing resolution in steps is employed. For example, the LAPSRN consists of a Laplacian pyramid structure, which increases resolution to 2x, 4x, 8x, etc. after each step. The extracted residual images after each layer are also added to the interpolated input of the next layer, resulting in a shared learning process between layers and improving image quality at higher scales [10]. In many cases, super resolution does not just intend to generate a sharper image based on pixel-by-pixel differences but rather create an image perceptually pleasing to the human eye. To improve quality in this regard, an adversarial approach can be used, utilizing not only the previously discussed loss functions but combining them with the adversarial loss of the discriminator. Such a GAN-based architecture is used by SRGAN, EnhanceNet, and ESRGAN. Although they result in lower PSNR scores, these models achieve a higher MOS and better visual quality [11].

2.3 preCICE

Coupling simulations in space and time can be achieved using preCICE, an open-source coupling library for partitioned multi-physics and multi-scale simulations written in C++ and available through the GNU Lesser General Public License. Currently, it supports commonly used simulation software programs, including OpenFOAM, SU2, Code_Aster, CalculiX, Nutils, and more. A general conceptual overview of the capabilities can be seen in Figure 2.6.

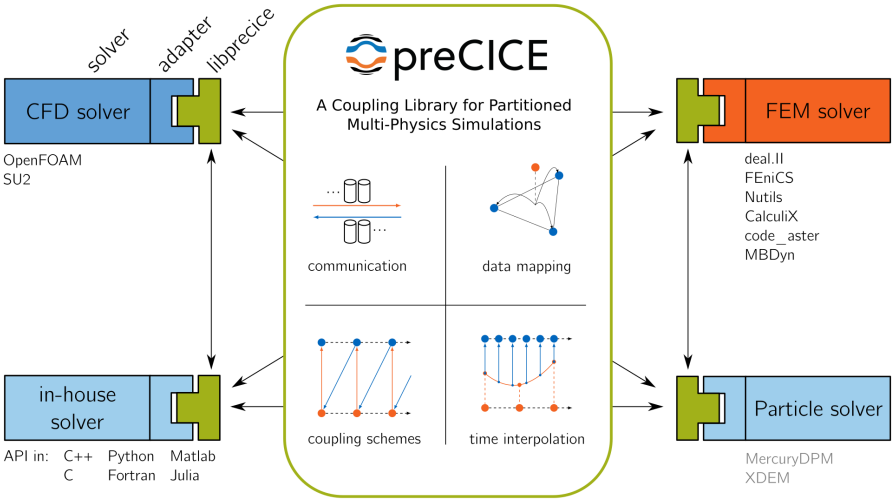

Figure 2.6: An overview of the capabilities of preCICE [3]

preCICE offers parallel communication between solvers as well as data mapping schemes for coupling. Coupling between solvers and neural networks works by projecting the data that is to be transferred onto a mesh, which is then mapped onto the receiving mesh and can afterwards be used by the receiving participant. The structure of coupling, including participants, transfer directions, data types, and mapping styles, is defined in a configuration file using XML. Furthermore, the configuration includes the time frame and time steps for coupling, allowing for synchronization of solvers in time, even allowing for time interpolation, and through the periodic nature of the data exchange, allowing the coupled solvers to run independently. Other key features of preCICE include mechanisms to robustly ensure the convergence of coupled simulations, load balancing, and error estimation for improved stability of the simulations [3].

3 Upsampling fluid dynamics with CNNs via preCICE

3.1 Problem Statement

The purpose of this project can be expressed as a major research question and a minor one.

1. Is it possible to apply super-resolution through the use of convolutional neural networks to computational fluid simulations?
2. Can preCICE be used to couple simulation programs/solvers and neural networks?

The first question attempts to answer the possibility of improving otherwise computationally expensive and time-consuming high-resolution fluid simulations through super-resolution methods. To aid in this combination of technologies, the second question also possibly provides a solution for coupling and streamlined implementation through preCICE. This would also show a modular approach to effortlessly coupling any type of fluid simulation to any pre-trained neural network, as well as data-independent training and testing of such a network.

3.2 Neural Network Architecture and Similarity Metrics

The deep neural network used in this thesis is a super-resolution convolutional neural network, as it is a rather simple model with some range of flexibility in general image super-resolution. For more complex fluid simulations, more advanced networks may be considered, as well as LAPSRN architecture for super resolution of higher scale values, though the simple fluid simulation starting at a base resolution of 60x16 does not necessitate the utilization of such models. Generative adversarial models have already proven the capability to upsample fluid simulations while outperforming standard non-machine learning-related methods [19, 12]; however, since this specific field is currently not particularly well-researched, the SRCNN creates an adequate baseline for testing the super resolution as well as coupling.

