



A New Interoperability Framework for Data-Driven Building Performance Simulation

Citation

Han, Jung Min. 2022. A New Interoperability Framework for Data-Driven Building Performance Simulation. Doctoral dissertation, Harvard Graduate School of Design.

Permanent link

<https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37372963>

Terms of Use

This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA>

Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. [Submit a story](#).

[Accessibility](#)

HARVARD UNIVERSITY
Graduate School of Design



THESIS ACCEPTANCE CERTIFICATE

The undersigned, appointed by the Doctor of Design Program, have examined a dissertation entitled

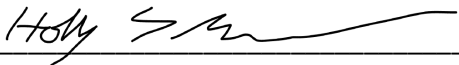
A New Interoperability Framework for Data-Driven Building Performance Simulation

Presented by
Jung Min Han


candidate for the Doctor of Design degree and hereby certify that it is worthy of acceptance.

Signature  _____

Professor Ali Malkawi

Signature  _____

Professor Holly Samuelson

Signature  _____

Pavlos Protopapas, Scientific Program Director and Lecturer, SEAS

Date: December 9, 2021

A New Interoperability Framework for Data-Driven Building Performance Simulation

A dissertation presented

by

Jung Min Han

to

Harvard University Graduate School of Design

in partial fulfillment of the requirements

for the degree of

Doctor of Design

Harvard University

Cambridge, Massachusetts

May 2022

Copyright © 2022 by Jung Min Han. All rights reserved.

This dissertation is dedicated to my parents.

For their endless love, support, and encouragement

Advisor
Ali Malkawi

Author
Jung Min Han

A New Interoperability Framework for Data-Driven Building Performance Simulation

Abstract

Machine learning (ML) and deep learning (DL) have become more prominent in the building, architecture, and construction industries. One area ideally suited to exploit this powerful new technology is building performance simulation (BPS) for sustainable building design. Physics-based models have traditionally been used to estimate the energy flow, air movement, and heat balance of buildings. The algorithms behind physics-based models, however, involve solving complex differential equations that require many assumptions, significant computational power, and a considerable amount of time to output predictions. With the advent of DL, which can handle large amounts of computation in a short period of time, data-driven models for predicting the physical properties of buildings are becoming increasingly popular due to their simplicity and efficiency. As such, artificial neural networks (ANNs) with measured or simulated data for environmental analysis are likely to be a more feasible option for designers during the early design phase.

To train ANN models, 3D data is an asset to computer vision because they provide rich information about the geometry and the related environment. Depending on the 3D data representation considered, different challenges may emerge when using trained ANN models.

Hence, an interoperability framework is required for converting building geometries and environment-related information into relevant 3D matrices for model training and utilization. However, to date there has been no research on this topic in the BPS field; thus, this research proposes a new data interoperability framework for ANN models with 3D buildings serving as inputs. The framework has been subjected to a trial investigation using several ANN modeling studies on radiation and airflow simulation. The result is a comprehensive process map that includes the BPS requirement for ANN modeling, related subprocesses (i.e., building geometry and environmental levels), specific rules and methods for modeling, and processing of input and output data. To accomplish this, data exchangers for the ANN models, geometry representation tool (GRT), and BIM specification tool (BST) were introduced and developed as computational tools. The comprehensive framework has been validated using the developed case studies, demonstrating its applicability for different Computer-aided design tools (i.e., Rhinoceros and Revit) and ANN models (i.e., solar radiation and airflow) and illustrating the future capacity of integrated ANNs to serve as a tool for use in BPS and early-stage modeling.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor, Ali Malkawi, for his enthusiasm, inspiration, and guidance throughout my journey. You were always open to new ideas and available when I needed help, regardless of the time or day. Your encouragement and sharp intuition motivated me to think deeper and make the best effort possible. I will never forget your sincere advice and assistance.

I would also like to thank Holly Samuelson and Pavlos Protopapas for being my committee members. I have benefited greatly from your constructive feedback and kind support of my research. Holly, you were a great mentor, and your thoughtful advice led me to become a better educator. Yoon Kim also gave useful practical advice, particularly when I was developing deep learning models for this dissertation. I thank you for your support and guidance.

More broadly, I am thankful to my many mentors and collaborators: Christoph Reinhart, Krzysztof Z. Gajos, Martin Bechthold, Volker Hartkopf, H-Sang Seung, Sunwoo Park, Daekwon Park, Nari Yoon, Adrian Chong, Yuqian Ang, Bin Yan, and Wentao Wu. Thank you also to my colleagues at the GSD and the CGBC for inspiring me with your incredible talent and passion.

Special thanks to my sister, Jung Won, for your support and love throughout my time in the US. Because of you, I could eat, sleep and smile during this difficult time in my life. I would also like to thank my grandparents. My grandfather, I wish you were here to share it with you.

Most of all, I would like to thank my parents, who always trusted my decisions, showed me unconditional love, and prayed for me. Doctor Han and Professor Oh, this dissertation would not have been possible without your love and support.

Contents

Title Page	I
Abstract	IV
Acknowledgements	VI
List of Figures	X
List of Tables	XIV
Chapter 1 Introduction	1
1.1 Research Questions.....	3
1.2 Objectives.....	7
Chapter 2 Literature review	8
2.1 History of software interoperability and framework.....	9
2.1.1 History of buildingSMART and the industrial foundation class	9
2.1.2 Building Information Modeling	13
2.1.3 Building Performance Simulation	15
2.1.4 History of Building Performance Simulation (BPS) interoperability	16

2.1.5 Interoperable data format	21
2.2 Building performance simulation (BPS) and data-driven modeling.....	24
2.2.1 Surrogate modeling in as the alternatives of physics-based models.....	25
2.2.2 Artificial Neural Networks (ANNs) for BPS modeling.....	27
2.2.3 ANNs model architecture.....	29
2.3 Survey of computer vision tasks for 3D data representation.....	31
2.3.1 History of computer vision tasks.....	31
2.3.2 Data representation in computer vision	33
2.3.3 DL architecture for 3D vision tasks.....	39
2.3.4 3D data representation for BPS tasks	45
Chapter 3 Methodology	47
3.1 Development of new BPS workflow with ANN model	47
3.2 Development of data exchange methods for new BPS workflow	51
Chapter 4 Experiments.....	55
4.1 Development of ANNs for different BPS tasks	55
4.1.1 ANN-driven solar radiation simulation.....	55
4.1.2 ANNs-driven Airflow Simulation	76
4.2 Algorithms for 3D data exchangers	90
4.2.1 Robust voxelization	91
4.2.2 Surface and mesh generation.....	92
4.2.3 Discussion and Findings	94

Chapter 5 Interoperability Framework.....	97
5.1 ANN and data modeling for different BPS tasks	99
5.2 Development of data exchangers for different BPS tasks	100
5.2.1 Data exchangers for solar radiation simulation	103
5.2.2 Data exchangers for airflow simulation	107
Chapter 6 Validation.....	111
6.1 Python packages and software integration	113
6.2 Discussion	115
Chapter 7 Conclusion.....	116
Appendix	
Bibliography	

List of Figures

Figure 1.1: Time–cost exchange in different design stages.....	2
Figure 1.2: Early design decision-support tool development for collaborative working environments	3
Figure 1.3: Physics-based data-driven models and the requirement of domain knowledge.....	6
Figure 2.1: The history of Interoperability and future direction	20
Figure 2.2: From BIM to BEM and Web.....	23
Figure 2.3: 3D data representation of training data-driven models	34
Figure 3.1: Traditional framework for BPS design-consulting process	48
Figure 3.2: Proposed framework for BPS design-consulting process with ANN models.....	49
Figure 3.3: Inputs for modeling different performance assessment tools	50
Figure 3.4: Comparison of BIM and BPS frameworks.....	51
Figure 4.1: Generic workflow and related software.....	57
Figure 4.2: Binary padded voxel matrix (ARINet).....	58
Figure 4.3: 3DCNN architecture of ARINet.....	60
Figure 4.4: Reference geometries for the completely new buildings with boundaries.....	61
Figure 4.5: Analysis and comparison results (shading).....	62

