GENERATIVE MACHINE LEARNING MODELS FOR AIRFLOW PREDICTION OF ARCHITECTURAL SPACES

by

Fernando José Claudio Rodríguez

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Architecture and
Master of Science in Information Technology

Charlotte

2021

Approved by:

Prof. Jefferson Ellinger

Prof. Elizabeth McCormick

Dr. Chengde Wu

Dr. Minwoo Lee

ABSTRACT

Fernando J. Claudio Rodríguez. GENERATIVE MACHINE LEARNING MODELS FOR AIRFLOW PREDICTION OF ARCHITECTURAL SPACES
(Under the direction of Prof. Jefferson Ellinger)

Computer Fluid Dynamics (CFD) airflow simulations are not used as often in architectural settings primarily due to time constraints. The proper use of CFD airflow simulations involves a complex setup and run-time process that requires a large amount of expertise on the different stages. This study aims to apply existing generative machine learning algorithms to compute CFD wind velocity simulations to significantly shorter run times while maintaining a relatively high accuracy level. In order to test the proposed hypothesis that machine learning can be used as a method to produce rapid and acceptable results for airflow CFD simulations in the early design stages, multiple machine learning models were created, trained, and tested.

The evaluation metrics consisted of using different image similarity methods to evaluate the accuracy of the images produced by the machine learning model compared to their CFD engine counterparts. In addition, run times between the CFD engine and the trained machine learning model were recorded and compared. These results obtained indicate that GAN application for CFD airflow predictions can produce acceptable results showing a significant run time difference of over a minute between the CFD simulation and the machine learning model.

TABLE OF CONTENTS

LIST OF FIGURES vi

LIST OF TABLES: viii

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: BACKGROUND 5

Section 2.1 – Natural Ventilation & Wind Comfort Criteria: 5

Section 2.2 – Wind Tunnels: 6

Section 2.3 – Computer Fluid Dynamics: 7

Section 2.4 – CFD Setup Process: 8

Section 2.5 – Coupled vs. Decoupled Approach: 9

Section 2.6 – Machine Learning: 9

Section 2.7 – Deep Learning & Artificial Neural Networks (ANN): 10

Section 2.8 – Generative Adversarial Networks (GAN): 11

CHAPTER 3: LITERATURE REVIEW 13

Section 2.1 – Optimization in environmental simulation: 13

Section 2.2 – Architecture & Machine Learning: 14

Section 2.3 – Common Themes, Debates & Gaps: 18

CHAPTER 4: METHODOLOGY 21

Section 3.1 - Data Set Creation: 21

Section 3.2 - Machine Learning & Training: 25

Section 3.3 – Data Evaluation 25

CHAPTER 5: RESULTS AND DISCUSSION 28

5.1 Initial & Padded Data Set: 28

5.2 Padded Data Set Results: 31

5.3 Rectangular Floor Plan Typology Test & Results: 34

CHAPTER 6: CONCLUSION & FUTURE WORK..... 36

REFERENCES: 38

LIST OF FIGURES

Figure 1: Natural Ventilation Strategies: (Left) Cross Ventilation, (Middle) Stacked Ventilation, (Right) Single Sided Ventilation [8] 5

Figure 2: Basic components of an open-circuit wind tunnel [9] 7

Figure 3: (a) Coupled and (b) decoupled approach for analysis of wind-induced cross-ventilation of buildings. [14] 9

Figure 4: Diagram of an Artificial Neural Network 11

Figure 5: GAN Training Framework [21] 12

Figure 6: Detailed set of chosen office models, comparison of simulation and Artificial Neural Networks results and computation time [24] 15

Figure 7 : Combination of 1,840 possible design scenarios [25] 16

Figure 8: Comparison between input image (left), simulated results from actual simulation (center) and predicted results from machine learning model (right) [26] 17

Figure 9: Machine Learning Development for Eddy3D using GAN & Pix2Pix [29] 18

Figure 10: Grid density variations 23

Figure 11: Post Processing Visualization results within Rhinoceros / Grasshopper 24

Figure 12: ORB Feature Detection Example [33] 26

Figure 13: The Structural Similarity Measurement System [34] 27

Figure 14: Initial Dataset Training Results at Epoch 5 28

Figure 15: Initial Dataset Test Result with Different Floor Plan Layout 29

Figure 16: (Left) Initial Dataset Image / (Right) Padded Dataset Image 30

Figure 17: Epoch & Batch Size Graph in Comparison in Relation to Image Similarity .. 32

Figure 18 : Epoch & Batch Size Training Comparison 33

Figure 19: Run Time Comparison Between CFD Simulations and GAN 33

Figure 20: Example of Rectangular floor plan (Left) Padded Image / (Right) Non-Padded Image..... 34

Figure 21: Test Results of Rectangular Floor Plan on Both Initial and Padded Machine Learning Model..... 35

Figure 22: Initial & Padded Machine Learning Model for Rectangular Plan Image Similarity Comparison 35

LIST OF TABLES:

Table 1: Criteria for wind comfort and danger in NEN 8100, after Willemsen and Wisse (2007) [10] 6

Table 2: Grid Density Time Comparison..... 23

CHAPTER 1: INTRODUCTION

Over the past couple of decades, buildings have been identified as contributing to the negative environmental impacts worldwide [1]. Being responsible for 40% of carbon emissions [2], with 56% of all energy use going to air conditioning and ventilation in hot-humid climates [3]. Although we often rely on mechanical engineers to size conditioning systems, architects are directly responsible for the performance of building enclosure systems. In order to properly design these, architects must evaluate the current climatic context upon which the building will be constructed. Each analysis is developed on a case-to-case basis, due to each location's different climatic conditions, despite similar tools and procedures.

As energy consumption due to the excessive use of active mechanical systems increases, passive cooling strategies, such as cross-ventilation, have become a reliable alternative. Natural ventilation can decrease carbon emissions, energy consumption, and costs while promoting a healthier lifestyle. However, designing for cross ventilation is no easy feat, as it requires architects and designers to study in detail the building context as it will affect how the airflow will move across the site. Simultaneously, considering multiple factors such as volumetry, window types, and placement to maximize the natural ventilation potential.

Technological advances have simplified and reworked the way architects design, from drawing by hand to later drawing on computer drafting software such as AutoCAD and BIM (Building Intelligent Modeling). Similar processes have occurred for the methods to perform environmental analysis. With the use of different computer software, these

processes can be expedited and performed from a single two or three-dimensional model inside a computer. As buildings rely increasingly on complex mechanical systems, environmental simulations have become more complex, which can be overwhelming to a design professional not trained in comprehensive environmental modeling. Consequently, the production of models, proper setup, and run-time of these tests has become a rigorous and time-consuming process.

However, these environmental simulations are optional in many locations and not required by building codes. Commonly, architects with constant deadlines and time constraints choose not to spend their time running these simulations, as many have identified that "*design time is usually quite short and anything adding to that is an obstacle.*". [4]. Additionally, most design offices do not have the resources or time to explore these problems.

The lack of guidance to understand which parameters are significant for the simulation confuses its users, as simulations do not provide a method for designers to confirm and ensure that the obtained simulation results are accurate and valuable. The purpose of generating these models is to provide helpful information, to inform the building's design, rather than get the most accurate results—subsequently the lack of performance guidelines on the comprehension of the numerical outputs. Making simulation results harder to interpret and, in some cases intimidating for architects [5]. Due to the lack of quality assurance, time constraints, and performance guidelines, it is uncommon for architects and designers to incorporate these testing simulations within their workflow.

Due to improvements in the creations of user-friendly tools, architecture firms are increasingly implementing daylight, radiation, glare, and energy simulations into their

workflows. These tools rely on efficient, straightforward, and well-documented engines that produce data graphically and legibly. Wind and natural ventilation studies, on the other hand, are much more complicated due to exponentially increasing mathematical calculations required to simulate the interaction between airflow and the defined enclosure.

In recent years data has become one of the large's byproducts in history and has revolutionized how we use information. New methods are being developed to manage the increasing size of data sets. A solution to automize the data management process has been incorporating artificial intelligence, specifically, the use of machine learning algorithms. The reliance on these algorithms is significant as computers can now learn from these increasing data sets and perform multiple functions from complex mathematical calculations to image generation.

This study aims to apply existing generative machine learning algorithms to compute CFD wind velocity simulations to significantly shorter run times while maintaining a relatively high accuracy level. These generative algorithms aim to provide faster run times for CFD simulations than traditional simulation processes and obtain relatively accurate results. The upcoming sections will cover the different concepts required for applying both of these fields, exploring different methodologies for creating an efficient data set to train a generative machine learning model. Finally, establishing a set of metrics to evaluate the efficiency of the machine learning model results and comparing them with actual wind velocity simulation.

This study's long-term goals are to continue developing the integration of these technologies with the architecture profession. This integration will allow architects to extend the exploration period during those initial conceptual design states due to the ability

to make mass changes easier. These developments will help promote the use of cross ventilation and the incorporation of CFD models within the design workflow.

CHAPTER 2: BACKGROUND

As diverse topics from architecture and machine learning will be covered throughout this study, this chapter explains a general overview of these, ranging from natural ventilation implementation to understand basic cross ventilation strategies and wind comfort criteria. This section also explores the evolution from wind tunnel testing (WTT) to the use of computer fluid dynamics (CFD). Lastly, it will cover basic concepts of the functionality of machine learning algorithms, deep learning, and generative adversarial networks (GAN).

Section 2.1 – Natural Ventilation & Wind Comfort Criteria:

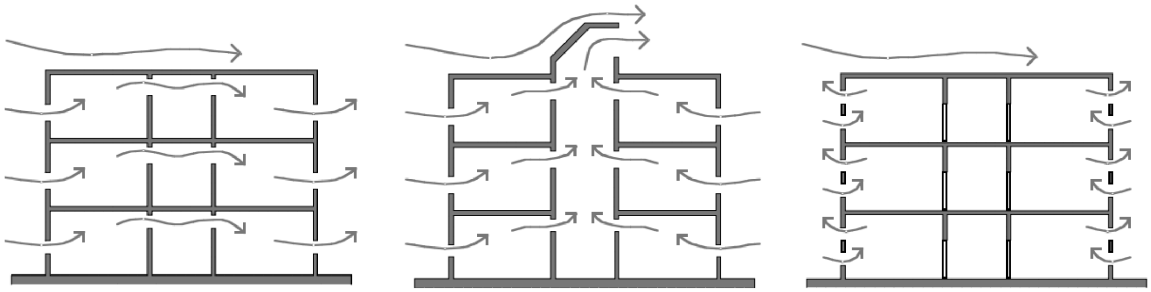


Figure 1: Natural Ventilation Strategies: (Left) Cross Ventilation, (Middle) Stacked Ventilation, (Right) Single Sided Ventilation [8]

Natural ventilation consists of moving air from an external source to an indoor space due to the changes in pressure without any mechanical system. Its usage has many purposes, from air quality control to passively cooling an interior space. Although there are multiple methods for it to be applied to a building, there are three predominant approaches. Wind-driven cross-ventilation consists of providing apertures at opposite ends of the interior space, providing an entry and exit point. Stacked ventilation consists of drawing cool air at lower sections of the building and exhausting the warm indoor air at

higher apertures. Lastly is single-sided ventilation, which provides a local ventilation alternative in single spaces [6].