The output of fluid simulation solvers at each time step is represented in a grid of points, each of which contains the density of the substance inside the fluid, as will be explained in detail later. Since this grid is equivalent to an image with a single channel (or color value) for each point (or pixel), this structure can be seen as an image and treated as such in the

context of super-resolution. Therefore, the simulation output at certain time steps and the inputs and outputs of the neural network will be referred to as images in this paper.

As per the original SRCNN paper [5] the model structure is built using three convolutional layers. The first takes an input channel size of 1, as the density of the species at every discrete position is represented using one value, and the output channel count is 64 for the extracted patches. The kernel size in this layer is 9, with a padding of 4, and the result is passed through the ReLU activation function. Next for the non-linear mapping step, the convolutional layer narrows the output channel size down to 32, this time with a 5x5 kernel, padding of 2, and once again concluded by the ReLU. Finally, the last convolutional layer reconstructs the image back to a single channel, again with a 5x5 kernel and padding of 2 pixels. There are two pre-processing steps required for this structure: the conversion of data and interpolation.

1. While the data is being gathered, it has to be converted to a rectangular format, as convolutional layers require it to parse the image using a squared kernel. Therefore, pixels of zero density species have to be inserted into the mesh where the channel contains the obstacle, creating images of rectangular shape and without holes.

2. The stage before the image can be used as an input for the SRCNN is the bilinear interpolation step. Since the model requires equal input and output dimensions, the images are scaled to their desired output size.

The loss function used for backpropagation is MSE loss, and the model is initialized using the Adam optimizer with a learning rate of 0.0001. Training the model utilizes the complete dataset over 100 epochs, splitting the whole set randomly into 70% training data and 30% evaluation data. Each epoch, first the model is trained on all of the training data, then in the evaluation step, the average PSNR and SSIM are calculated, and the best weights are selected based on the highest average PSNR value of that epoch.

It should be noted that bilinear interpolation tends to keep the structured similarity of images close to the original, usually lowering the SSIM to no less than 0.95. This results in a generally high SSIM for the SRCNN output as well and makes the SSIM rather unsatisfactory for evaluation, thus establishing PSNR as the metric to determine the best weights for the model. There are of course other metrics that can be used, including adversarial loss if the model is trained in an adversarial approach; however, for the scope of this thesis and considering the usage of a simpler SRCNN, more advanced loss functions do not add a significant improvement and only complicate the presented scenarios. PSNR is not considered a high visual quality metric, though perceptual image quality is of less importance in the context of fluid simulations, and since it is still used as a very common measure, it will also be the main comparison point in the following simulations. Other metrics such as the Visual Saliency-induced Index, Spectral Residual Based Similarity, and other more recently developed methods of visual quality benchmarking, including deep learning-based similarity, heavily focus on subjectively finer visual comparisons and are, as previously stated, of minor to no importance for simulation evaluation.

3.3 Coupling and Data Generation

The coupling schemes and preCICE configurations differ between training and evaluation runs. As the SRCNN model is trained on a fixed set of time steps over multiple epochs and iterations, the training data is gathered from the solvers before the training process begins. For any given timestep of the simulation, first the fluid solver calculates the fluid velocity fields and sends the information through preCICE over to an instance of a course solver and one of a fine solver for the transport problem. The coupling scheme hereby described is serial-explicit, meaning there is a one-way data flow from fluid solvers to transport solvers. The mapping between the meshes is based on the C2-polynomial radial basis function to approximate the values to the mesh of the receiver at a different resolution. After the advection-diffusion calculations for a timestep are complete, the substance densities are further forwarded in the same way as the velocity values to the SRCNN participant, which gathers both solutions for a timestep and stores them in a dataset. This process is repeated for the predetermined number of time steps until there is a data pair with a high resolution and a low resolution solution for each step. The coupling scheme for this is shown in Figure 3.1.

After completion of gathering the data, the preCICE interface is finalized and the model trained as described in the SRCNN and Similarity Metrics section 3.2. Since the data collection process and model training are run sequentially, the training process can be run after an arbitrary amount of time after collection. Collecting the training data using preCICE coupling is a convenient method of modularly attaching it to any type of solver. It should also be noted that another possibility for generating data for network training could be to only generate the high-resolution data, downsample it to a lower resolution, and use that as an input for the model, as is commonly used in model training for image super-resolution. This method, however, might not necessarily create the same kind of low-resolution data a course-scale solver would create because downsampling does not consider physical constraints like conservation of mass while decreasing the number of data points. To utilize and evaluate the trained model without the pre-calculated higher-resolution data, a slightly different scheme is used, as the integration of the second fine solver is no longer necessary. The configuration of fluid and transport solvers is kept as in the training step, excluding the fine solver, as shown in Figure 3.2. For this, the model is simply initialized with the saved weights at the beginning of the program on the SRCNN participants side. During coupling, the SRCNN participant then receives the data once again from the transport solver, except this time it is directly interpolated and fed into the neural network, allowing for parallel simulation and interpolation in real-time.