Figure 4.6: Analysis and comparison results (round).....	63
Figure 4.7: Analysis and comparison results (inner core)	63
Figure 4.8: Ternary padded voxel matrix (CoolVox)	65
Figure 4.9: 3DCNN architecture for CoolVox1	66
Figure 4.10: Sample results of actual values (top left), predictions (top right), and error plots of all models (middle left: CoolVox1; middle right: CoolVox2; bottom left: ensemble; bottom right: ARINet) respectively for the South-East and North-East views	68
Figure 4.11: : Simulated vs. predicted radiation intensities with boundaries (CoolVox)	69
Figure 4.12: Simulated vs. predicted radiation intensities with boundaries (CoolVox).....	71
Figure 4.13: Simulated vs. predicted radiation intensities without boundaries (CoolVox)	72
Figure 4.14: Simulated and predicted radiation maps for validation sets with boundaries	73
Figure 4.15: Error plots of multiple buildings (left), canopy-style buildings (middle), and tall buildings with self-shading (right)	74
Figure 4.16: Error plots on the test set: CoolVox (left) and ARINet (right)	75
Figure 4.17: The workflow of modeling ANNs-based CFD	77
Figure 4.18: Input and output grids as inputs for 3DCNNs	78
Figure 4.19: Different 3DCNNs architectures for training airflow network.....	79
Figure 4.20: Visualized results comparison: Simulated vs CNNs predictions.....	82
Figure 4.21: Visualized results comparison: Simulated vs. other models predictions	83
Figure 4.22: wind speed analysis for different height of the rooms among different models	84
Figure 4.23: Result plots with different height levels (Conv-Deconv-Skip).....	86
Figure 4.24: The comparison of results with simulated values in different section levels.....	87
Figure 4.25: The comparison of results with simulated values and error plots (Test geometry).	88
Figure 4.26: The result comparison of the different positions of building in the voxel.	88

Figure 4.27: The result comparison of the different building's inlets location	89
Figure 4.28: Diagram of converting both surface to voxel and voxel to surface	90
Figure 4.29: Ray-casting solution and algorithm application	91
Figure 4.30: Marching cube points reference	92
Figure 4.31: Voxel classification process	92
Figure 4.32: Voxel labeling process	92
Figure 4.33: Symmetries from 256 cases	93
Figure 4.34: Get edge list from lookup table	93
Figure 4.35: Find the location of the vertex along the edge	93
Figure 4.36: Calculate the normal at each cube vertex	93
Figure 4.37: ANNs-solar workflow with 3DCNNs representation	95
Figure 5.1: CAD/BIM to ANN-based BPS comprehensive framework.	98
Figure 5.2: Detailed components for the section 5 and section numbers.	98
Figure 5.3: ANN models in the comprehensive framework.	99
Figure 5.4: Final ANN models for different BPS tasks (left: solar radiation, right: airflow).	99
Figure 5.5: Main data exchangers: pre- and post-processing.	101
Figure 5.6: Data exchangers (GRT and BST) for ANNs-solar.	102
Figure 5.7: Data exchangers for solar diagram	103
Figure 5.8: GRT-Solar input and output information.	104
Figure 5.9: GRT-solar stepwise process	104
Figure 5.10: GRT-Solar value padding on each voxelated grid.	105
Figure 5.11: The IFC inputs for the CAD and voxel models.	106
Figure 5.12: Data exchangers for airflow diagram.	107
Figure 5.13: GRT-Airflow input and output information	108

Figure 5.14: GRT-Airflow stepwise process diagram	109
Figure 5.15: GRT-Airflow value padding on each volumetric grid.....	109
Figure 5.16: The IFC inputs for CAD and ANNs-based BPS models	110
Figure 6.1: Software used for the workflow demonstration	111
Figure 6.2: Workflow for solar radiation ANN model.	112
Figure 6.3: Workflow for airflow ANN model.	112
Figure 6.4: Analysis properties and representation for CAD software (GRT).	113
Figure 6.5: Analysis properties and representation for BIM software (BST).....	114

List of Tables

Table 2.1: Comparison of two data schemas: IFC and gbXML.....	21
Table 2.2: Summary of 3D data representation for DL modeling and training	39
Table 2.3: 3D data representation for building performance simulation.....	45
Table 3.1: 3D data representation for building performance simulation	53
Table 4.1: Customized radiation loss computation	67
Table 4.2: Performances of four 3DCNN models	67
Table 4.3: MSE by 3DCNN model and boundary condition.....	70
Table 4.4: Loss comparisons of proposed CNNs.....	85
Table 4.5: The results comparison between the computation time and model accuracy.....	96

Chapter 1

Introduction

The modeling of net-zero-energy buildings is of increasing interest in both the architecture and sustainable consulting industries. To realize this goal, it is imperative to combine the use of passive and active systems. Most passive building design strategies can be implemented in the early design stages with few design-cost changes, guiding designers to pursue sustainability in built environments and bringing about positive outcomes and low-cost changes (Attia, 2010). Figure 1.1 on the left explains the highest effort put into modelling during the construction and documentation phase of the timeline. However, the graph on the right illustrates the same peak in the schematic design phase yielding the significantly low cost of design changes.

Integration of building performance simulations (BPS) during the early design phase is one way of encouraging designers to participate in this type of endeavor. However, this requires a high level of expertise and is computationally expensive and labor-intensive. Moreover, relevant tools are currently unavailable in a diverse modeling interface that exhibits proper connectivity (Augenbroe et al., 1999). The process of practical building design includes various methodologies

with different design objectives such as thermal, daylighting, airflow, and other associated metrics. Therefore, relevant BPS tools that allow architects to find optimized solutions regarding green design decisions are needed.

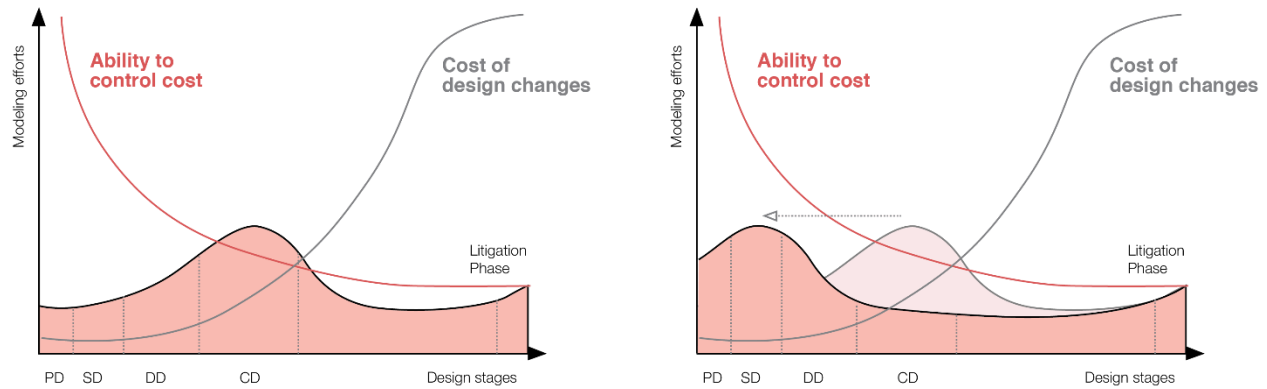


Figure 1.1: Time–cost exchange in different design stages

To achieve this goal, interoperability between diverse models and simulation engines is important, especially when analyzing environmental impacts on buildings. At the same time, input parameters and tuning processes must be integrated as parts of a comprehensive design-consulting workflow.

The present research aimed to develop software and a workflow for early design-decision support by bridging different BPS software with different algorithms and data available from diverse sources. For example, developing models that easily integrate into the CAD software would be a promising approach in early design decision support. Data-driven models have become popular for developing high functional approximators for energy, radiation, airflow, and other performance metrics because of its simple model architecture and intuitive input–output definition. Over the past 40 years, scientific and technological developments have led to an increase in predicting skills through the use of artificial intelligence such as machine learning (ML) and deep learning (DL). Since current building simulation practices rely substantially on

physics-based simulation engines, data-driven models such as ML and DL can be newly deployed to predict a performance measure in buildings and built environment. By taking data-driven modeling methods, environmental assessment and performance-driven design workflow will be integrated into early design decision making process. Therefore, the radical early engagement from designers in the performance driven design can be ideally achieved.

1.1 Research Questions

The importance of the early design engagement of performance simulation has been highlighted to reduce potential cost change loss during the design construction timeline. In both academia and industry, building performance simulation tools have become increasingly popular for early design decision-making (Attia and Herde, 2011). Figure 1.2 shows an ideal collaboration between different domain experts, such as engineers and architects, during the early stage of the design. However, currently available BPS software requires a certain amount of domain knowledge and high computational power, resulting in workflow being discretely spread among different areas of expertise and causing a lack of communication.



Figure 1.2: Early design decision-support tool development for collaborative working environments

Limitations of the current BPS consulting environment include a lack of domain knowledge for assumptions regarding related parameters, computational burden of physics-based numerical calculations, lack of software integration and interoperability between CAD and BPS software, and complicated user interface. Because it is challenging to involve all individuals with domain expertise during the early design and analysis workflow, data-driven surrogate models are an attractive alternative providing performance feedback based on physical knowledge (Forrester, Sobester, and Keane 2008).

Historically, the two easily utilized models are the physics-based model and the data-driven model. Physics-based models have been used and validated more than three decades. However, it requires making many assumptions as inputs and models are computationally expensive yielding considerable calculation time. Therefore, the data-driven models have gotten fame due to their advantages in the early design adaptation. Data-driven models are simple, computationally efficient, and useful for optimization, design space exploration, prototyping, and sensitivity analysis (Gorissen and Dhaene 2010). The advantages of data-driven models are; black box models do not require many assumptions, the GPU enables training data-driven models efficiently with a low calculation time, once trained, the data-driven model can integrate into any type of software, the user interface can be easily simplified. Due to their advantages, data-driven models have been developed as surrogates of existing physics-based models. Artificial neural networks (ANNs), kernel methods, Bayesian inference, and ML are the most popular surrogate modeling techniques for facilitating design optimization workflow while also minimizing computation time and workflow complexity.