Wind comfort				
$P(V_{IS}>5m/s)$ in % hours per year	Grade	Activity area		
		Traversing	Strolling	Sitting
< 2.5	A	good	good	good
2.5–5.0	B	good	good	moderate
5.0–10	C	good	moderate	poor
10–20	D	moderate	poor	poor
> 20	E	poor	poor	poor

Wind danger		
$P(V_{IS}>15$ m/s)	Limited risk	0.05-0.3 % hours per year
	Dangerous	> 0.3 % hours per year

Table 1: Criteria for wind comfort and danger in NEN 8100, after Willemsen and Wisse (2007) [10]

There are currently multiple criteria and standards for evaluating the wind conditions at a pedestrian level of comfort and determining their favorability. There are three primary standard criteria, these being Lawson, Davenport, and NEN 8100. These criteria account for a range of activities and include a threshold value of the wind speed and the maximum allowable limits for those threshold values [7]. The NEN 8100 method is one of the most recent standards to be developed, and it accounts for the activities performed and provides a grade of comfort related to the wind speed [8].

Section 2.2 – Wind Tunnels:

Wind Tunnels are tunnel-shaped machines used for producing an airstream to study its movement and effects on different models, ranging from aircraft to buildings. Closed-circuit and open circuit are the two main types of wind tunnels. This study will focus more on the open-circuit type, as it is the most used in architecture. These tunnels contain a fan on one end which will draw air into the tunnel, and pass across the model and release the air on the other end. Models usually contain a set of sensors to measure both pressure and

velocity across the model. Smoke is used, in some cases, to visualize the airflow movement inside the models. [9]

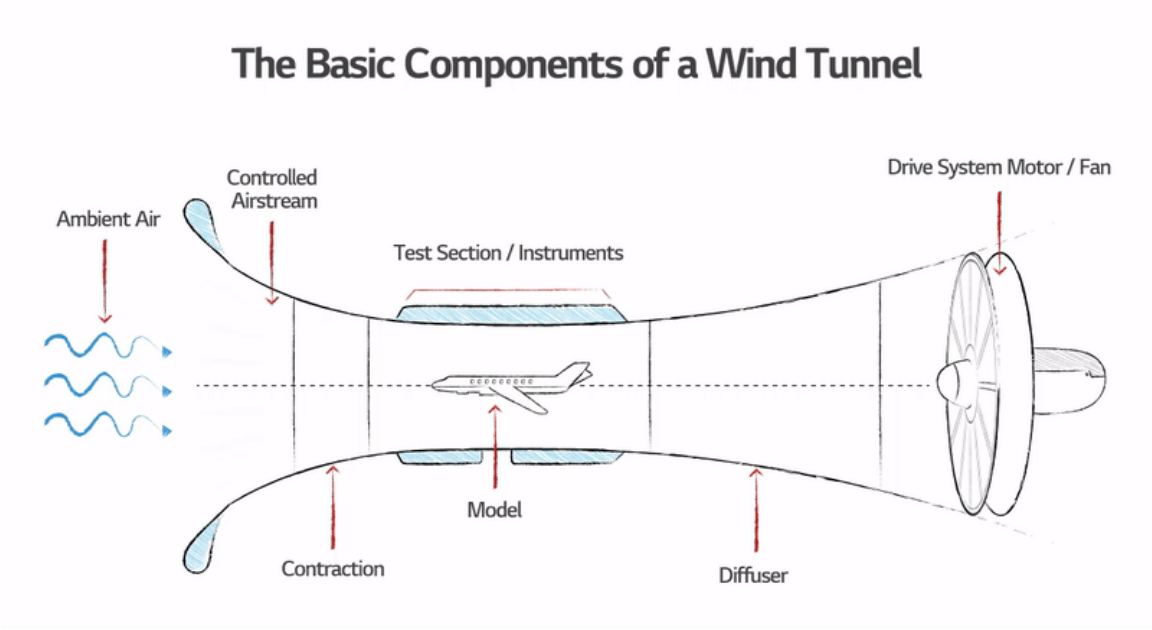


Figure 2: Basic components of an open-circuit wind tunnel [9]

Section 2.3 – Computer Fluid Dynamics:

Computer Fluid Dynamics (CFD) is a combination of multiple fields to simulate various fluids' flow. Although its beginning was in the early 1970s, it is not until a decade later that it began to set the base for some of the modern analysis used today, more specifically the incorporation of Navier-Stokes equations (N-S equations) [10]. These are partial differential equations that describe the flow of incompressible fluids such as water and air. [11]. In architecture, these are used in scenarios to examine natural ventilation, infiltration, and dispersion of air contaminants. The process consists of generating a geometric model and divide it into a cell grid, which helps define the precision level from which the airflow will be examined [5]. After inputting a set of criteria, such as wind velocity and temperature, the simulation will calculate the desired geometry's airflow. CFD

simulations can also help solve many other criteria, but simulations will focus mainly on wind velocity (U) values for this study.

Section 2.4 – CFD Setup Process:

Typically, to set up and run CFD simulations, the process comprises three main stages: pre-processing, solver, and post-processing. The first stage of pre-processing consists of generating the two or three-dimensional geometry for which the identified fluid will be applied. After generating the desired geometry, it is subdivided into a mesh, a collection of vertices, edges, and faces. Lastly, defining the fluid material properties, flow physics model, and boundary conditions. [12].

The solver stage consists of using the previously stipulated parameters and sending them to the solver to identify and perform all the equations and mathematical calculations required for the identified conditions. Once the simulation is solved, its results are stored for further stages. The post-processing stage consists of generating visualizations based on the solver's results to understand better and interpret the numerical values [13].

Although specific software can perform all three stages, common practice consists of generating the three-dimensional models in software such as Rhinoceros 3D or Revit. Use these to edit and convert them into the appropriate format to be imported into another software. The second software prepares all the required conditions. It runs the solver, which in many cases, the results need to be exported to another software such as ANSYS CFD-Post or ParaView to visualize the obtained results.

Although this three-stage process appears very straightforward, it presents a large set of challenges, as it is prone to errors, either while exporting the models between

programs or improperly setting the parameters for the solver. Therefore, just running a simple simulation can be very laborious and time-consuming, not to mention the expertise required for each stage.

Section 2.5 – Coupled vs. Decoupled Approach:

In the use of CFD simulations, there are two main approaches to simulating airflow in relationship to buildings, and these are coupled and decoupled. As its names suggest, the coupled approach involves connecting the outdoor and indoor airflow in a single model. The counterpart is the decoupled model, which only involves analyzing indoor or outdoor environments [14]. As studies suggest ([15],[16],[17]), the use of the coupled model is the optimal approach when it comes to cross-ventilation CFD studies, as it takes into account how the pressure of the building envelope will be affected by the presence of the opening.

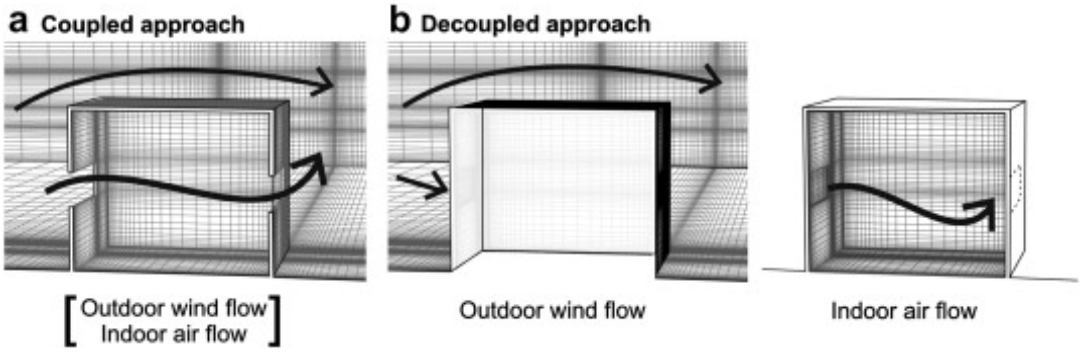


Figure 3: (a) Coupled and (b) decoupled approach for analysis of wind-induced cross-ventilation of buildings. [14]

Section 2.6 – Machine Learning:

There are multiple sciences to be applied in computation; artificial intelligence is one of them. However, within artificial intelligence, there is a subset called machine learning, which has the quest to teach computers to perform a series of tasks without being explicitly programmed for them. This method comprises three key components, first, is the data set from which the model will learn during its training process. Particularly the format

of these can range from all kinds of data types. The secondary component is the features, telling the machine the essential factors to be aware of. Lastly is the algorithm, which provides a particular method to solve the problem. Currently, there are different algorithms to solve similar problems. These can vary in their accuracy and speed, depending on the problem at hand [18].

As there are different kinds of algorithms, there are also multiple ways to teach or use a machine to help solve problems. As described in Python Machine Learning by Raschka & Mirjalili, this consists of three methods, supervised, unsupervised, and reinforcement learning. The supervised machine learning model is trained with a dataset containing labeled information. Its outcomes are known, allowing the model to receive new unlabeled data as input to make predictions. In reinforcement learning, the model generates a "reward signal," which in comparison to supervised learning is not the correct answer but a comparison of how well the performed action relates to the reward function. Lastly, there is no labeled or structure data in unsupervised learning as opposed to either of the previous two alternatives. It is up to the machine to use specific techniques to find patterns and relationships within the data [19]. This study will use supervised machine learning to improve CFD analysis run times to generate predictions on new architectural models to predict the airflow movement.

Section 2.7 – Deep Learning & Artificial Neural Networks (ANN):

Deep learning is a subfield of machine learning inspired by the structure of the human brain. This structure is better known as an artificial neural network comprised of three main layers: input, hidden, and output. These layers are composed of a series of neurons, the core entity for the neural network, that process the information and transfer

them from one layer to another through channels [20]. Each neuron contains a set number called a bias, which gets applied to an active function. Its results determine if the neuron gets activated and passes its value down the network. This process continues until results are passes into the output layer. Although this system is efficient in dealing with unstructured data, it requires a considerable amount to produce valuable results. With more data, the longer these models require to be trained appropriately [21].