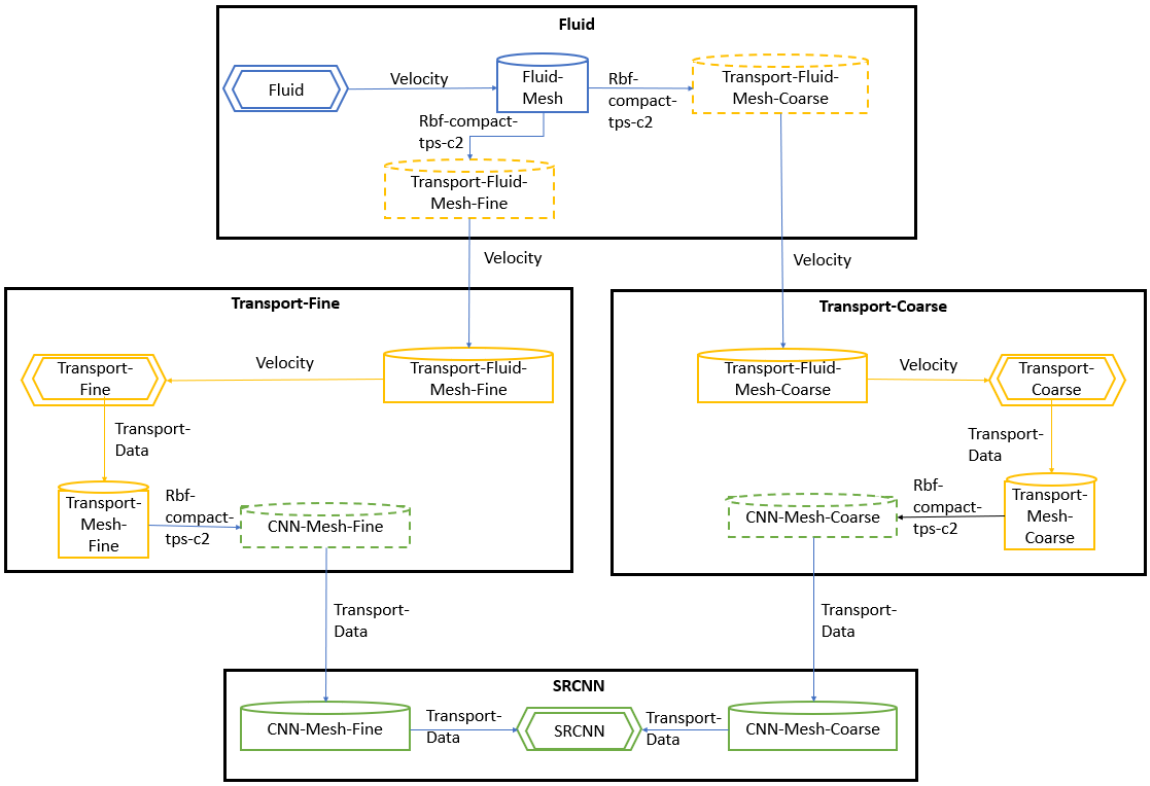

Figure 3.1: The coupling scheme for data gathering.

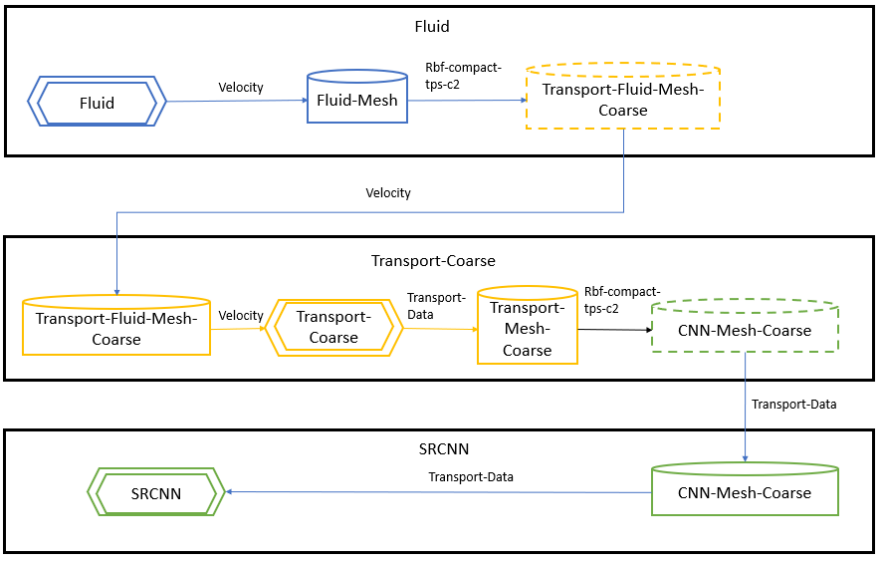

Figure 3.2: The coupling scheme for SRCNN evaluation.