Because ANNs require fewer inputs and less computational time compared to numerical methods (Miller, 1968), offer superior performance (Kalogirou and Bojic, 2000), and have the potential for data augmentation (Wang and Perez, 2017), they are widely used in predicting solar radiation and

airflow patterns on and around structures, and building energy use. ANNs can rapidly provide innovative design solutions, allowing designers to receive instantaneous feedback on the effects of a proposed change to a building's design (Nguyen et al., 2014). The advantages of ANN models include reproducibility, time-efficiency, and scalability (Goldstein and Coco, 2015). ANN models can reduce the time complexity of calculating optimized design solutions during the iterative simulation process and thus are relevant for use in the early design stages. Furthermore, because of their lightweight structure, minimizing the computational barriers to BPS modeling can be with increments in tool interoperability between CAD software and ANN models. However, there are drawbacks to using ANNs, including the limited number of input and output parameters, lack of design options for detailed modeling and simulation (Geyer and Schlüter, 2014), and absence of interpretability of their mathematical structure (Singaravel et al., 2018b).

The recent trend of combining the strengths of data-driven and physics-based models in a single hybrid model has been suggested as a valuable step forward in the software development industry because of potential increases in accuracy (Babovic et al., 2001) and speed (Krasnopolsky and Fox-Rabinovitz, 2006). A few studies used this approach in the field of environmental science and BPS, embedding ML components directly into models for rainfall (Jain and Srinivasulu, 2004) and flow simulation (Corzo et al., 2009) and during energy model calibration (Chong and Menberg, 2018). Figure 1.3 shows the usage of physics-based numerical rules to develop physics-guided data-driven models. Data-driven models' development and enhancement can be by using physical rules as loss functions or constraints for variables or propagating data points with simulated data from physics-based models.

The following questions drive the current research for the use of physics-guided data-driven models in BPS include: In what ways can data-driven models reduce the need for domain experts, How can we achieve a high degree of computational efficiency, How can data-driven models be

integrated into CAD modeling software, What is the next generation of BPS interface design with data-driven models in terms of usability.

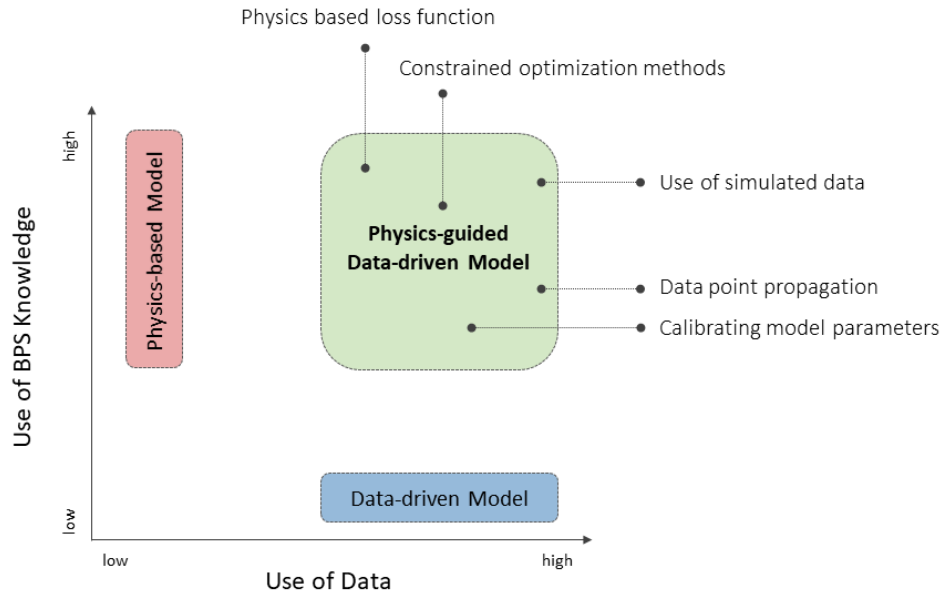


Figure 1.3: Physics-guided data-driven models and the requirement of domain knowledge

To achieve an efficient performance modeling workflow with an intuitive and user-friendly interface in existing CAD software, a seamless framework is required to compute the building information to the model and further analyze the results. Therefore, this dissertation addressed issues relevant to the realization of early design integration of the proposed modeling methods.

1.2 Objectives

- Develop a framework to integrate physics-guided data-driven models with any type of CAD software
- Investigate if physics-guided data-driven models would reduce the number of assumptions needed as inputs, thus enabling the introduction of a simplified user-friendly interface
- Investigate if performance-driven building design can be enhanced by the active use of physics-guided data-driven models early in the process of a building's design
- Introduce a comprehensive framework for 3D data representation in ANNs that will serve as a means of representing the physical properties of buildings and the environment

The following chapters investigated applicable data-driven models such as ANNs for predicting diverse building performance metrics utilizing 3D data schema and data processing methods. Based on the proposed ANN models, the potential application of data-driven simulation of existing simulation workflow will be validated and highlighted.

Chapter 2 gives a brief overview of the history of building performance simulation for software interoperability and a 3D data representation for ANNs in computer vision. Chapter 3 provides a methodology for completing an ANNs-based BPS framework. Chapter 4 includes several published works regarding the detailed modeling methods for ANNs for different BPS tasks and the algorithms implemented. Chapter 5 delineates the comprehensive interoperability frameworks for this dissertation. Chapter 6 validates the proposed interoperability framework with current modeling software and newly developed computational tools. Finally, chapter 7 concludes and discusses the future outlook.

Chapter 2

Literature Review

Dynamic data-driven simulations require the ability to incorporate data into existing models, and in reverse, the application must have the capacity to steer dynamic modeling processes. In recent years, there has been a significant maturation of measurement infrastructures ranging from instruments to sensor systems, data storage technologies, and remote data access and prediction mechanisms (Darema 2011). Data-driven and self-learning simulation methods promise improvements in modeling processes, augmenting the analysis and prediction capabilities of building simulations, and advancing the efficiency of simulations and effectiveness of measurement systems (Dimitriou et al., 2016). With these trends, the quality advancements in dynamic data-driven building simulations will only be accomplished by developing the measurement infrastructure for BPS, as well as computational capacities and BIM technology. In this chapter, the history of interoperability and the techniques in computer vision were comprehensively reviewed.

2.1 History of software interoperability and framework

2.1.1 History of buildingSMART and the industrial foundation

class

For many years, the International Alliance for Interoperability (IAI), now called buildingSMART, has made efforts for the reliable exchange of Building Information Model (BIM) data among stakeholders in the architectural engineering and construction (AEC) industries (Karlshøj et al., 2012). Autodesk launched the Private Alliance (PA) for interoperability in building design industries in 1995. It engaged with a variety of companies around the globe, such as Autodesk, Archibus, AT&T; HOK Architects; Honeywell; Jaros, Baum, and Bolles; Lawrence Berkeley Laboratory; Primavera Software; Softdesk Software; Timberline Software; and Tishman Construction (BuildingSMART, 2018). After a year of effort, the companies reached three critical conclusions: the commercial potential of interoperability and the need for open and international standards and open parties around the globe. Based on these needs, they established IAI on May 16, 1996, with representatives from North America, Europe, and Asia. A total of 12 groups and several international companies, other than the ones mentioned above, participated. The mission of IAI was to provide a universal basis for process improvement and information sharing in AEC industries (BuildingSMART, 2018).

IAI released version R1.0 of the Industrial Foundation Class (IFC) in January 1997. Seventeen companies participated in the pilot implementations. Equipped with limited functionality, including five processes for architectural design, such as Heating, ventilation, and air conditioning (HVAC) design and construction and facilities management (Kiviniemi, 1999), was the IFC R1.0. Twenty-seven software vendors spent ten months developing IFC R1.5 (released in November 1997). However, the main evolution of the features in the current IFC occurred at the

IAI Washington summit in April 1999, as project IFC R2.0. The IFC R2.0 scope included architecture extensions, HVAC systems, code checking, cost estimating, occupancy management, property management, general-purpose networks, and external document references (Liebich, 2010). Following the creation of IFC R2.0, in October 2000 and May 2003, released respectively were IFC R2x and IFC R2x2. The increased scope included 2D geometry interpretation, structural analysis, and detailing (Laakso and Kiviniemi, 2012).

The ifcXML software, the official XML representation of IFC, was developed and released in late 2003. It provided XML data structure bindings to the IFC EXPRESS schema. However, the STEP-File and XML modeling structures were inherently different, so the translation of the IFC STEP-file to ifcXML resulted in a needlessly significant loss in the un-optimized conversion of XML modeling (Behrman, 2002). During this period, the formation and initiation of the ProIT project took place, which was a Finnish effort that ran between 2002 and 2005. ProIT (2004) demonstrated the importance of using BIM technology in projects, serving as modeling guidelines for architectural and structural design (Laakso and Kiviniemi, 2012).