Section 2.8 – Generative Adversarial Networks (GAN):

Generative Adversarial Networks (GAN) uses generative modeling in combination

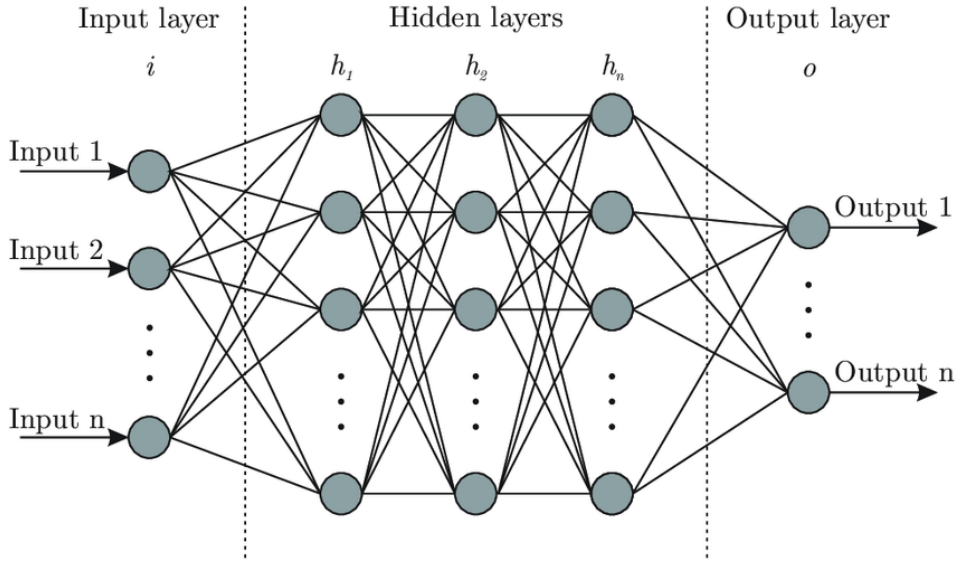


Figure 4: Diagram of an Artificial Neural Network

with convolutional neural networks. The final purpose of GAN is to generate new data predictions from nothing. The network is composed of two models, the generator and the discriminator. The first generate new examples, while the second tries to identify whether the generator's examples are real or fake in a "zero-sum game" [22]. This kind of model's primary data type is images, although other types have been used, such as music. Both networks are trained in an alternating state during the training process until the discriminator model cannot tell the difference between real and fake images created by the

generator [23]. This study will be beneficial as it will generate new predictive visualizations for the new input models based on the trained data set.

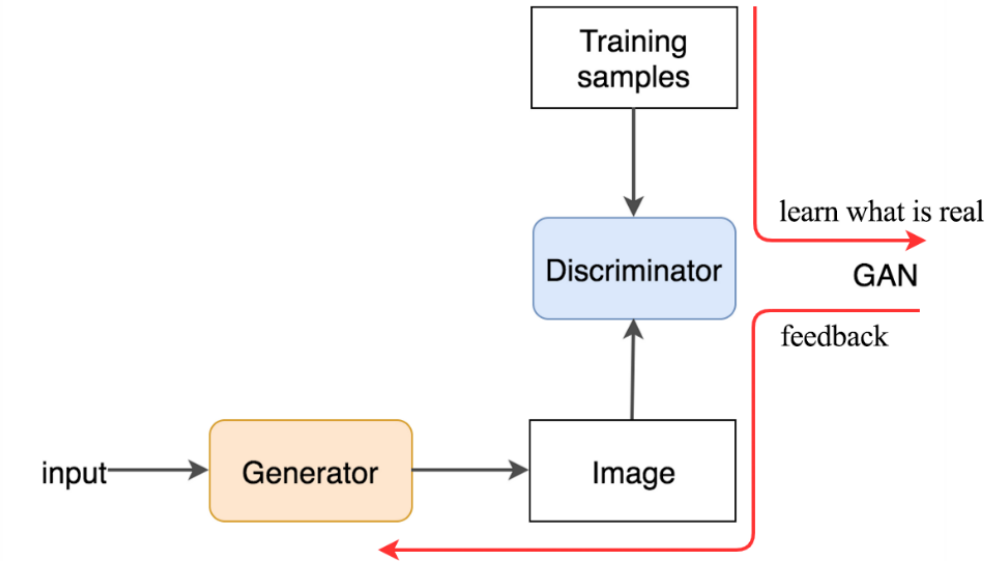


Figure 5: GAN Training Framework [21]

CHAPTER 3: LITERATURE REVIEW

This section aims to discuss existing examples that have optimized environmental simulations by either reducing their setup or run times—exploring the use of cloud-based computing in relationship to CFD simulations. Including other recent studies that incorporate the use of machine learning algorithms within daylight and airflow simulations. All to discover the common themes, debates, and gaps on approaching these algorithms within either daylight or airflow simulations.

Section 2.1 – Optimization in environmental simulation:

Digital simulations have been present in the architecture profession since the beginning of the 1980s. However, CFD simulations expressly have been severely limited by the development of computer hardware. The large computers, high maintenance, physical space consumption, and quickly outdated technologies did not appeal to small and medium-sized firms [24]. However, with the successful commercialization in 1999 of Cloud Computing by Salesforce [25], it became a game-changing course for CFD simulations. Cloud-Based Computing platforms helped to reduce simulation times as well as improve workflow integrations significantly. Since no physical hardware was required and only a monthly fee was required, this began to make the use of CFD simulation more attractive as it did not require any update, maintenance, or physical space consumption.

Some of the benefits mentioned using cloud systems contain another set of benefits, such as little to no preparation time. As these platforms become easier to adapt to for newcomers, it addresses the lack of environmental metrics and guidelines. They provide a series of guided tutorials, a user-friendly interface, a smaller learning curve, and a robust

support network of interdisciplinary online support that can get help if needed. An example of this is the online-based platform called SimScale, which allows users to perform simulations from fluid dynamics to heat transfer. This platform allows users to export their models from their modeling program of choices, such as Rhinoceros, Revit, or Sketchup, and upload them to the cloud platform.

Although not related to CFD simulations, another platform widely accepted by the design industry is called COVE.TOOL. This platform presents its users with tools to perform energy, cost, daylight, and design automation studies within the design workflow, allowing users to iterate through multiple ideas in less time.

Section 2.2 – Architecture & Machine Learning:

The use of cloud-based systems has become a primary alternative to substitute traditional in-house CFD simulations due to their decreased run times. However, there are a series of upcoming alternatives that involve using different machine learning algorithms that significantly reduce the required simulation setup and analysis time. These include artificial neural networks (ANN) and generative adversarial networks (GAN) applied to daylight and airflow simulations.

Regarding the use of ANN, work has been performed for both daylight and airflow simulations. Kacper Radziszewski & Marta Waczyńska (2018) [26] focused on creating a tool to replicate simulation-based results for daylighting. Their data set was comprised of small office spaces and used supervised machine learning to train their neural network. The outcome of these simulations was the use of mean daylight factor (DF), daylight autonomy (DA), and daylight glare probability (DGP). Results were compared with traditional

simulations, indicating an astounding improvement in time ratio and value differences. Their results indicated differences as low as 0.15 in mean daylight factor and computing time ratio of 32.16 (DA), 2.88 (DF), and 16.72 (DGP).

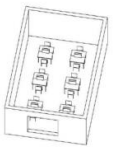
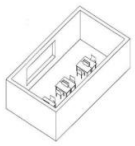
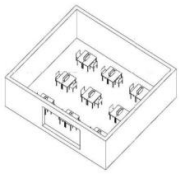
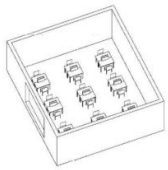
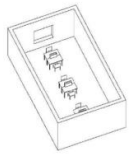
		layout #1	date:	10am 21.03	layout #2	date:	10am 01.09	layout #3	date:	12am 01.12	layout #4	date:	4 pm 01.03	layout #5	date:	4 pm 22.03
		DA	DF	DGP	DA	DF	DGP	DA	DF	DGP	DA	DF	DGP	DA	DF	DGP
Value [%]	simulation	30.5	0.89	27	51	1.95	25.5	26.4	1.28	26	33.4	1.23	34.5	3.4	0.58	23.6
	ANN	28.176	0.888	26.7	57.106	2.171	27	28.3	1.489	27.7	33.52	1.485	29.3	7.047	0.51	24.7
	difference	2.324	0.002	0.3	6.106	0.221	1.5	1.9	0.209	1.7	0.12	0.255	5.2	3.647	0.07	1.1
Time [s]	simulation	15.3	1.4	8.7	12.8	1.2	8.4	17.5	1.6	8.2	18.3	1.7	8.3	14	1.3	8.2
	ANN	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	difference	14.8	0.9	8.2	12.3	0.7	7.9	17	1.1	7.7	17.8	1.2	7.8	13.5	0.8	7.7
																

Figure 6: Detailed set of chosen office models, comparison of simulation and Artificial Neural Networks results and computation time [24]

On the other hand, Chao Ding & Khee Poh Lam (2019) explored the use of coupled models at primarily high-density cities to identify potential cross-ventilation situations [27]. This project addresses variables such as window-wall ratio, wind direction, relative sinuosity, height variation, and urban form, generating around 1840 design scenarios. Due to the simulation's geometrical complexity, these large urban environments, integrating the coupled models' approach, can take up to hours or even days to develop predictions. However, with the incorporation of these algorithms, it can allow for near real-time predictions, thanks to the data-driven model. CIOI is a function of 6 key design parameters that allow for integrating indoor and outdoor environments. More specifically (urban density (UD), height variation (HV), relative sinuosity, wind direction (WD), target building height (TBH), and opening-to-wall ratio (OWR) [27]. Unlike some other CFD machine learning approaches, this contains the most complex deployment and considers many variables for a proper simulation.

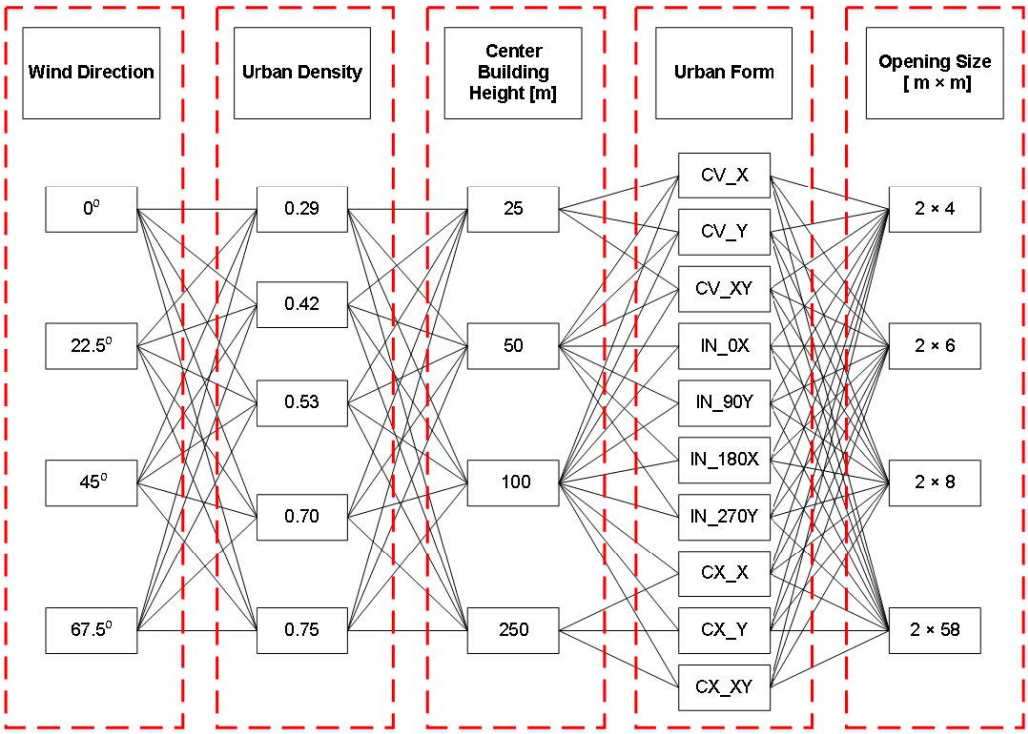


Figure 7 : Combination of 1,840 possible design scenarios [25]