3.4 Computational Experiments

To test the proposed theory through computational experiments, the previously described coupling and super-resolution structure is applied to the CFD problem as described in the preCICE tutorial library "Channel transport" [1]. The original simulation is a CFD problem, coupled to a transport problem in a unidirectional way. Originally, a two-dimensional incompressible fluid flowing through a channel with an obstacle at the bottom of the channel was modeled, and a circular blob of substance in proximity to the inflow was placed in the channel as shown in Figure 3.3.

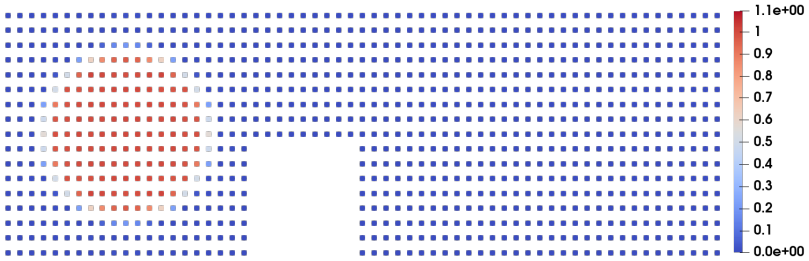

Figure 3.3: Regular channel modeled in a 60x16 grid with a step in the middle.

The flow of the blob species is simulated over up to 200 timesteps, flowing left to right with boundaries at the top and bottom, including the step. After gathering the data for the 60x16 low-resolution and the higher-resolution simulation of 120x32 points, the SRCNN is initialized. The model is then trained on all 200 timesteps over 100 epochs. The best-performing weights, determined by the highest PSNR from all epochs, are used further for general evaluation in different scenarios.

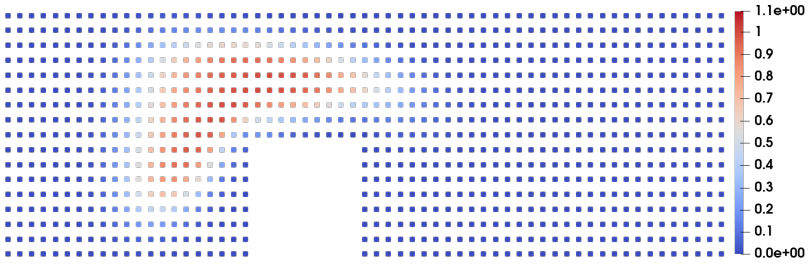

Figure 3.4: Solver output at timestep 17 with 60x16 resolution.

The color in all figures represents the density of the species in the fluid channel. Higher red values depict a higher density, while blue values show the fluid containing less or none of the species' particles. For further comparisons, figure Figure 3.4 shows the simulated location of the species at timestep 17 with a baseline resolution of 60x16.

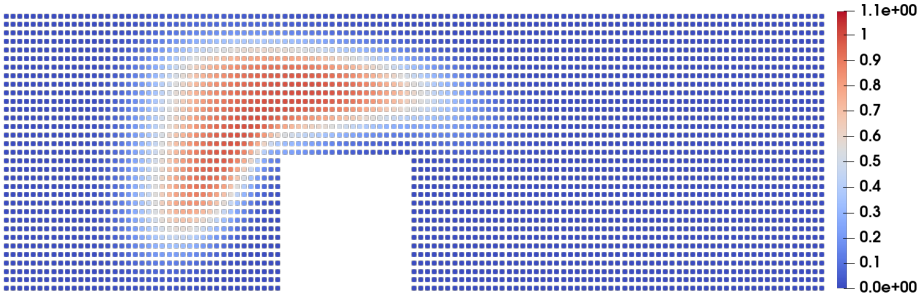

Figure 3.5: Solver output at timestep 17 with 120x32 resolution.