In January 2008, IAI reformed, changing its name to buildingSMART, to pursue a mature reflection on the nature and goals of the organization. Whereas the old vision of IAI aimed “*to enable software interoperability in the AEC/FM industry,*” the new vision went beyond the technical aspects to emphasize what interoperability might mean in business: “*Improving communication, productivity, and delivery time, cost, and quality throughout the whole building life cycle*” (Stangeland, 2009). Currently, buildingSMART has 13 chapters around the world. In each, two delegates meet twice annually in an international council to coordinate business and technical strategies.

In its approach to overall standardization, an increased amount of focus was on the minimalistic/bottom-up methods of narrowing down IFC data exchanges into manageable,

predictable, and implementable specifications (Hietanen and Lehtinen, 2006). Reducing the scope of information exchange from dealing with the entire IFC data model to well-supported and predictable workflows, they considered it a gateway through which the industry and implementers might increase their support for the standard. Subsequently, incremental increases in the number and scope of the supported exchanges would be easy using standard growth (Laakso and Kiviniemi, 2012). One outcome of this minimalistic approach was the Information Delivery Manuals (IDMs), introduced in 2007 as a part of IFC standardization. Another key feature was the IFC Model View Definition (MVD) format, proposed in 2005 by BLIS. The IFC data model served as the foundation for defining specific MVDs. The IDMs provided the documentation and workflow guidelines for IFC exchanges, designed by acknowledging the functionality of specific MVDs. The purpose of both the cross-referenced exchange layers was for facilitating the deployment of IFC-supported interoperability (Bell and Bjorkhaug, 2006).

To date, the development and utilization of various versions of the IFC models have been by different vendors in the architecture and construction industries. IFC 4 is the most recent version of the standard, focuses on placing quality over speed and obtaining the full ISO standard (Liebich, 2010). Twelve categories across the various AEC industries are currently available on the IFC4 application, such as architecture, building performance energy analysis and simulation, data server, facility manager, model viewer, geographic information system, et cetera. Different software companies throughout the world's BIM industries including, Autodesk, ACCA Software, Bentley Systems, Tekla, Allplan, Archicad, Trimble, Aconex, Venturis, and DDS-cad (BuildingSMART, 2018) have approved the IFC certification process. For example, Autodesk and Bentley enable AEC project teams to combine related features in integrated workflows by supporting reciprocal use of available resources (Autodesk, 2008).

Public sector property owners worldwide are among the most influential supporters of IFC-based interoperability in connection with the issuing requirements and guidelines for the increased use of BIM technology in situations where IFC plays an integral part in keeping information open and non-proprietary. On 17 January 2008, AEC/FM sector government client organizations: from the US (GSA), Denmark (DECA), Finland (Senate Properties), Norway (Statsbygg), and the Netherlands (Rijksgebouwendienst) issued a signed “Statement of Intention to Support Building Information Modeling with Open Standards” (Winstead et al., 2008), which explicitly committed signers to facilitate the use of the IFC standard. Scandinavian countries such as Finland and Norway have long been pioneers in demanding BIM with IFC deliverables (Kiviniemi et al., 2008; Lê et al., 2006). In sum, in the last few decades, both the public and private sectors have maintained and renewed the IFC.

According to (Laakso and Kiviniemi, 2012), IFC development faced challenges in acquiring sufficient resources to manage and revise the standard, due to weak levels of coordinated market demand and open BIM participation. Despite limitations on market values and profits, the development of the standard was organized, and its scope was extended to lifecycle and building energy simulations. Thus, rigorous engagement from diverse industry sectors and academia is required. Building performance simulation (BPS), the six currently supported IFC-compatible tools are IDA ICE, RIUSKA, Simergy, OpenStudio, IES-VE, and TerMus. Except for RIUSKA, all tools incorporate an importing function for data deserialization, while serialization is essential in sustainably maintaining the data exchange process.

Moreover, this commercially available IFC-compatible software can dramatically reduce the cost and time needed for energy analysis (Bazjanac and Crawley, 1999); therefore, the need for active use of IFC in BPS is increasing.

2.1.2 Building Information Modeling

A Building Information Model (BIM) definition by international standards is a “shared digital representation of the physical and functional characteristics of any built object that forms a reliable basis for decisions” (ISO, 2010). BIM offers solutions to several inefficiencies and systems failures plaguing the design, construction, and maintenance industries (Eastman and Jeng, 1999). The first introduction of the BIM technique was in the early 2000s through BIM and IFC pilot projects that published pragmatic guidelines and instructions for architecture and engineering practice (Penttila et al., 2007). The use of BIM incorporates building design, construction, related infrastructure, and building lifecycles; it is not limited to simple geometry but rather extends to complex systems (Akbarnezhad et al., 2014). According to a recent survey, BIM is adequate for larger and more complex geometric and performance management systems and improves the quality of projects and collaboration networks among owners and involved stakeholders (Becerik-Gerber and Rice, 2010).

BIM is a tool for managing comprehensive building systems throughout a structure’s lifecycle. It allows for powerful data maintenance and the sustainable use of involved information (Akbarnezhad et al., 2014). The initial purpose of BIM was to support the design construction process and information regarding fundamental building attributes already documented, such as the coordinates of the geometry, structural elements, and material properties. Adding or updating other relevant information such as the Life-cycle assessment (LCA) and building performance metrics can be as a functionality requirement (Volk et al., 2014). Because BIM deals with multiple information criteria, to realize the software is through object-oriented programming consisting of parametric objects representing building components (Lee et al., 2006). According to (Wong and Yang., 2010), objects can have geometric and non-geometric attributes communicating through functional, semantic, and topological information. To elaborate on this point, functional features

contain cost and project durations, while semantic attributes include connectivity, aggregation, and intersection. Topological aspects address information regarding location, adjacency, coplanarity, perpendicularity, et cetera (Volk et al., 2014). Due to these involved functionalities and the complex relationships among them, organizational efficiency concerning data exchange and communication processes are crucial.

The Information Delivery Manual (IDM) framework and Model View Definitions (MVD) provide relevant information, facilitating data exchange and avoiding uncertainties (Afsari et al., 2016). The IDM framework defines the functionality-related exchange process for a particular topic. For example, the IDM presentation for energy analysis focuses on energy modeling (US GSA, 2010). MVD comprises a subset of the IFC essential to satisfying diverse data exchange requirements. In other words, the bases of creating and maintaining MVD are on required functionalities such as energy and structural analyses and refer to BIM objects and related attributes in interactive views (Volk et al., 2014).

The Industry Foundation Class (IFC), organized in ISO, is the dominant non-proprietary exchange format between AEC and FM software. It was developed to represent building information and other details throughout a building's lifecycle (Cho et al., 2010) and facilitate data transfer between BIM modeling software (e.g., Autodesk and Bentley), IFC viewers (e.g., IFCStoreyView), and advanced knowledge-based programs (e.g., EnergyPlus and OpenStudio). The limitations of BIM's interoperability are due to the incomplete, vague, black-box language type used for IFC denotations and contents (Abanda et al., 2010). Recent developments in IFC data exchange focused on implementation of semantic web technologies in open format ontologies like HTML, XHTML, gbXML, COBie, ifcXML, IFC, and CIS/2. These enable software applications but most are currently unavailable in academia (Eastman et al., 2011; Abanda et al., 2010; Mohd-Nor and Grant, 2014).

To date, there have been tremendous advancements in BIM modeling and maintenance; however, a few significant challenges remain, such as automation of the data imputation process, regular updating and maintenance of the database, and the handling of uncertain data and object notations (Liu et al., 2017). Despite these shortcomings, rapid developments and the recent standards release indicate the need for the process automation and functional extension in the future (Afsari et al., 2016). The growing capacity of an increased digitalized database and automated process is expected to stimulate BIM implementation in building performance modeling (BPM) and maintenance through semantic web technology, cloud computing, and mobile BIM devices (Nicolle and Cruz, 2011).

2.1.3 Building Performance Simulation

Despite the prevalent use of BIM in AEC industries, Building Energy Modeling (BEM) remains rarely employed in building design, commissioning, and operation, due to the relatively expensive labor cost and computation time. According to (Bazjanac, 2008), BEM's use in project delivery occurs in less than 1 percent of the "run of the mill" new US building stock, and a similar trend is noticed even on the global scale. BEM requires architectural definitions and HVAC specifications, as well as definitions for plug and lighting loads and occupancy patterns. Such datasets result in arbitrary assumptions when preparing the input for a simulation, and the computing process can be lengthy, laborious, and resource consuming. All sorts of data change from their original forms, whereas others are unavailable in any form. Commonly, these are estimated or guessed at, resulting in numerous coding errors. Consequently, arbitrary assumptions are unavoidable when preparing input for BEM simulations (Bazjanac, 2008).