As opposed to Kacper Radziszewski & Marta Waczyńska or Chao Ding & Khee Poh Lam, the previously mentioned projects used ANN, a group called Layout5 used a more holistic approach by applying GAN. Layout5 developed a *"real-time daylight prediction pipeline."* [28]. For this project, the tools consisted of Rhinoceros 3D, Grasshopper 3D, Ladybug Tools, and the deep learning library, pyTorch. This application takes a two-dimensional floor plan as input with labeled features, such as walls, windows, and doors, which pass through the machine learning model and, as a result, produces a daylight prediction heatmap for that configuration. This program's setup begins with generating the training data set, setting up the cycleGAN environment alongside the pix2pix algorithm implementation in pyTorch. Subsequently, the implementation would use the newly created data to train the model and prepare it for new inputs. After setting up the machine learning component, using Grasshopper, they created an interface in which

the user's selected inputs would pass through the model returning the output in seconds. The application provides a more straightforward strategy as training the models would not require understanding any physics or mathematical equations used in previous simulation software. In GAN's case, the machine learning would only need a set of images of the simulation heatmap outputs. Based on those, it would be trained to understand a series of patterns and later generate a new prediction based on those previous experiences.

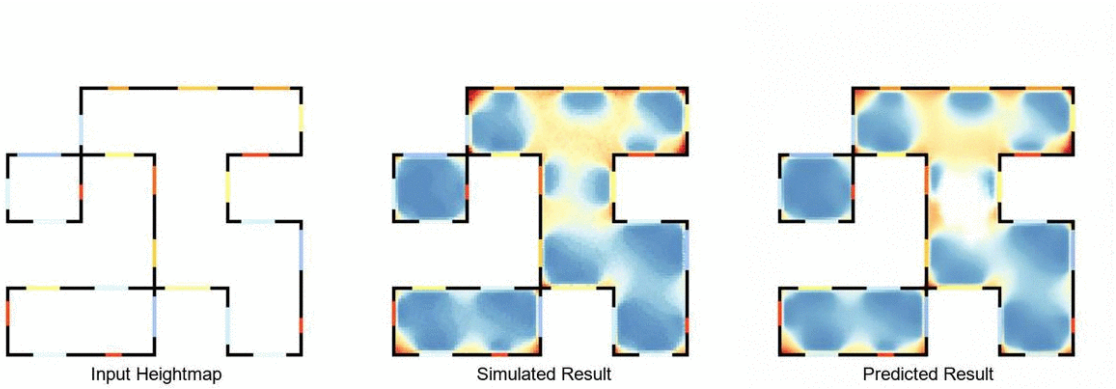


Figure 8: Comparison between input image (left), simulated results from actual simulation (center) and predicted results from machine learning model (right) [26]

Lastly, the work by Dr. Timur Dogan, Patrick Kastner, Remy Mermelstein, et al. named Eddy3D focuses on *"airflow and microclimate simulations for Rhinoceros and Grasshopper."* [29]. This project uses all the tools mentioned in the work of Layout5, in addition to applying cylindrical and box-shaped simulation environments. They used a similar workflow of feeding the machine learning model an input image and returning an output with a heatmap of the simulation results, in this case, focusing on urban ventilation potential and outdoor comfort . By incorporating the use of machine learning algorithms simply and effectively, it ends up lowering the learning curve for architects to use CFD simulations. The last two projects presented are still work in progress and are currently

open source. However, Dogan and Layout 5 are becoming pioneers in developing and integrating multiple environmental simulations into the architectural workflow more seamlessly and quickly.

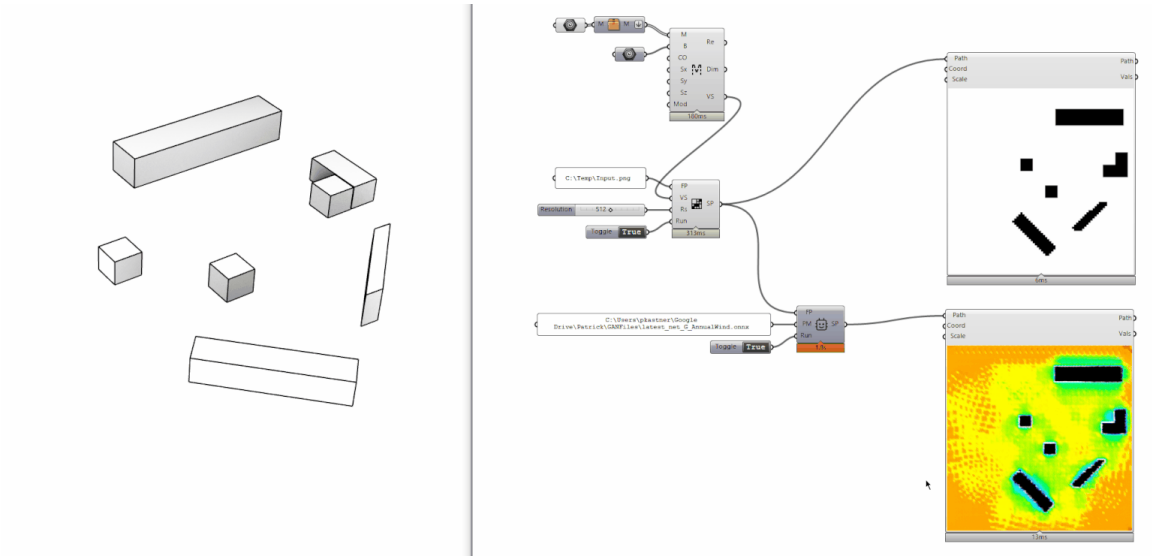


Figure 9: Machine Learning Development for Eddy3D using GAN & Pix2Pix [29]

Section 2.3 – Common Themes, Debates & Gaps:

Across the list of identified precedents, these present a series of common themes that have helped set the foundation blocks for this research. Primarily is the pursuit of improving the simulation and setup times in CFD simulations for architects to use, either with GAN or ANN. The use of these algorithms has proven to reduce analysis time by fractions of a second. Although many of these precedents hold a similar goal, there are specific key differences in selecting the best approach. As for the software used to implement these systems, Rhinoceros & Grasshopper appear to be the most used in architecture education and practice. As these are standard tools used by architects, this can help provide smoother workflow integrations for the design community.

ANN can be used more to integrate the different mathematical equations necessary to predict CFD simulations accurately. However, although this appears to be the proper way to incorporate machine learning algorithms into these kinds of simulations, they still hold the problem with the "*lack of performance guidelines."* Although it increases the accuracy of the prediction, more variables still increase the percentages of providing a wrong setup, increasing the learning curve.

Incorporating GAN appears to be a more popular option due to the more straightforward and less mathematical approach to the problem. This method mainly consists of generating a data set of images for the desired simulation output, in the case of CFD, which would be airflow movement. Use this data set to train the machine learning model to generate new predictions. This method would appeal more to the architectural profession as it is visually engaging. Also, using these kinds of algorithms appears to require less computational knowledge and is easier to deploy correctly. However, due to this solely based on images, the data set quality is a crucial step. Another discrepancy is the use of coupled vs. decoupled approach. Although it has been identified in some of the precedents that most of the CFD simulations are implemented in a decoupled exterior approach. However, the previous section of this study identified the inefficiency of the decoupled method compared to coupled methods.

Based on a thorough literature review, the author has identified an opportunity to develop machine learning CFD simulations to integrate into architectural practice successfully. To achieve this, multiple elements would be integrated from some of the precedents. There is a need to incorporate more coupled approach CFD model predictions for architects that do quick results and promotes exploration during those initial design

stages. To achieve its goals, the application must incorporate simple features to allow quick-rapid prototyping that gives architects a clear understanding of the models' airflow and pressure changes.

CHAPTER 4: METHODOLOGY

As an approach to the set of problems previously stipulated, this section will explore the use of supervised machine learning algorithms in combination with coupled airflow CFD analysis in architectural buildings. The process consists of three major steps: creating the data set, training the machine learning model, and evaluating the trained model results. The data set would be comprised of images. The used geometry was generated using parametric software, which would later pass through the CFD engine. For this study, only cross ventilation was studied on a two-dimensional plane. The engine to run the airflow simulations used in this study was Butterfly, a plug-in for Grasshopper from the Ladybug Tools set developed by Mostapha Sadeghipour [30]. This tool runs CFD simulations using OpenFOAM. When making this study, OpenFOAM was the most validated open-source engine for running advanced simulations [30]. The machine learning model was based on the Pix2PixHD framework developed by NVIDIA, which runs using the pyTorch library [31]. After obtaining the resulting images, these were compared alongside the butterfly engine results.

Section 3.1 - Data Set Creation:

Creating the data set consists of three major stages: form generation, simulation run, and post-processing. These steps have been performed in a combination of Rhinoceros, Grasshopper, and OpenFOAM. The form generation step is made inside Grasshopper, beginning with the generation of a 3.3m x 3.3m square for the base plan, offsetting 0.1m walls on all sides and extruding 2m in height. After the generation of the walls, panoramic windows were created, which were later subdivided into three sections, providing 12

windows for this scenario. To limit the number of possible variations, these were reduced to combinations that contained between two and four windows. As a result, this leads to a total of 781 possible variations. Based on the selected combination, these were extruded and cut out of the walls. Finally, a horizontal plane was created at the building's mid-height to perform the analysis. Another applied constant was the wind speed at 4.5m/s and its direction, coming from the south.

Before getting to the second stage, it is important to mention that these simulations can be performed at different grid densities, severely affecting performance and accuracy. Multiple variations were studied using the same model to identify the optimal number, evaluating both accuracy and run time. Five densities were evaluated ranging from 1.0, 0.5, 0.3, 0.2 to 0.1 meters. Based on the obtained results, 0.2 was determined to be the best alternative as it provided results far more accurate than the coarse alternatives and similar to the finer ones. Run times resulted in an average of 64 seconds per simulation for the 0.2 density, while the finer 0.1 density had an average of 206 seconds.

The second phase, the simulation run stage, is performed in two steps, recipe generation and execution. Once all these final iterations are generated, it then passes through the butterfly engine, which will write a specific recipe for the OpenFOAM engine to understand the required steps to run the simulation stored on an external folder. After all, the desired recipes are written for all the possible combinations, using an automated batch python script inside the OpenFOAM terminal to automatically run all the simulations, one at a time, which automatically runs the next one after the first one is finished.