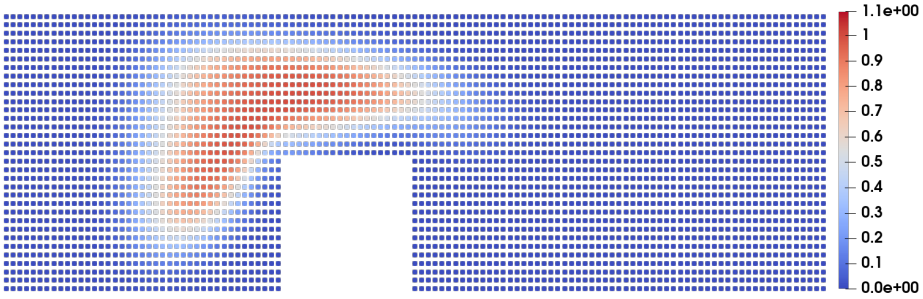

Figure 3.6: SRCNN output at timestep 17 with 120x32 resolution.

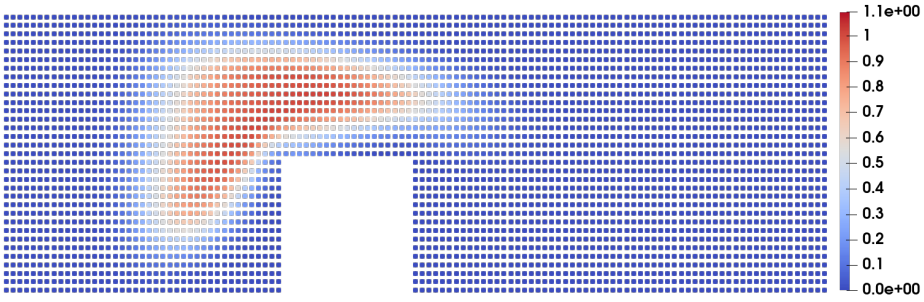

Figure 3.7: Bilinear interpolation output at timestep 17 with 120x32 resolution.

Doubled resolution simulations, which are also the training set for the SRCNN, have their outputs compared in Figure 3.5 for the fine resolution solver and Figure 3.6 showing the output of an already trained network. Visually, there are only a few differences that are recognizable, and the SRCNN achieves an average PSNR over all 200 timesteps at double resolution of 48.86 dB as well as a SSIM of 0.997. Comparatively, the bilinearly interpolated simulation shown in Figure 3.7, which also serves as the input of the model, merely achieves an average PSNR of 33.99 dB and a SSIM of 0.986. This already shows a major improvement of the SRCNN model over the interpolation method. Measuring the time required to calculate the coarse simulation, interpolation, and SRCNN output combined, as well as the fine calculation, reveals that the super-resolution setup on average requires 45.36% less time to compute the double resolution images. The interpolation and SRCNN only require 0.15% of the computational time of the whole process, while the rest is used by the low-resolution solver.

Although the simulations focus on concrete data-point generation for further use with less emphasis on visual quality, it is still advantageous to look at the generated output comparisons, as this might give useful insights on what causes errors or irregularities between the images.

The next experiment is performed at a higher resolution to test the scalability of the model. In this case, the resolution is increased to x3, or more specifically, 180x48 pixels. Figure 3.8, Figure 3.9 and Figure 3.10 show examples of the conventional solver's solution, the model output, and the interpolated solution, respectively, at timestep 17. As expected, the interpolation method shows rough edges and a blocky gradient visually and only reaches an average PSNR of 31.85 dB and a SSIM of 0.969 compared to the solver's output. In comparison, the SRCNN model shows a much smoother distribution and achieves an average PSNR of 37.16 dB and a SSIM of 0.988. While the metrics do show a significant decrease in quality, time efficiency seems to improve on larger scales. The triple resolution super-resolution, including the low-resolution solver, interpolation, and SRCNN, now on average performs 76.46% faster than the conventional fine-scale solver. Most of the time is again used by the low-resolution solver, as interpolation and SRCNN require less than 0.2% of the time.

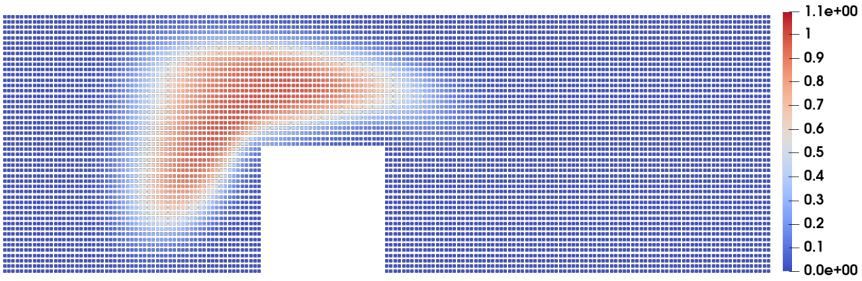

Figure 3.8: Solver output at timestep 17 with 180x48 resolution.