BIM-integrated BEM modeling is an essential factor in the performance-driven design and the automated data transfer process (Venugopal and Eastman, 2012). BEM users can efficiently

extract BIM data from a digitalized database for individualized use. However, due to the tremendous amount of building information and related functionalities, data loss occurs when converting variables from BIM to BEM. Moreover, the reconstruction cost is high, requiring different areas of expertise (Moon et al., 2013). According to (Hijazi, Kensek, and Konis, 2015), project losses caused by interoperability, as issued in August 2004 by the US National Institute of Standards and Technology (NIST), estimations were around \$15.8 billion annually. To mitigate this loss, the creation of a seamless data exchange pipeline between 3D BIM software and BEM simulation tools is essential (Hijazi et al., 2015). The US General Services Administration (GSA) released the GSA BIM Guide for Energy Performance as a strengthening method for reliability, consistency, and usability of predicted building energy use and cost results (GSA, 2009). Currently, conducting rudimentary building performance analyses may be direct via a few existing BIM tools (e.g., Revit® MEP, Rhinoceros 3D, etc.). However, these tools are not comparable to typical, full-scale simulations conducted using standalone analysis mechanisms such as EnergyPlus™ and DOE-2 algorithms due to the current absence of data exchange capacity.

2.1.4 History of Building Performance Simulation and software

Interoperability

Importantly, traditional applications and systems for building performance simulations (BPS): though complex, have lacked accuracy in predictions. This is because the various input parameters have, for decades, relied on empirical models, averaged data values, and modelers' assumptions (Nouidui et al., 2018). Several parameters and associated equations can lead to inaccurate results and range gaps (Malkawi and Augenbroe, 2004). Quality advancements in dynamic data-driven building simulations can only be through developments in the measurement infrastructure for buildings, data from the high-fidelity simulation engines, better computational capacities, and the use of BIM technology.

BPS software packages, depending on their intended purpose, can differ in terms of the input, output, programming language, and Graphical User Interface (GUI) platform. Currently, BPS professionals frequently use dedicated GUI such as Simergy (Karlshøj, See, and Davis, 2012), OpenStudio (Guglielmetti, 2011), DesignBuilder (Tindale, 2005), and IES-VE™ to facilitate environmental and energy analyses (Shelden, 2009). To date, contemporary industrial users have often employed BIM. Revit Architecture (Autodesk, 2018), Building Designer V8i (Bentley System, 2018), and ArchiCAD (GRAPHISOFT, 2018), also assist in the design process in comprehensive ways. The past and ongoing research are aimed toward integrating BIM into BPS; however, there continues to be a lack of interoperability and connectivity between the BIM and BPS tools.

A typical BIM to BPS workflow transfers the data input for the energy analysis from a BIM to a BPS engine (Appendix A). Sometimes, the BPS to BEM transfer is necessary for visualization and documentation (Bazjanac, 2008). In the 1990s, IDD and GLIDE (Eastman, 1977), ARMILLA (Gauchel and Hovestadt, 1992), COMBINE and COMBINE 2 (Augenbroe, 1992, 1995), Knowledge-based Design Support (KNODES) (Rutherford, 1993), SEMPER (Mahdavi et al., 1997), and BDA (Papamichael, 1999) were developed as consecutive efforts to integrate BIM into the BPS software. The COMBINE and COMBINE-2 projects demonstrated the potential of linking existing tools such as those for energy, daylighting, and other aspects. The EXPRESS language has become the binding block for the KNODES framework (Rutherford, 1993).

The realization of significant advancements in integration tools; was through Design Analysis Integration (DAI) and SEMPER (Mahdavi et al., 1997). The DAI workbench offers a process-centric toolkit for overcoming the limitations of data-centric interoperability approaches (Augenbroe and de Wilde, 2003). SEMPER provided a more active, multi-aspect design environment that later expanded to the web-based SEMPER II or S2 (Lam et al., 2004). In this

building model representation, the improvising of IFC is through the dynamic capabilities of temporal databases to mine, learn, and dynamically respond to changes in building states. Continuous developments include DeST (Yi et al., 2007), BCVTB (Wetter, 2011), SimModel (J. O. Donnell et al., 2011), Simergy (See et al., 2011), CBEMS (Wang et al., 2011), and D-BIM (Ravi Srinivasan, Charles Kibert, Paul Fishwick, Zachary Ezzell, Siddharth Thakur, Ishfak Ahmed 2016). The development of DeST was to share generic data in IFC with other BPS simulations such as ventilation and lighting analyses. The creation of BCVTB was to incorporate design optimization processes into BPS. CBEMS uses a tiered integration application of different BPS matrices with policy learning and artificial intelligence techniques: while D-BIM offers an open-source graphical engine and user-friendly GUI for early design support. Most of the tools presented after the late 1990s actively use the IFC (IAI 1997) data schema for software interoperability due to its universal means of exchanging BIM data among commercial software.

Existing BIM file formats for BIM to BPS data exchange include HyperText Markup Language (HTML), gbXML, IFC, and ifcXML (Volk, Stengel, and Schultmann, 2014). Two main methods for extracting data for BPS from BIM are IFC and green building XML (gbXML; Miller et al., 2014). Such formats are still not fully connected to BPS tools; for example, EnergyPlus uses the intermediate data format (IDF) and an E+ specific structure based on XML schema. Several studies on data conversion have been conducted to overcome the pervasive lack of connectivity. They include gbXML to IDF (Dimitriou et al., 2016), IFC to IDF (Kim et al., 2012), semi-automated BIM to BEM data exchange (O'Donnell, 2014), VFEA (GIS and BIM-based virtual systems; Wu et al., 2014), and BIM2BEM (IFC to Modelica; Jeong et al., 2016) using the IFC data schema.

Recent investigations related to data transfer and BPS consulting has addressed the rapid adoption of web applications that are inherently cross-platform, mobile, easy to access, and offer

better user interactions. JavaScript Object Notation (JSON) is a widely used data format on the web for asynchronous browser-server communication employed as a replacement for XML in some systems. JSON is a language-independent data format derived from JavaScript that offers several advantages, overcoming the drawbacks of IDF, ICF, and gbXML. FloorspaceJS (Macumber et al., 2018) was developed as a web-based open-source framework for BPS using the JSON schema. New et al. (2018) released a JSON-based IDF converter for advanced BPS modeling and analysis.

Another reason for using the web interface for the simulation is to ensure a seamless workflow for model deployment and data management on the web. The currently available cloud-based GPU and database enable us to manage large sets of data and efficiently perform model training. This trend reflects the contemporary movement in BPS research in the recent times as well as for the future. Figure 2.1 illustrates the history of interoperability in the chronological order.

Process-driven interoperability, intelligence-based optimization techniques for new tools, and novel means of interaction between users and BPS are desirable capabilities and necessary requirements for future BPS software (Malkawi, 2004). As discussed above, the universal use of available data and models for building simulations and intelligence-driven performance analysis can be expected. The BPS community will likely leverage the contributions of this research in new and creative ways.