Grid Density	Min (seconds)	Max (seconds)	Average (seconds)
0.1	206	257	219
0.2	59	72	64
0.3	17	27	19
0.5	9	12	10
1	5	14	7

Table 2: Grid Density Time Comparison

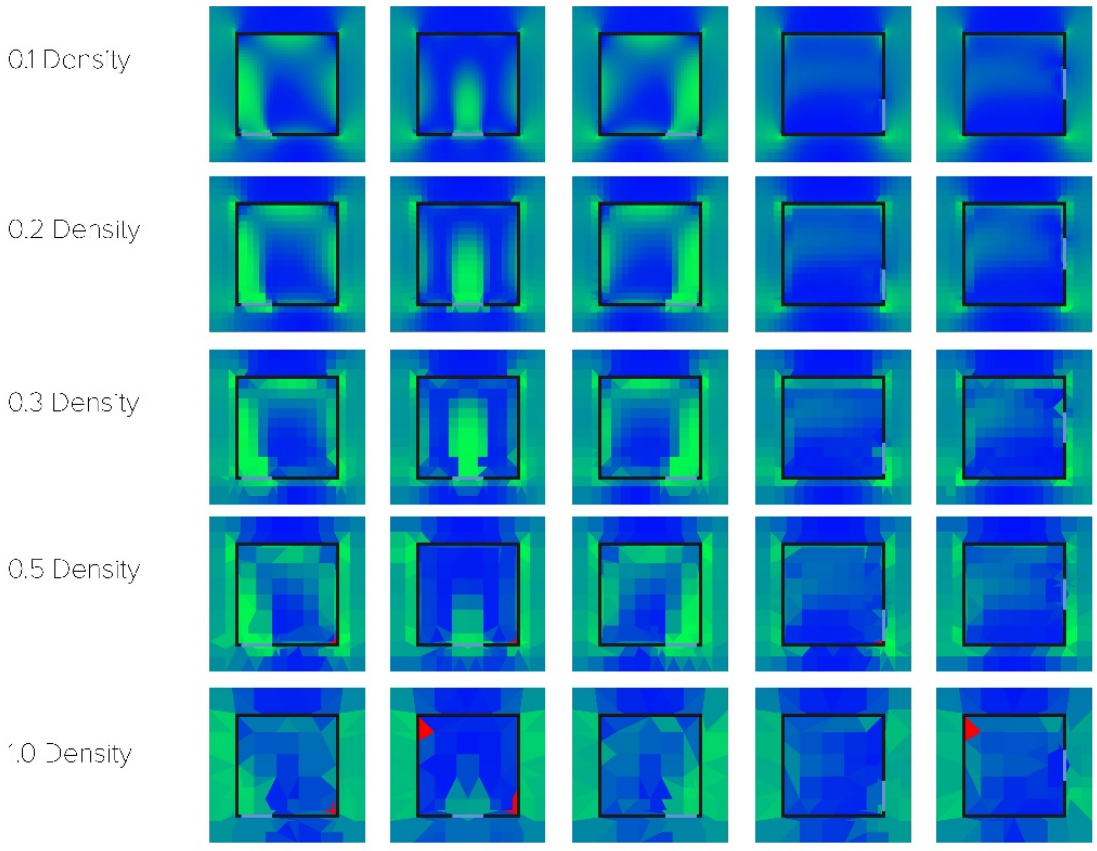


Figure 10: Grid density variations

The third stage is post-processing, performed inside Rhinoceros and Grasshopper. This stage consisted of reloading the original three-dimensional model identified in the simulation name and loading the butterfly recipe back into the Grasshopper, which contained the visualized simulation results. Once both were loaded, a 2048px x 2048px image was captured of the building's plan view alongside the airflow simulation results and stored in an external folder. These will be the inputted data for the machine learning model. Besides the CFD map images, an image without the simulation outcome was also captured to serve as a base input.

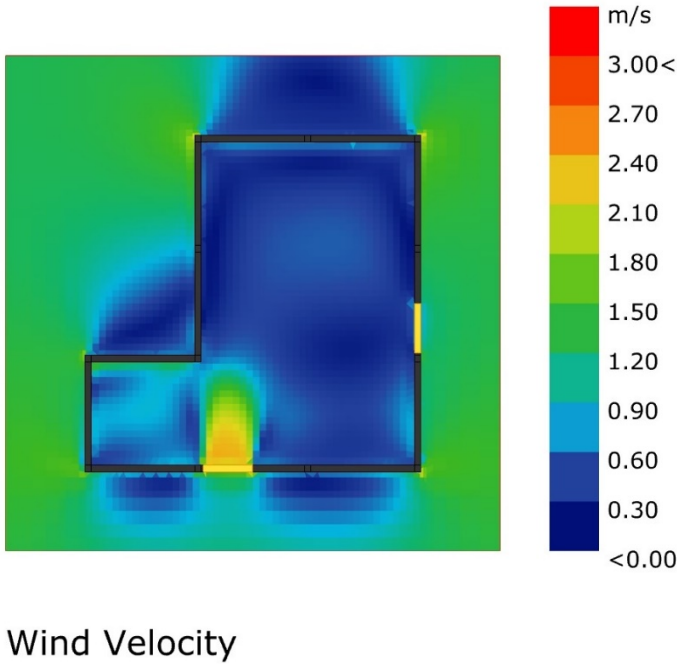


Figure 11: Post Processing Visualization results within Rhinoceros / Grasshopper

After all the data was collected, it was split into two groups, testing, and training. This division was done at an 80/20 ratio (training, testing) to use the same dataset to train and test the model without generating an additional data set. The purpose of using an 80/20

ratio as opposed to a 60/40 was to provide the machine learning model with a larger data set to train as it increases the model's accuracy.

Section 3.2 - Machine Learning & Training:

For the machine learning model, the Pix2PixHD by NVIDIA was used as a base. This framework uses the CycleGAN technique, which automatically trains the image-to-image translation models without using paired examples—instead of using a collection of images, having a source and a target, which do not need to be related. Multiple models were trained during this stage to identify how manipulating hyperparameters would affect the model's efficiency when predicting airflow patterns.

The training parameters were the following: the data set contained no labels, epoch checkpoints saved in increments of five, images were not cropped or resized, images were flipped, and loaded at a resolution of 512px by 512px. Two discriminators were used to reduce the possibility of overfitting. The main parameter altered was the batch size, tested in 1, 3, and 5 variations. In addition, the dataset was modified by adding additional padding to the images. After setting these parameters, the models began training with a mixture of using Google's Colab and local resources. All models were trained for 100 epochs and took an average of eight hours to reach the 100-epoch mark.

Section 3.3 – Data Evaluation

As other machine learning models are trained with a loss function until convergence, evaluating GAN's results is not straightforward [32]. Given that no objective loss function is used to train the GAN generator, there are no objective assets to measure its development and progress. As a result, given that GAN's are image-based, the best

method to determine their efficiency is a visual inspection. As visual inspections can be subjective, therefore a more objective visual approach was used.

This approach consists of measuring the similarity between the image produced by the GAN model and the one produced by the Butterfly engine. These images were compared using two techniques within a python script using the PIL and cv2 image libraries. The ORB (Oriented FAST and Rotated BRIEF) feature matching technique identifies different characteristics within the images to compare them. This method was selected compared to others as it is open source, requires less computational cost [33].

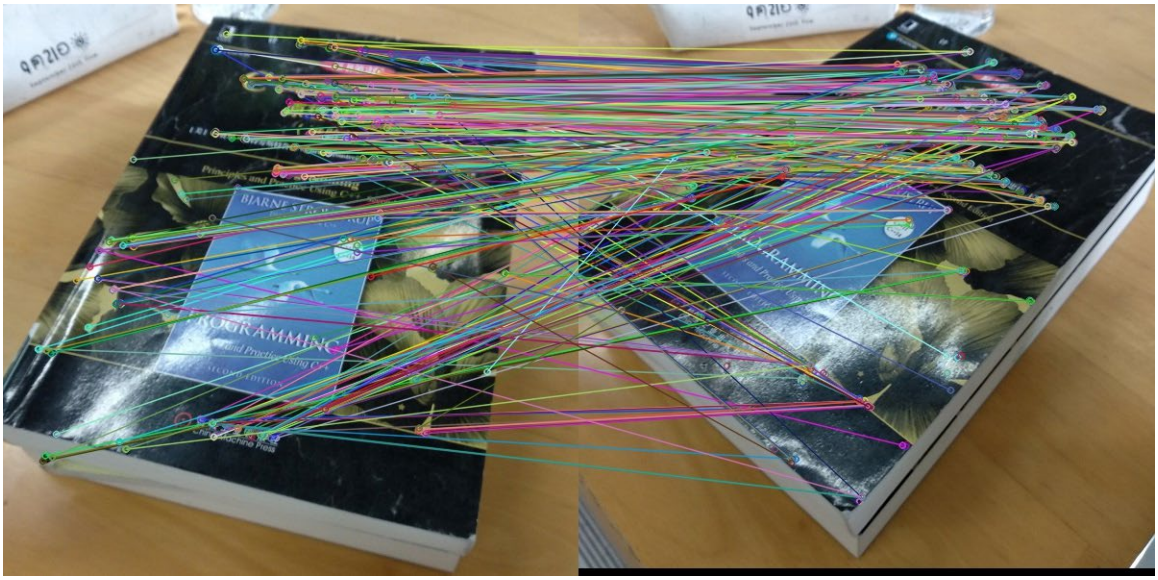


Figure 12: ORB Feature Detection Example [33]

In addition to this method, images were evaluated by the use of structural similarity (SSIM), which compares images on three main metrics: luminance, contrast, and structure [34] [35]. This process takes a more holistic approach by evaluating images by looking at a group of pixels instead of a pixel to pixel-based to determine their similarity, similar to how humans would perceive images. Both evaluation methods return a number from 1 to

-1, the highest indicting the most resemblance between the images and lower indicating the latest difference between the images. As these metrics evaluate images differently, the average from both image similarity evaluation methods was calculated to obtain an objective result. These methods were applied at different training stages of each model to examine their learning process and how each of the different parameters affected its learning curve.