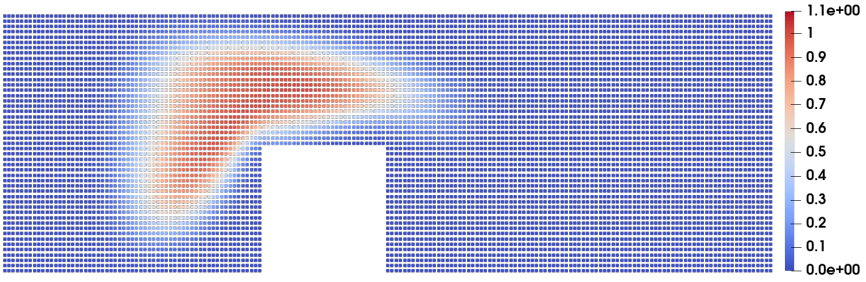

Figure 3.9: SRCNN output at timestep 17 with 180x48 resolution.

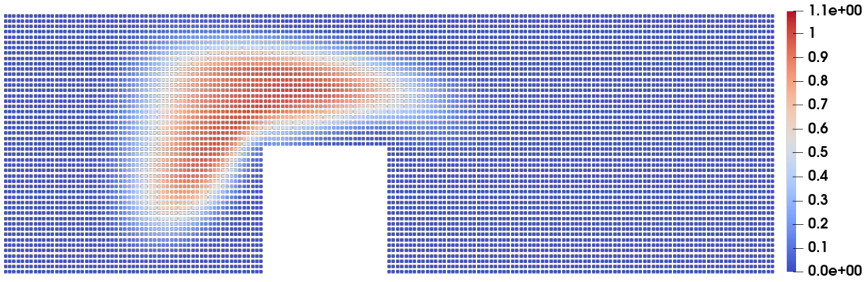

Figure 3.10: Bilinear interpolation output at timestep 17 with 180x48 resolution.

A further computational experiment intends to examine the capability of the SRCNN to upsample a non-unified set of conditions. For this, instead of using a singular placement of the species as a starting point, two smaller concentrations of the same species are placed in the channel next to each other, as shown in Figure 3.11. This is then equally simulated over 200 timesteps, and the performance of the super-resolution process is evaluated.

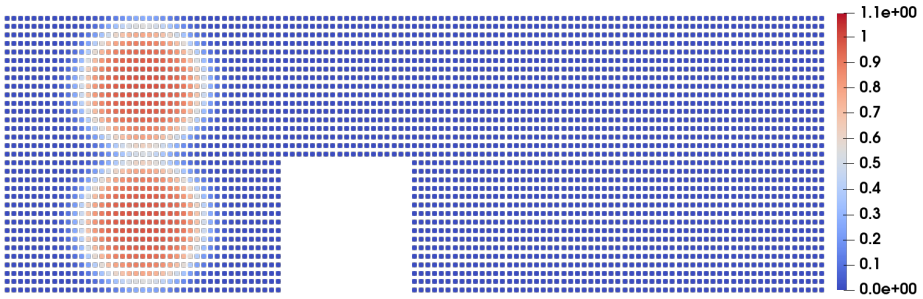

Figure 3.11: Solver starting position of two blobs with 120x32 resolution.

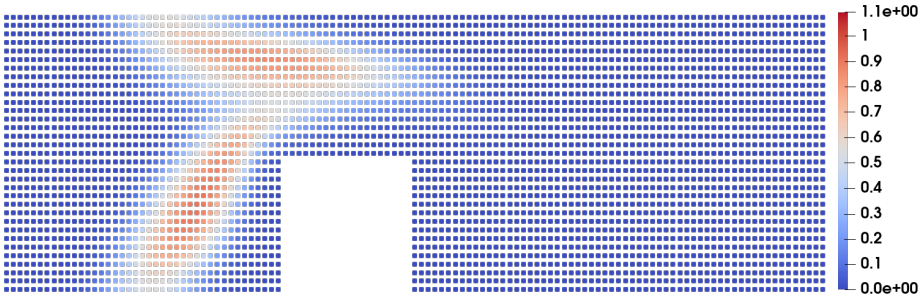

Figure 3.12: Solver output at timestep 17 with 120x32 resolution using two blobs as a starting condition.

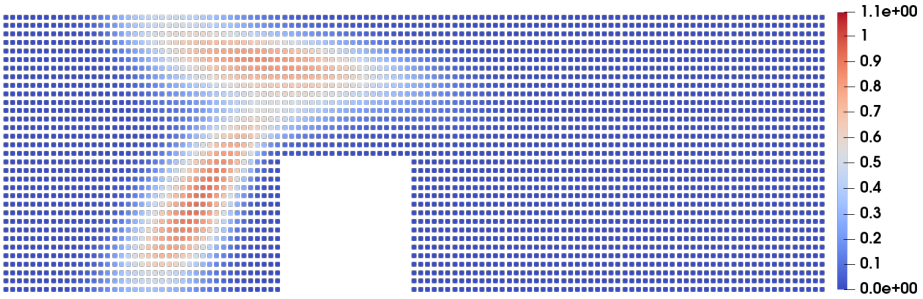

Figure 3.13: SRCNN output at timestep 17 with 120x32 resolution using two blobs as a starting condition.