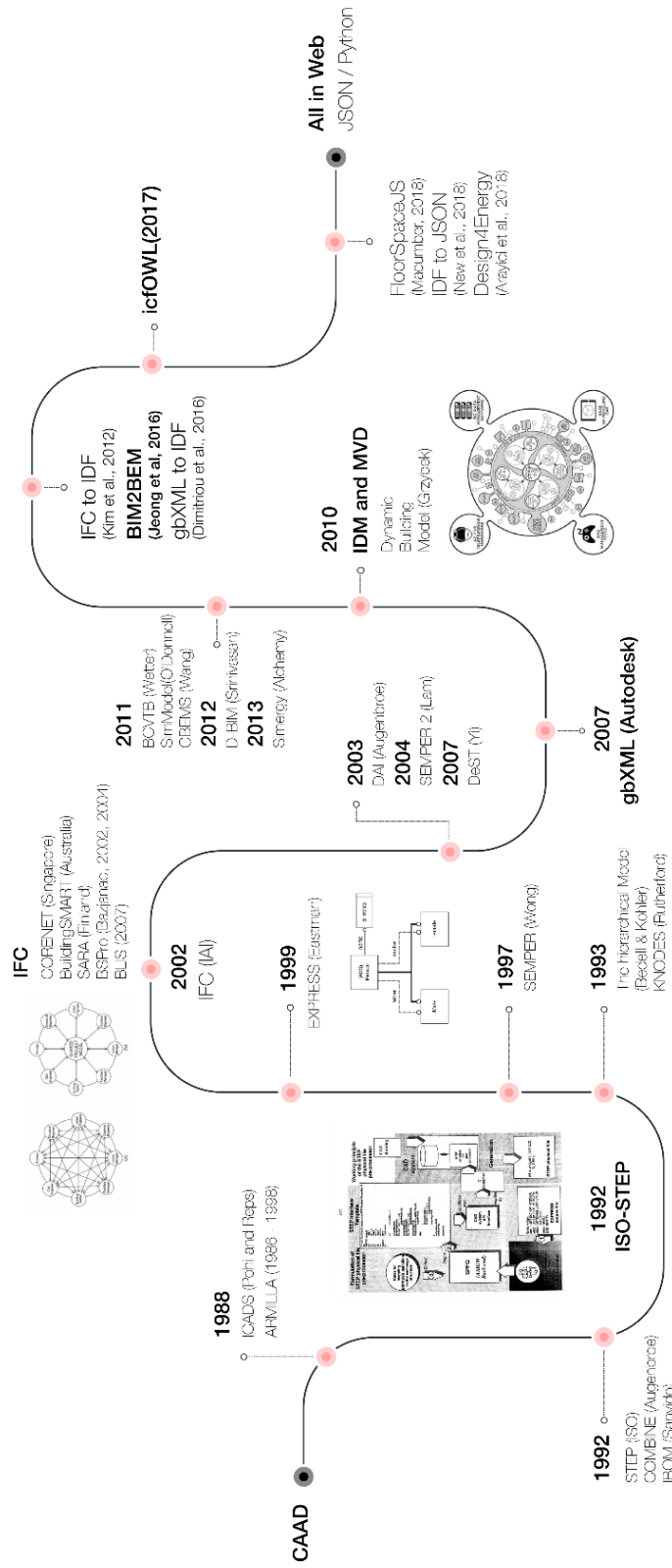


Figure 2.1: The history of Interoperability and future direction

2.1.5 Interoperable data format

To date, the existing BIM file formats for data exchange includes HTML, gbXML, IFC, and ifcXML (Volk, Stengel, and Schultmann, 2014). In the architecture, engineering, and construction (AEC) industries, two main methods for extracting data for BIM-based building performance simulations are IFC and green building XML (gbXML) (Miller et al., 2014; Dong et al., 2007). IFC is a vendor-neutral open data exchange specification with an object-oriented file format, mainly used in BIM. The International Alliance for Interoperability (IAI) developed IFC, and Building SMART International (ASHRAE, 2005) has maintained it since 2005. IFC uses EXPRESS-G to identify classes. It is a graphical modeling notation developed on the ISO-STEP schema to represent models. The gbXML is an energy simulation-specific Extensible Markup Language (XML) developed by Autodesk (2007). The IFC file type is more comprehensible and generic; it aims to represent extensive building project areas with all related information, whereas gbXML focuses on the properties of the building project closely related to energy simulation (Lam et al., 2012). Table 2.1 shows the pros and cons of each data schema described in this section.

Table 2.1: Comparison of two data schemas: IFC and gbXML

	IFC	gbXML
Domain	Building construction to operation	Building energy simulation
Geometry	Any generic shape	Rectangular shape
BPS connectivity		Possible to get input for EP or EQuest
Complexity	Top-down and relational approaches	Bottom-up approach
Hierarchy	Complex data representation with large data file	Flexible and straightforward

To increase the accuracy of BPS, BIM, BPS metrics, and further predictions are crucial when using an integrated exchange method between the measured data obtained from sensors and weather stations (Donnell et al., 2013); similarly, the amount of time consumed can be reduced and the

accuracy of the results delivered can be improved by automating the data exchange process between BIM and BEM. According to (Vladimir Bazjanac and Crawley, 1999) the building geometry, climate information, mechanical system, and operational schedule are essential factors in building performance simulations. Recognizing this exchange among the software packages for BPS has been the focus of the academia as well as the industry to support better decision making.

However, the BIM format, which is a copious building information source, cannot still be connected to BPS tools; for example, EnergyPlus uses the Intermediate Data Format (IDF), which has an E+ specific structure based on XML schema. There have been studies on overcoming the lack of connectivity related to data conversion methods for energy simulations, such as gbXML to IDF (Dimitriou et al., 2016), IFC to IDF (Kim et al., 2012), the SimModel by O'Donnell (2013), and a semiautomated BIM to BEM data exchange by O'Donnell (2014) that uses the IFC file format. These attempts have enriched the building simulation process and allowed for more accurate building calibration and prediction.

However, such discretized processes should be integrated for a more comprehensive data flow. With the development of data analysis techniques, metadata collection, and transfer methods, the amount of big data available from building environments and boundary conditions, as well as from the direct inputs from Internet Of Things (IoT), has increased. Many industries are now rapidly adopting web applications and platforms that are inherently cross-platform, mobile, and easy to transfer and distribute. The BPS community is beginning to notice this trend, with a small but growing number of BEM applications that have begun incorporating or moving to the web environment.

JavaScript Object Notation (JSON) is the widely used data schema on the web for asynchronous browser-server communication, including serving as a replacement for XML in some systems. JSON is a language-independent data format derived from JavaScript that has several advantages

and overcomes many of the drawbacks of IDF, ICF, and gbXML. The essential value-based data structure (ECMA-404, JSON standard) is not positional (i.e., index-based) and can incorporate extraneous fields; hence, allowing coding for multiple attributes on the unlimited length of the extensible fields (Mark et al., 2017). Reusability and minimal dependencies were essential design considerations in the software’s development; JavaScript was developed to have maximum portability and reusability in web applications. The emphasis on the format is not limited to web interoperability and can be extended to 3D modeling, with connectivity to the C++/C# language-based modeling method. There are several advantages of using JSON data schema.

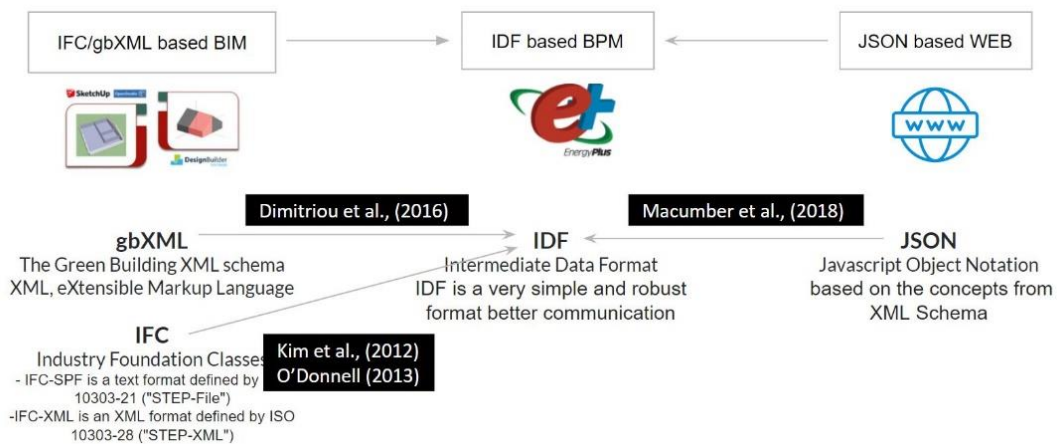


Figure 2.2: From BIM to BEM and Web

JSON data structure has been used and developed for building performance simulation schema and workflow (New, Ph., and Adams, 2018; Macumber et al., 2018). The discussions on how users can employ several languages and tools to efficiently create, manipulate, and validate input files using the JSON schema (New et al., 2018) happened. The integration of the JSON input into EnergyPlus 8.9, FloorspaceJS (Macumber et al., 2018) increases the potential capacity of integrating the web environment into 3D BPMs using the JSON structure. The coding of FloorspaceJS was in Javascript with a consistent JSON data schema that can translate data into

gbXML, IDF, IFC, and so on. Furthermore, in the future, automation of the conversion process can be done using any programming language (Figure 2.2).

With the increasing attention to digital twins, building modeling industries now rely on semantic web technologies to support data exchange through the web. This had led to improved information exchanges with sources outside the traditional BIM environments and the existing techniques. The primary methodology to connect the semantic web to the IFC schema is Web Ontology Language (OWL) ontology for IFC (Pauwels et al., 2017). Unlike EXPRESS data schema, ifcOWL exploits semantic web technologies' enablers regarding data distribution, diverse use of the data model, and querying and reasoning by keeping the IFC standard for representing building information. The ongoing efforts on developing and implementing ifcOWL ontology have been ongoing since 2016, and now the adoption of the ifcOWL schema is the main schema from BuildingSmart initiatives (Pauwels and Terkaj, 2016).

2.2 Building performance simulation and data-driven modeling

For dynamic data-driven simulations, the data must be incorporated into existing models and the application must also have the capacity to steer the measurement processes. In recent years, there has been a significant maturation of the measurement infrastructures ranging from simple instruments to sensor systems, data storage technologies, and remote data access and prediction mechanisms (Darema, 2011). Data-driven and self-learning simulation methods have led to improvements in modeling processes, thereby augmenting the analysis and prediction capabilities of building simulations and advancing the efficiency of simulations and effectiveness of measurement systems (Dimitriou et al., 2016).