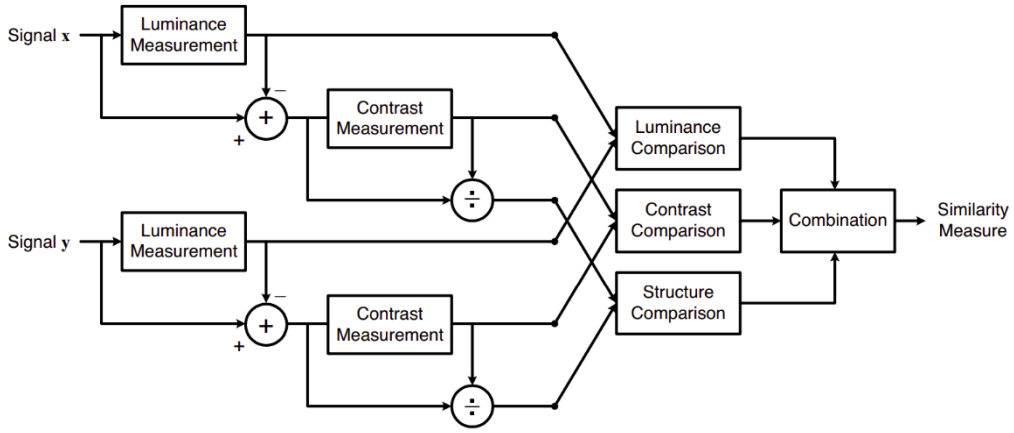


Figure 13: The Structural Similarity Measurement System [34]

In addition to comparing the image similarity between the image produced by the GAN model and the one produced by the Butterfly engine, run times were evaluated. To properly evaluate run times, custom python scripts were created to record the machine learning model and the CFD engine run times. As computer hardware can significantly impact the speed of these simulations, the same computer was used for both tests. The computer used for testing contains the following specification: 12 core Ryzen 3900XT, 32 GB of RAM, and 8GB of dedicated GPU memory.

CHAPTER 5: RESULTS AND DISCUSSION

After generating the data set and training the machine learning models, evaluated results tested the proposed hypothesis of using GAN to obtain relatively accurate results for CFD airflow simulations. Multiple models were tested at different training intervals to observe how their training progressed. After identifying the most effective model, this one was tested using a different floor plan layout to observe how the model would react to completely unknown images and compare it with the initial data set.

5.1 Initial & Padded Data Set:

In order to evaluate the previously explained methodology, a test was conducted, consisting of a single square floor plan layout, modifying only the window configurations, using a data set of 624 images to train the machine learning model for 100 epochs. After finishing the training phase, the model was tested using 15 unknown images. To the naked eye, images would appear almost identical. However, after evaluating the structural similarity, the images average around 95% compared to their CFD simulation counterparts. When evaluating run times. the traditional CFD simulations took an average of 62 seconds

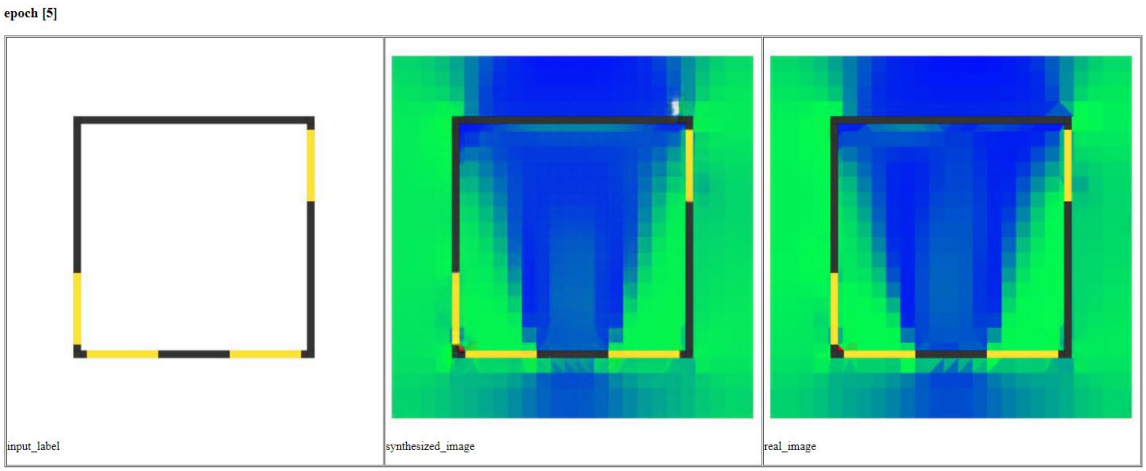


Figure 14: Initial Dataset Training Results at Epoch 5

to complete, while its machine learning model took an average of 0.2 seconds. These results present a clear indication of the efficiency of GAN to generate images. The images produced indicated that these kinds of machine learning models can be applied to CFD airflow simulations while still maintaining a relatively high level of accuracy. However, these results presented a problem with the speed at which the model was learning to produce accurate results. Even within the first couple of epochs, the model produced substantially accurate results, raising questions of the efficiency of the data set or the hyperparameters used.

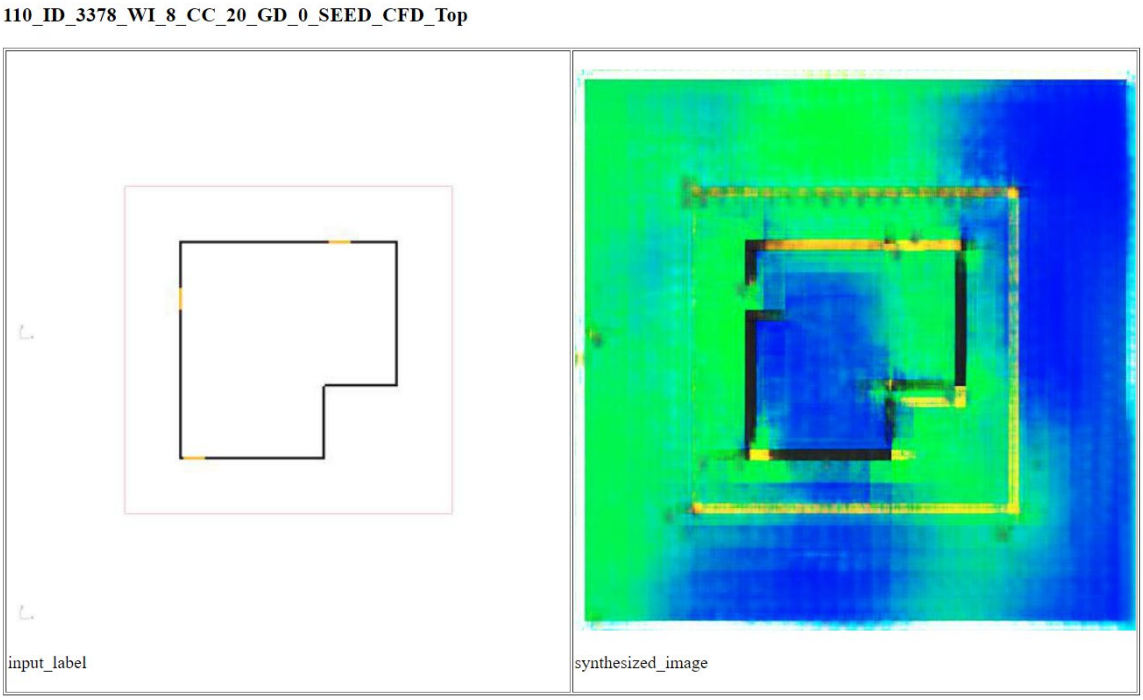


Figure 15: Initial Dataset Test Result with Different Floor Plan Layout

After evaluating these concerns, it was identified that the model had memorized specific pixels of the images from the data set, causing rapid and accurate results. These images all contained the floor plan centered, which helped the model predefine its location in further iterations. Even after flipping the images, they presented no significant change

in the training set, given them being symmetrical. This hypothesis was confirmed by inputting a different set of floor plans to the model, which produced unsatisfactory and inaccurate results. These results showed how the model tried to generate the square floor plan in the exact location, and all the walls would be of the same thickness.

In order to correct this issue, a new dataset was created based on the initial data set. This new data set consisted of adding padding around the existing images to change the scales of the floor plans while maintaining the same square aspect ratio. This shift helped avoid having the floor plan always centered on the image and at the same scale, providing an additional level of complexity for the model to predict.

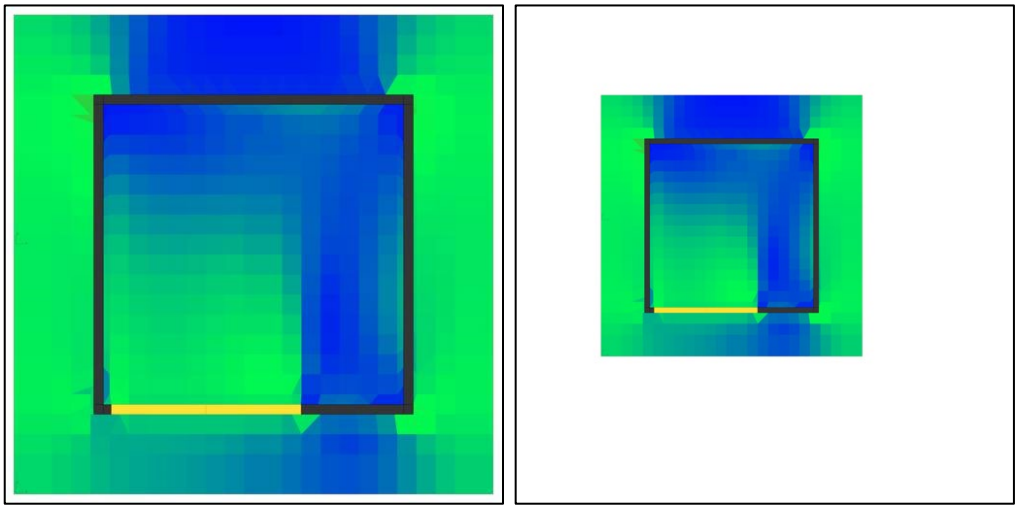


Figure 16: (Left) Initial Dataset Image / (Right) Padded Dataset Image

As a next step, three different models were trained for 100 epochs using 624 images of this new data set, changing their batch size to observe how it would affect the training process of the models. Batch size refers to the number of images the model will use during its training in each iteration or epoch. The batch sizes used were 1, 3, and 5. None of the presented hardware had the capabilities to incorporate larger batch sizes. After training all

these models, they were tested at different epoch checkpoints to see the development of the images as the model continued training. The set training checkpoints selected were 5, 25, 50, 75, and 100 epochs. The purpose of choosing these intervals was to examine the model at four different stages of its training. The number 5 checkpoint was selected to compare the models learning results with the initial data set model. For these testing, the previous two similarity methods were applied (ORB & SSIM), in addition to their run times.

5.2 Padded Data Set Results:

As a result of the training of the different machine learning models, visually, all models began their training producing similar results. The model with a batch size of 1, however, showed a slightly sharper image. However, after passing the 25 epoch mark, all three models began to produce images with an average of 90 percent similarity with the original CFD images, producing relatively accurate results. The model with a batch size of 3 produced the most accurate and consistent results at the end of the training phase, followed by the model with a batch size of 1, and lastly, the model with a batch size of 5. This last model appeared to have trouble predicting images even after the 100 epoch mark. One of the images obtained one of the lowest scores of 79 percent similarity when compared against their CFD counterpart.