This particular experiment shows the difficulty for the SRCNN to predict the correct values at the edges of the simulations. As seen in Figure 3.13 the model predicts smaller values at the top and bottom of the channel than are actually calculated by the high resolution solver in Figure 3.12. Noticeably, the PSNR of the SRCNN averages at 36.80 dB, only barely increasing from the interpolation average PSNR of 35.61 dB.

The last scenario considered in this thesis involves simulating a different mesh. Similar to the other scenarios, a species is placed in the channel with a fluid running through it from left to right. Here, however, the obstacle in the channel is placed in the middle of the channel, with the fluid flowing around it and causing sharper gradients.

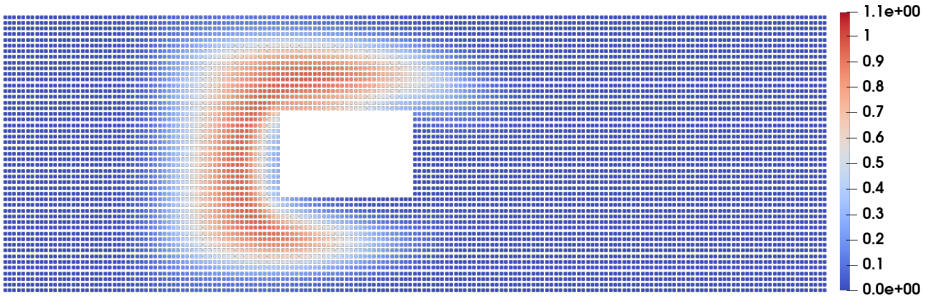

Figure 3.14: Solver output at timestep 23 using 180x48 resolution and including an obstacle in the middle.

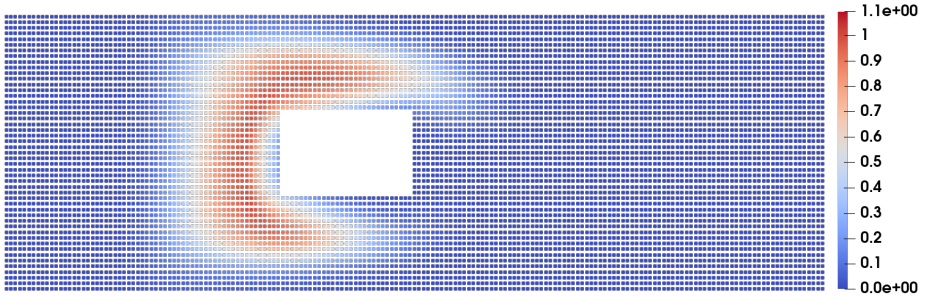

Figure 3.15: SRCNN output at timestep 23 using 180x48 resolution and including an obstacle in the middle.

Figure 3.14 and Figure 3.15 compare the results of the solver and SRCNN, respectively, at triple resolution. A visual comparison shows the model's inability to consider the shape of the underlying structure. Since the input for the model necessitates the conversion to an image and the model receives no additional information on the specific shape or forces, besides the low resolution density values, the model output in Figure 3.15 seemingly allows for the substance to move while clipping through the obstacle in the top left corner. In contrast, the solver's output in Figure 3.14 correctly moves the substance along the side of the obstacle. In terms of metric comparison for this simulation, the SRCNN performed with an average PSNR of 39.03 dB for 2x resolution and 32.04 dB for triple resolution. This compares to the interpolated values of 33.19 dB for double and 29.76 dB for triple resolution. SSIM values once again stay above 0.96 for both methods.

In conclusion, the following Table 3.1 can be created for comparison. Hereby, experiment Normal refers to the original channel transport simulation; Two blobs is the run with two separate smaller instances of the species; and Obstacle shows the values for the experiment with the block placed in the middle of the channel. The shown values of PSNR and SSIM are averages and comparisons to the high-resolution simulations. The rightmost data labeled "Time as % of solver" shows the average computation time taken by the super-resolution process as a percentage of the time taken by the higher precision solver. The time measured for super-resolution includes low resolution solver calculations, interpolation, and SRCNN. It should be mentioned that the interpolation and SRCNN calculations combined take a negligible 0.1% to 0.25% of the total super-resolution time, essentially making this data a comparison of the low resolution solver to the high resolution solver.

As expected, higher scale values for the simulations result in lower PSNR ratings and a slightly lower SSIM, although this metric is barely influenced over all simulations. Since most of the computation time is utilized by the solvers, super-resolution drastically increases the speed compared to the regular solver.