2.2.1 Surrogates as the alternatives of physics-based models

An increasing number of architects and engineers face the need for BPS software and its engine to achieve performance-driven building design. Models are a useful tool for understanding the processes and physical properties of buildings and making inferences about the future, and giving feedback on design changes and optimization processes (Clarke, 2015). Physics-based model's traditional use is to estimate the energy flow, air movement, and heat balance of buildings. The algorithms behind physics-based models; involve solving complex differential equations and require many assumptions, high-performing hardware, and a considerable amount of computation time (Malkawi, 2004).

Utilizing artificial neural networks (ANNs) as a surrogate model has become indispensable due to the computational cost of high-fidelity simulations (Gorissen and Dhaene, 2010). In other words, the usage of statistical models can be surrogates for detailed simulation models to support performance-driven design (Westermann and Evins, 2019). Such surrogate models (SMs) are compact and have proven useful for tasks, such as early design space exploration, sensitivity analysis, uncertainty quantification, and design optimization (Nguyen et al., 2014). For early design decision-making, utilizing building simulation tools has become increasingly popular in academia and industry (Attia and Herde, 2011). However, currently available BPS software requires a certain level of domain knowledge and high computational power. Because it is tough to incorporate all of the necessary domain expertise in the early design and analysis workflow, SMs are an alternative that provides performance feedback based on physical knowledge (Forrester, Sobester, and Keane, 2008).

Commonly in early design, setting up one simulation case for one specific concept involves the manual definition of complex parameters (Nguyen et al., 2014). Furthermore, the runtime is

relatively long and may interrupt the architect's train of thinking for the conceptual design; ideally, the software feedback time is less than 10 seconds (Miller, 1968). As the evaluation of surrogates is instant (< 0.1 sec; Liesje et al., 2014), they are able to provide rapid feedback and an expeditious optimization. Moreover, the computational layout of SMs tends to be lightweight, and embedding it into any BPS software can be easy.

Other uses of SMs are sensitivity analysis and uncertainty quantification. Sensitivity analysis serves as a preliminary step in BPS processes in reducing problem complexity (Samuelson et al., 2016). There are two different approaches: local and global. Global methods investigate the influence of parameters design space; thus, they require many processed samples with all possible design pairs. SMs speed up the time of sample generation (Wei, 2013). The purpose of a sensitivity analysis is to quantify the effect of the design changes with different inputs and outputs. Uncertainty quantification analysis determines the likelihood of changes in output induced by input (Heo et al., 2012). The use of SMs in uncertainty modeling is similar. SMs are strong enough to produce the performance distribution, which requires a significant number of simulation samples. For sample generation, SMs cannot only be fit initially and then used for the control and optimization process (Wang and Shan, 2007).

However, SMs have drawbacks, including the limited number of input and output parameters and design options for detailed modeling and simulation (Geyer and Schlüter, 2014) and the lack of interpretability of their mathematical structure (Singaravel et al., 2018). SMs often predict aggregated design metrics (e.g., annual energy use and radiation intensities) rather than detailed time series results (e.g., hourly energy use and monthly radiation intensities worldwide). Recurrent neural networks are a type of ANN, and their use has been to solve issues related to predicting time series-based values (Han, 2021; Babovic, Can, and Rene, 2001).

There are many ways to generate a relatively accurate approximation model while minimizing the BPS simulation time. Linear regression is the most widely used method for calculating building energy use (Gratia, 2002; Jaffal et al., 2009; Hygh, Decarolis, Hill, and Ranjithan, 2012; Catalina, Iordache, and Caracaleanu, 2013; Geyer and Schlüter, 2014; Ritter, 2015). ANNs and support vector machines are being increasingly used in early design performance assessment (Rackes, Paula, and Lamberts, 2016; New, Ridge, and Parker, 2017; Ascione et al., 2017; Singaravel, Suykens, and Geyer, 2018). Other than function approximators, Bayesian networks are actively used in building energy simulation and calibration to cover areas of uncertainty in modeling (Heo, Choudhary, and Augenbroe, 2012; Chong and Menberg, 2018). In addition, evolutionary algorithms expand the ways of exploring design space with proper models; thus, regression-based SMs have been widely adapted in BPS (Machairas, Tsangrassoulis, and Axarli, 2014).

The suggestion based on the recent trend of combining the strengths of data-driven and physics-based models in a single hybrid model is a valuable step forward in the software development industry because of increases in accuracy (Babovic et al., 2001) and speed (Krasnopolsky and Fox-Rabinovitz, 2006). Few studies have already used this approach in the field of environmental science and BPS, embedding ANNs components directly into rainfall models (Jain and Srinivasulu, 2004) and flow simulations (Corzo et al., 2009) and the localized weather predictions (Han, 2021).

2.2.2 Artificial Neural Networks (ANNs) for BPS modeling

Over the last several decades, in industry and academia, there has been extensive use of physics-based models to evaluate the performance of buildings. Physical models are now being replaced by ANNs. Because ANNs allow fewer inputs, require shorter computation times (Miller, 1968), exhibit superior performance (Kalogirou and Bojic, 2000) and offer the potential for data

augmentation (Wang and Perez, 2017), their use in the prediction of solar radiation and airflow patterns on and around structures and building energy use has drawn significant attention. ANNs can rapidly provide innovative design solutions, enabling designers to receive instant feedback on the effects of a proposed change to a building's design (Nguyen et al., 2014). The benefits of an ANN in building design include reproducibility, time efficiency, and scalability (Goldstein and Coco, 2015). ANN can also be used to reduce the time complexity of calculating optimized design solutions during the iterative simulation processes and thus are relevant in the early design process. Furthermore, because of the lightweight structure of ANN models, the computational barriers to BPS modeling can be minimized with increments in tool interoperability between CAD software and ANN models.

ANN models are a class of models that can learn a hierarchy of features by constructing high-level features from the low-level ones (Goodfellow, 2016) through the combination of information from all the channels, adequate analysis of all the phenomena related to building indoor areas. In particular, the recent use of convolutional neural networks (CNNs) has been in the 3D representation of building geometry. Due to the availability of both large 3D datasets and increase in the computational power, it is now possible to apply DL to understand specific tasks related to 3D data, such as segmentation, recognition, and correspondence (Ahmed et al., 2019). Parallel-trained 3DCNNs have demonstrated superior performance in 3D object classification (Sedaghat et al., 2017). 3DCNNs can also be used to distinguish defect features from geometric data such as voxels with a high level of accuracy (Dizaji et al., 2019). They are usually directly trained on the task of interest with a large amount of data, and the weights are updated at every iteration. The trained weights of the 3DCNNs are used to predict the outcomes and superior performance can be achieved within a short time. Researchers have recently attempted to improve 3DCNN models by combining the weights of multiple different 3DCNN architectures (Dolz et al., 2019; Wang et al., 2020).

2.2.3 ANNs model architecture

2.2.3.1 Feedforward neural networks

DL models are a class of models that can learn a hierarchy of features by constructing high-level features from the low-level ones (Goodfellow, 2016). Deep neural networks can identify the best weights set to produce the desired output by iterating over the data. Deep neural networks perform well when predicting nonlinear functions with multiple hidden layers, because capturing physical phenomena and environments of buildings requires the training of well-structured neural networks. Appendix B lists the key terminologies for this section.

Several studies have estimated the surface solar radiation on building façades using DL (Mohandes et al., 1998; Yadav and Chandel, 2014; Voyant et al., 2017). The estimated values can be used to predict a building's energy consumption. In industry and academia, the physical sky model (i.e., Hulstrom, 1981 and Perez, Seals, and Michalsky, 1993) can be used to evaluate daylighting performance and energy consumption (EnergyPlus). The replacement of a physical model with a DL model is a new approach for BPS, the benefits of which include reproducibility, time-efficiency, and scalability. DL models can reduce the time complexity of calculating optimized design solutions during the iterative simulation process and thus are relevant in the early design stages.

2.2.3.2 Convolutional neural networks

The utilization of Convolutional neural networks (CNNs); has enabled the three-dimensional (3D) representation of building geometries. The difference between a CNN and a regular neural network is that the former has one or more convolution and pooling layers. A convolution layer uses a filter matrix, typically of the 2×2 or 3×3 form, to perform the convolution operation and

obtain a convolved feature map after applying a filter. The filter in a convolution layer allows a neuron to receive its input from multiple units in the preceding layer. The convolution operation in the neural network has two main benefits. First, the units' number in the network reduces because its mapping is many to one; thus, the chances of overfitting decrease as the model is simpler than a regular neural network. Second, the characteristics of small neighborhoods can be captured, which is crucial in BPS because the CNN can capture the characteristics of different areas within a building. A pooling layer reduces the dimensionality of the feature map. For example, it can combine the output of four adjacent neurons into a single neuron, decreasing the dimensions of the feature map. A pooling layer uses different filters to identify an image's different parts, such as edges and corners.