These machine learning models were tested using a set of five images. The images were similar as they used the same square floor plan with different window combinations and different image compositions. A large gap was observed regarding run times when comparing the machine learning models with the CFD engine. It took on average 60 seconds to perform an analysis on the CFD engine. In contrast, the machine learning models performed their prediction on an average of 0.12 seconds, accounting for a

difference 600 times faster than the CFD engine when presented similar data. However, this difference is to be expected, as similar results were observed in the work of Layout 5 and Kacper Radziszewski & Marta Waczyńska in their daylight analysis. However, in all the tests, when observing run times, it indicated that the first image to go through the machine learning model takes on average 1.38 seconds compared to the other ones. It takes that time to generate the models and prepare the neural network. After the first image, the following run times were of fractions of a second. Observing these results, the model with a batch size of 3 proved to be the most effective for this task due to producing the most accurate and consistent results. Nonetheless, this emphasizes the efficiency of machine learning models and supports the arguments made in this study to help designers obtain relatively accurate CFD airflow predictions in less time.

Figure 17: Epoch & Batch Size Graph in Comparison in Relation to Image Similarity

Time [Seconds]

	Layout 1	Layout 2	Layout 3	Layout 4	Layout 5
Simulation	63	64	61	63	60
GAN	0.13	0.12	1.53	0.12	0.12
Difference	62.87	63.88	59.47	62.88	59.88

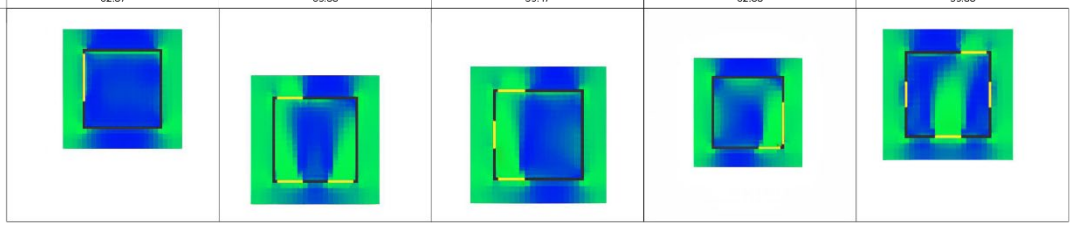


Figure 19: Run Time Comparison Between CFD Simulations and GAN

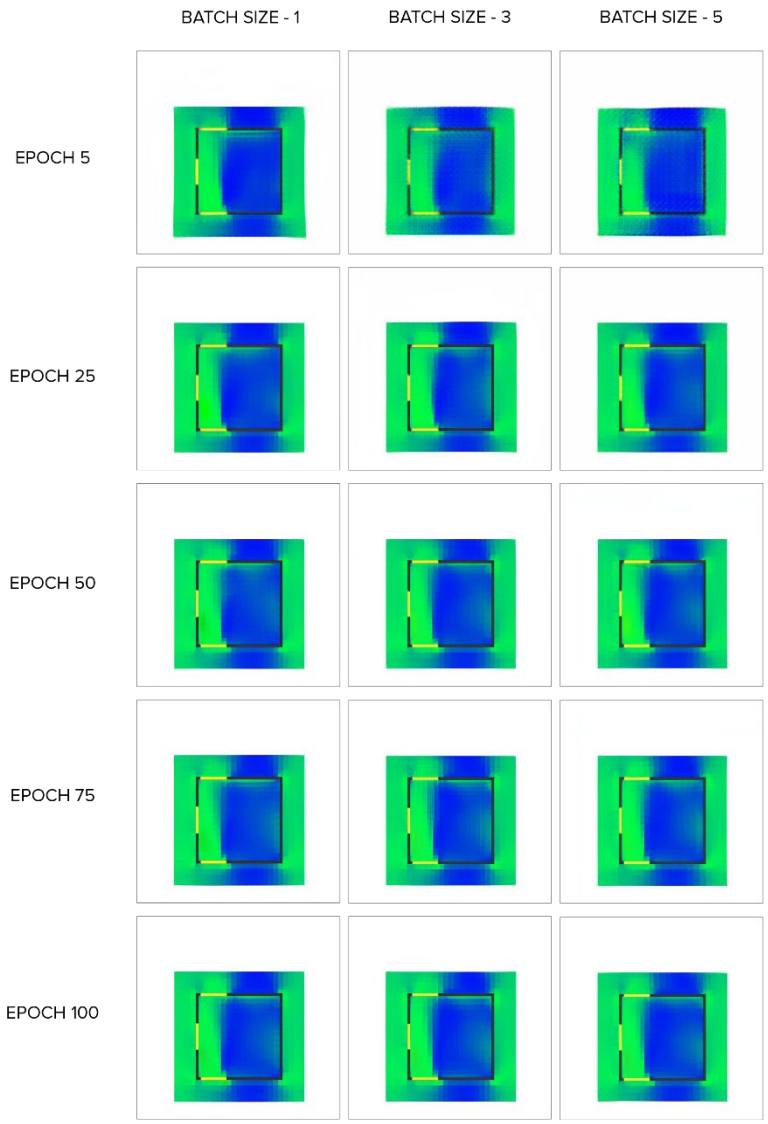


Figure 18 : Epoch & Batch Size Training Comparison

5.3 Rectangular Floor Plan Typology Test & Results:

As with these machine learning models, they will eventually be exposed to unknown data and generate predictions based on their trained data. Based on the results from the previous tests, the model with the best performance was the one with a batch size of 3. This model, alongside the original model trained with the initial data set, was exposed to a rectangular floor plan to test their efficiency when presented with different data. Five different window combinations were selected on a rectangular floor plan to provide different scenarios for this test. Given that the original model was not trained on images that contained padding as the second model, both were tested with a set of five images with padding and without padding.

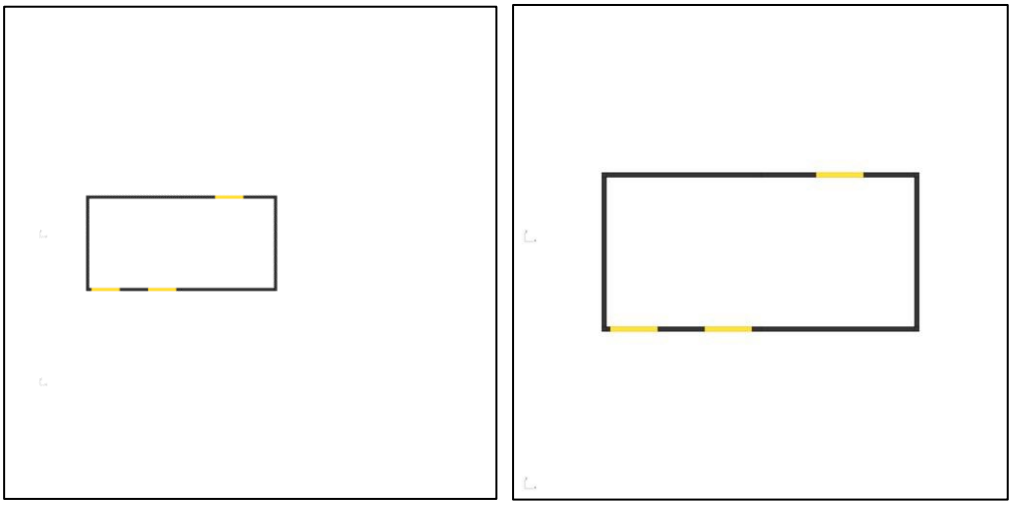


Figure 20: Example of Rectangular floor plan (Left) Padded Image / (Right) Non-Padded Image

As a result of this test, the original machine learning model performed poorly, as it could not generate satisfactory results of the airflow pattern for the rectangular floor plans. However, it performed marginally better when presented the non-padded images, as in this case, even though the airflow patterns are unclear, a floor plan layout was defined. The machine learning model trained on the padded images outperformed by a large margin the

original model. It produced relatively accurate results of the airflow patterns on both the padded and non-padded images. When comparing the generated images with their simulation counterparts, the original model obtained an average of 39% image similarity compared to the padded model with an average of 79%, indicating a difference of 40%.

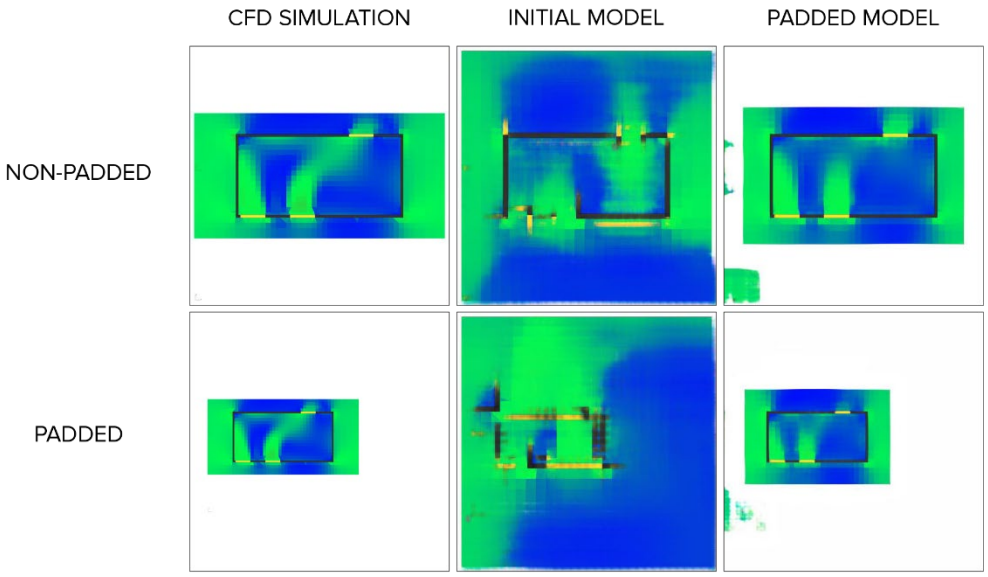


Figure 21: Test Results of Rectangular Floor Plan on Both Initial and Padded Machine Learning Model

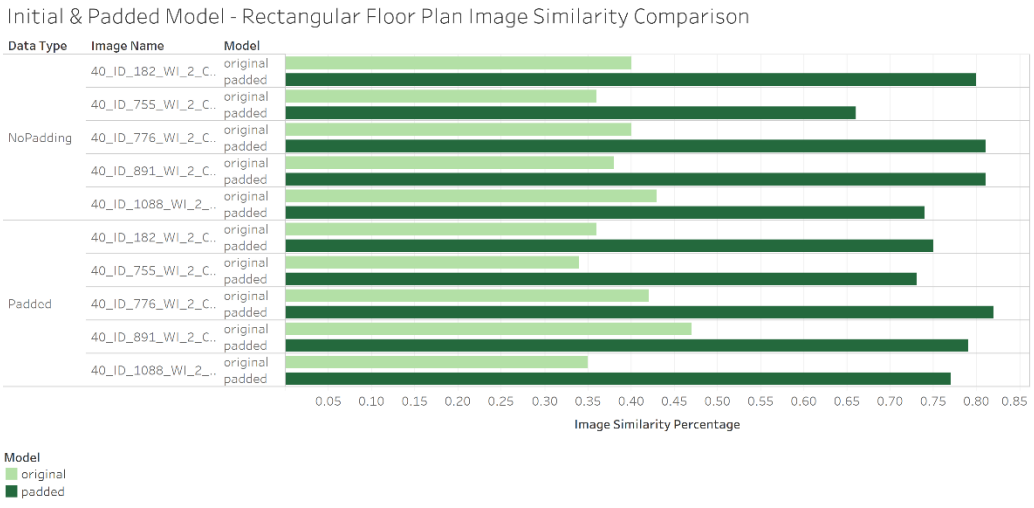


Figure 22: Initial & Padded Machine Learning Model for Rectangular Plan Image Similarity Comparison

CHAPTER 6: CONCLUSION & FUTURE WORK

The test result obtained in this research indicates that GAN application for CFD airflow predictions can produce acceptable results for designers to allow them to advance their design process through an alternate method with marginal time differences compared to traditional CFD simulations. Results show a significant run time difference of an average of 60 seconds between the CFD simulation and the machine learning model. The GAN models generated predictions 600 times faster on both similar and unknown data, generating relatively accurate results with a range between 79 to 95 percent image similarity with the CFD simulation output. While seconds to a minute difference might not appear significant, minutes can grow exponentially to hours and even days when performing complex CFD simulations. Therefore, these time differences indicate a significant cut in run times. However, as with most machine learning models, the data set is a crucial element for the successful performance of the models.