Table 3.1: Comparison of the SRCNN and interpolation methods to the solver in PSNR, SSIM and computation time.

	PSNR in dB		SSIM		Time as % of solver
Experiment	SRCNN	Interpolation	SRCNN	Interpolation	SuperRes
Normal x2	49.61	35.01	0.99	0.99	54.68%
Normal x3	36.71	31.85	0.99	0.97	23.53%
Two blobs x2	36.80	35.62	0.99	0.99	63.93%
Obstacle x2	48.54	33.69	0.99	0.98	60.9%
Obstacle x3	32.04	29.76	0.97	0.97	12.76%

4 Conclusion

The objective of this thesis is to investigate the feasibility of utilizing super-resolution techniques in fluid simulations and demonstrate the possibility of coupling simulations and solvers with convolutional neural networks in the process using preCICE. It has is demonstrated that applying super-resolution to low-resolution fluid simulations is possible and drastically increases computation time, with the drawback of less precise data. It can also be concluded that coupling simulations to super-resolution is viable by using preCICE. This process allows for the connection of arbitrary solvers to any kind of upsampling technology very quickly. These conclusions are drawn from training an SRCNN on fluid simulations and performing computational experiments on it. Results of these experiments indicate decreased precision in upsampled simulations compared to high-resolution simulations, especially when increasing the resolution above 2x, as is the expected case in regular image super-resolution using SRCNN. Triple resolution in this case shows a reduction of up to 34% in PSNR values compared to double resolution simulations. Further drops in data precision occur when changing the geometry of the underlying mesh in the simulation and setting different initial conditions for simulations, especially for different amounts and positions of substances. Evidently, the SRCNN's missing input of underlying structure results in less accurate super-resolution when utilizing complex meshes and sharper gradients. However, the steadily high SSIM shows that the model is capable of keeping structure in simulated and upsampled results, making it a good solution for estimating fluid density at higher resolutions without the need to compute high-resolution simulations themselves. Other benefits include a shorter computation time. Training the SRCNN only takes constant time and is required once, while the evaluation computations for the model require little time compared to low-resolution simulations, essentially decreasing the effort required for a higher-resolution simulation to that of a lower-resolution one.

A way of improving correctness and scaling in future work might be to use more sophisticated models and larger datasets, even on three-dimensional data. The most effective model choice would be dependent on the type of simulation and desired result, like using LAPSRN for higher scale factor values. Additionally through the capabilities of preCICE coupling it is possible to rapidly interchange solvers and upsampling frameworks, allowing for quick testing capabilities when researching in this field.

Bibliography

[1] https://precice.org/installation-distribution.html.

[2] Gualtiero Badin and Fulvio Crisciani. *Variational Formulation of Fluid and Geophysical Fluid Dynamics*. Springer Cham, 2017.

[3] G Chourdakis, K Davis, B Rodenberg, M Schulte, F Simonis, B Uekermann, G Abrams, HJ Bungartz, L Cheung Yau, I Desai, K Eder, R Hertrich, F Lindner, A Rusch, D Sashko, D Schneider, A Totounferoush, D Volland, P Vollmer, and OZ Koseomur. preCICE v2: A sustainable and user-friendly coupling library [version 2; peer review: 2 approved]. *Open Research Europe*, 2(51), 2022.

[4] Cmglee. Comparison of 1d and 2d interpolation, 2020.

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015.

[6] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network, 2016.

[7] D. Foster. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*. O'Reilly Media, 2019.

[8] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

[9] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks, 2016.

[10] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks, 2018.

[11] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2017.

[12] Matthew Li and Christopher McComb. Using Physics-Informed Generative Adversarial Networks to Perform Super-Resolution for Multiphase Fluid Simulations. *Journal of Computing and Information Science in Engineering*, 22(4), 02 2022. 044501.

[13] Yin Li, Yueying Ni, Rupert A. C. Croft, Tiziana Di Matteo, Simeon Bird, and Yu Feng. AI-assisted superresolution cosmological simulations. *Proceedings of the National Academy of Sciences*, 118(19), may 2021.

[14] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution, 2017.

[15] Yongtao Lyu. *Finite Element Method*. Springer Singapore, 2022.

[16] Clovis R. Maliska. *Fundamentals of Computational Fluid Dynamics*. Springer Cham, 2023.

[17] Doug McLean. *Understanding Aerodynamics: Arguing from the Real Physics*. John Wiley and Sons, Inc., 2012.

[18] Ansse Saarimäki. Single image super-resolution using convolutional neural networks, 2018.

[19] You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. Tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Trans. Graph.*, 37(4), jul 2018.