When observing both coupled and decoupled approaches for GAN, the coupled model presents an easier path for training. Focusing on exterior airflow simulations, the GAN model would only need to detect filled shapes, which might require a smaller data set to train and produce accurate results. On the other hand, decoupled models require both a clear boundary and apertures, increasing the complexity as more boundary forms begin to be introduced, requiring an even larger data set to property train the models and obtain accurate results.

Alternating different hyperparameters, such as adding more discriminators or increasing the batch size, might help produce more accurate results when using a larger

data set. However, one of the major factors for a successful GAN model is the data set used for training. As observed in this study, the padded data set provided more varied and diverse information when compared to the initial data set—resulting in a 40% increase in image similarity to the actual CFD simulation, even when tested with unknown floor plan layouts. The variations referred to do not only account for the floor plan and window configurations themselves but also image composition—alternating parameters such as floor plan location, scale, and orientation within the image itself.

This project aims to expand the floor plan typologies to more complex forms and window configurations for future work. Alternatively, this research aims to explore other machine learning algorithms such as linear regression and other ANN and compare them alongside GAN to identify the most efficient machine learning algorithm for the desired CFD airflow simulation. The long-term goal is to convert the trained machine learning model into a Grasshopper and Dynamo plug-In. After implementing the plug-in, user studies would be conducted with architects to observe how this system would help them within their design workflow and implement them within their practice. This study is a small starting point for the application of GAN for CFD simulations. The current goal is to provide additional insight for future researchers to continue developing the use of GAN for different CFD simulations and incorporate them into architectural workflows.

REFERENCES:

[1] N. A. Aldossary, Y. Rezgui, and A. Kwan, "Domestic energy consumption patterns in a hot and humid climate: A multiple-case study analysis," *Applied Energy*, vol. 114, pp. 353–365, Feb. 2014, doi: 10.1016/j.apenergy.2013.09.061.

[2] C. Ghiaus and C. Inard, *Energy and environmental issues of smart buildings*.

[3] A. Katili, R. Boukhanouf, and R. Wilson, "Space Cooling in Buildings in Hot and Humid Climates – a Review of the Effect of Humidity on the Applicability of Existing Cooling Techniques," Aug. 2015. doi: 10.13140/RG.2.1.3011.5287.

[4] M. Donn, "Quality assurance – simulation and the real world," 1999.

[5] G. M. Abbas and I. Gursel Dino, "A parametric design method for CFD-supported wind-driven ventilation," Oct. 2019, vol. 609. doi: 10.1088/1757-899X/609/3/032010.

[6] S. J. Emmerich, W. S. Dols, and J. W. Axley, "Natural ventilation review and plan for design and analysis tools," National Institute of Standards and Technology, Gaithersburg, MD, NIST IR 6781, 2001. doi: 10.6028/NIST.IR.6781.

[7] "Lawson Wind Comfort Criteria: A Closer Look | SimScale Blog," *SimScale*, Jan. 17, 2020. https://www.simscale.com/blog/2020/01/lawson-wind-comfort-criteria/ (accessed Dec. 18, 2020).

[8] T. Stathopoulos, "Wind and Comfort," Jul. 2009. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.458.268&rep=rep1&type=pdf (accessed Dec. 18, 2020).

[9] S. Oh, "Wind on Our Terms: Wind Tunnel Technology – LG HVAC STORY." https://www.lghvacstory.com/wind-on-our-terms-wind-tunnel-technology/ (accessed Apr. 05, 2021).

[10] J. Blazek, "Chapter 1 - Introduction," in *Computational Fluid Dynamics: Principles and Applications (Second Edition)*, J. Blazek, Ed. Oxford: Elsevier Science, 2005, pp. 1–4. doi: 10.1016/B978-008044506-9/50003-5.

[11] "Navier-Stokes equation | Definition & Facts," *Encyclopedia Britannica*. https://www.britannica.com/science/Navier-Stokes-equation (accessed Dec. 15, 2020).

[12] S. Team, "3 Core Components of CFD Analysis." https://blog.spatial.com/cfd-analysis (accessed Apr. 20, 2021).

[13] M. Patel, "Typical Steps of CFD Simulation Process, Computational Fluid Dynamics (CFD) | HiTechCFD.com," Jun. 11, 2013. https://www.hitechcfd.com/cfd-

knowledgebase/seven-stages-of-a-typical-cfd-simulation.html (accessed Apr. 20, 2021).

[14] R. Ramponi and B. Blocken, “CFD simulation of cross-ventilation for a generic isolated building: Impact of computational parameters,” *Building and Environment*, vol. 53, pp. 34–48, Jul. 2012, doi: 10.1016/j.buildenv.2012.01.004.

[15] O. S. Asfour and M. B. Gadi, “A comparison between CFD and Network models for predicting wind-driven ventilation in buildings,” *Building and Environment*, vol. 42, no. 12, pp. 4079–4085, Dec. 2007, doi: 10.1016/j.buildenv.2006.11.021.

[16] J. M. Horan and D. P. Finn, “Sensitivity of air change rates in a naturally ventilated atrium space subject to variations in external wind speed and direction,” *Energy and Buildings*, vol. 40, no. 8, pp. 1577–1585, Jan. 2008, doi: 10.1016/j.enbuild.2008.02.013.

[17] G. Carrilho da Graça, Q. Chen, L. R. Glicksman, and L. K. Norford, “Simulation of wind-driven ventilative cooling systems for an apartment building in Beijing and Shanghai,” *Energy and Buildings*, vol. 34, no. 1, pp. 1–11, Jan. 2002, doi: 10.1016/S0378-7788(01)00083-4.

[18] Serokell, “Artificial Intelligence vs. Machine Learning vs. Deep Learning: What’s the Difference,” *Medium*, Nov. 23, 2020. https://medium.com/ai-in-plain-english/artificial-intelligence-vs-machine-learning-vs-deep-learning-whats-the-difference-dccce18efe7f (accessed Dec. 18, 2020).

[19] S. Raschka and V. Mirjalili, *Python Machine Learning*. Packt Publishing Ltd, 2017.

[20] J. Brownlee, “What is Deep Learning?,” *Machine Learning Mastery*, Aug. 15, 2019. https://machinelearningmastery.com/what-is-deep-learning/ (accessed Apr. 05, 2021).

[21] Simplilearn, *Deep Learning In 5 Minutes | What Is Deep Learning? | Deep Learning Explained Simply | Simplilearn*, (Jun. 03, 2019). Accessed: Apr. 05, 2021. [Online Video]. Available: https://www.youtube.com/watch?v=6M5VXKLf4D4&ab_channel=Simplilearn

[22] J. Brownlee, “A Gentle Introduction to Generative Adversarial Networks (GANs),” *Machine Learning Mastery*, Jun. 16, 2019. https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/ (accessed Dec. 18, 2020).

[23] J. Hui, “GAN — What is Generative Adversarial Networks GAN?,” *Medium*, Dec. 26, 2019. https://jonathan-hui.medium.com/gan-whats-generative-adversarial-networks-and-its-application-f39ed278ef09 (accessed Dec. 18, 2020).

[24] “The Past, Present, and Future of Cloud Computing and CFD,” *SimScale*, May 05, 2020. https://www.simscale.com/blog/2020/05/future-of-cloud-computing-cfd-engineers/ (accessed Dec. 18, 2020).

[25] K. D. Foote, “A Brief History of Cloud Computing,” *DATAVERSITY*, Jun. 22, 2017. https://www.dataversity.net/brief-history-cloud-computing/ (accessed Dec. 18, 2020).

[26] K. Radziszewski and M. Waczyńska, “Machine Learning Algorithm-Based Tool and Digital Framework for Substituting Daylight Simulations in Early-Stage Architectural Design Evaluation,” presented at the 2018 Symposium on Simulation for Architecture and Urban Design, Delft, Netherlands, 2018. doi: 10.22360/SimAUD.2018.SimAUD.001.

[27] C. Ding and K. P. Lam, “Data-driven model for cross ventilation potential in high-density cities based on coupled CFD simulation and machine learning,” *Building and Environment*, vol. 165, p. 106394, Nov. 2019, doi: 10.1016/j.buildenv.2019.106394.

[28] “TheodoreGalanos/DaylightGAN,” *GitHub*. https://github.com/TheodoreGalanos/DaylightGAN (accessed Nov. 12, 2020).

[29] T. Dogan, R. Mermelstein, and P. Kastner, “Eddy3d.” https://www.eddy3d.com/index.html#about (accessed Dec. 18, 2020).

[30] “Ladybug Tools | Butterfly.” https://www.ladybug.tools/butterfly.html (accessed Dec. 19, 2020).

[31] *NVIDIA/pix2pixHD*. NVIDIA Corporation, 2021. Accessed: Apr. 05, 2021. [Online]. Available: https://github.com/NVIDIA/pix2pixHD

[32] J. Brownlee, “How to Evaluate Generative Adversarial Networks,” *Machine Learning Mastery*, Aug. 25, 2019. https://machinelearningmastery.com/how-to-evaluate-generative-adversarial-networks/ (accessed Apr. 05, 2021).

[33] phpnewbie1979, “ORB Feature Point Extraction Algorithm Test,” *Programmers Think*, May 09, 2019. https://programmer.ink/think/orb-feature-point-extraction-algorithm-test.html (accessed May 10, 2021).

[34] P. Datta, “All about Structural Similarity Index (SSIM): Theory + Code in PyTorch,” *Medium*, Mar. 04, 2021. https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e (accessed Apr. 05, 2021).

[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessment: From Error Visibility to Structural Similarity,” *IEEE Trans. on Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004, doi: 10.1109/TIP.2003.819861.