The availability of large 3D datasets and substantial computational power has enabled the application of DL in learning specific tasks related to 3D data, such as segmentation, recognition, and correspondence (Ahmed et al., 2018). A parallel-trained 3DCNN demonstrated superior performance in 3D object classification (Sedaghat et al., 2017). 3DCNNs can also be utilized to distinguish the defect features in the geometric data (i.e., columns in buildings), such as voxels with high dimensionality (Shafiei Dizaji and Harris, 2019). Training of 3DCNNs is generally performed directly on the task of interest with a large volume of data, with weights updated at every iteration. The use of the trained weights is to predict the outcome, typically demonstrating better performance in a shorter period. Recently, researchers have attempted to improve 3DCNN models by combining the weights of multiple 3DCNN architectures. There is the use of an ensemble of different 3DCNN architectures in the medical and healthcare industries. An ensemble of densely connected 3DCNNs (3D-DenseNets), for example, is used to improve the diagnosis of Alzheimer's disease and mild cognitive impairment (Wang et al., 2019).

2.3 Survey of computer vision tasks for 3D data representation

DL has recently gained popularity owing to its state-of-the-art performance in different tasks related to texts, images, and sounds. Due to its wide applicability in research (e.g., 3D data processing and modeling), efforts to solve such scenarios are becoming increasingly necessary. It must be noted that it is currently possible to apply DL to 3D data-related tasks such as image segmentation, recognition, and motion translation. The current review surveys methods of applying DL to 3D data and provides rich information about the entire geometry of sensed objects and views.

The purpose of this review is to guide researchers in geometric modeling fields such as architectural design, building performance modeling, and object recreation with a better understanding of 3D computer vision tasks.

2.3.1 History of computer vision tasks

When computer vision first emerged in the early 1970s, the early pioneers of artificial intelligence believed that solving visual problem would be a meaningful path leading to higher-level reasoning and planning (Szeliski, 2010). Computer vision began with the desire to recover the 3D structure of the world from 2D images and use this technology as a stepping-stone toward gaining a complete understanding of the world (Azeriel Rosenfeld, 1976; Azriel Rogenfeld, 1966; Azeriel Rosenfeld, 1976). Efforts to design successful vision algorithms have led to investigations in the 1970s on topics of edge detection (Davis, 1975), object recognition (Baumgart, 1974; Baker, 1977), qualitative understanding of image formation, intrinsic images (Barrow and Tenenbaum, 1981),

and quantitative approaches such as optical flow (Enkelmann and Nagel, 1986) and motion structure (Longuet-Higgins, 1987).

The development of sophisticated mathematical formulae for image analysis began in the 1980s. Image pyramids (Adelson et al., 1984) started to be widely adopted for performing image blending and coarse-to-fine searches, further improving the concept of scale-space processing (Witkin, 1984; Andrew Witkin, 1987) and wavelets (Mallat, 1989; Bijaoui and Giudicelli, 1990; Epstein, Hingorani, and Czigler, 1992). Researchers developed a mathematical framework and more robust models using regularization (Terzopoulos, 1983 and 1988; Poggio, Torre, and Koch, 1985). Around that period, (Geman and Geman, 1984) introduced discrete Markov random field (MRF) models that enabled the use of higher performance search and optimization algorithms. MRF algorithms resulted in the use of Kalman filters in computer vision and other related topics, including regularization, flow, stereo, and high-level vision. More importantly, the investigation of 3D range data processing for acquisition, merging, modeling, and recognition continued.

With the development of projective reconstruction, factorization, and global optimization techniques, fully automated 3D modeling systems became available (Beardsley, Torr, and Zisserman, 1996; Brown and Lowe, 2003; Snavely, Seitz, and Szeliski, 2006). The exploration of the topics mentioned above took place and continued in the 1990s, and investigations on some topics such as optical flow were actively pursued and completed. A popular research topic involving 3D data was the multi-view stereo algorithms used for producing complete 3D surface model with 3D volumetric data from binary silhouettes (Srinivasan, Liang, and Hackwood, 1990; Seitz and Dyer, 1999). The first statistical learning techniques appeared in the 1990s, including the application of principal component analysis in facial recognition (Turk and Pentland, 1991) and linear dynamic systems for curve tracking (Isard and Blake, 1998).

In the 2000s, the interplay between vision and graphic fields was illustrated by the rubric of image-based rendering and high dynamic range image capture (Mann and Picard, 1996; Paul Debevec, 1998). these topics was computational photography. A second noticeable trend in this decade was the emergence of object recognition combined with learning (Ponce et al., 2006) and the development of more radical algorithms for comprehensive global optimization problems (Szeliski et al., 2008). During this decade, they recognized machine-learning techniques applied to computer vision as a synergistic field of research (Freeman et al., 2008). Most of the related research focused on different tasks with 2D applications for classification (Krizhevsky et al., 2012), segmentation (Shelhamer et al., 2017), detection and localization (Sermanet et al., 2014), recognition (He et al., 2016), and scene understanding (Farabet et al., 2013) using ML and DL. Due to the tremendous amount of 3D data and advanced sensing technologies available, the active use of DL for 3D data is now a crucial topic in computer vision. Recent investigations in the field of 3D computer vision involved the use of DL architecture for processing 3D data. The following chapter describes the ongoing advancements in the DL architecture for 3D data, along with applicable models and methods.

2.3.2 Data representation in computer vision

Figure 2.3 shows the different ways of representing the physical properties of geometric data for ANNs in both Euclidean and non-Euclidean realms. The application of DL to 3D data is not yet well established. Fortunately, with recent advances in sensing technology and data acquisition devices and techniques, the amount of training 3D data for DL has increased tremendously.

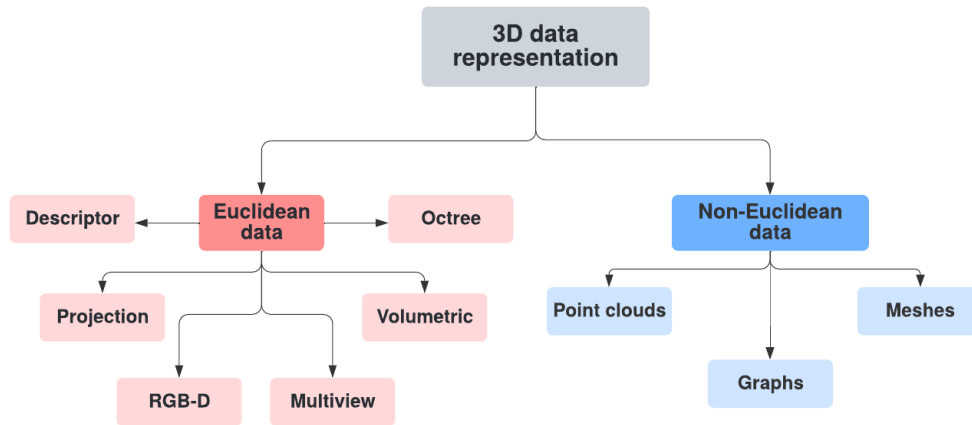


Figure 2.3: 3D data representation of training data-driven models

The availability of 3D data for addressing computer vision problems related to understanding 3D scenes and real-world phenomena continues to improve. Points' reduction, data structuring, and hardware exploitation are predominant issues in the conversion and generation of data for DL training (Ioannidou et al., 2017). Methods of 3D data representation are valuable research areas, especially with the rapid evolution of powerful GPUs and their recent successes. 3D data can have a unique structure, and their geometric properties vary with the application of different DL methods and experiments (Ahmed et al., 2018). 3D data classification can be into two distinct domains: Euclidean and non-Euclidean.

Simplified 3D Euclidean data have a grid structure, whereas 3D non-Euclidean data do not. The 3D Euclidean properties are an extension of the 2D structure of the traditional DL architecture; thus, they are more suitable for processing regularly populated data such as voxel data of generic objects. When the geometries become complicated, and the data are collected in an irregular manner, spatially distributed sensors such as LIDAR data clouds, 3D non-Euclidean properties such as point clouds or graphs are adequate for representation with minimal deformations (Bronstein et al., 2017).

2.3.2.1 3D Euclidean

The primary 3D data representations that fall into the Euclidean category are descriptors, projections, RGB-D, voxel, octree, and multi-view data.

1) Descriptors

A shape descriptor is a simplified representation of 3D geometry as an array containing a set of numerical values or a graph-like structure describing the shape geometrically or topologically (Mingqiang et al., 2008). With descriptors from a shape's geometry, topology, surface, texture, or any other descriptive features or combination of features can be explained (Ahmed et al., 2018). Since the 1990s, 3D shape descriptors have been widely used in 3D search engines and sketch-based modeling (Kazmi et al., 2013). Classifications of 3D shape descriptors include those based on views, histograms, transformations, or graphs, and hybrid 3D descriptors (Zhang, João, and Ferreira, 2004). View-based descriptors use silhouette, grayscale, or extracted images from multiple 3D objects views. Histogram-based descriptors collect feature-based domains of 3D shapes in different bins as numerical values. The basis of the development of transform-based is on classical image processing methods such as Fourier transform. Graph-based descriptors delineate the topology of a 3D shape and object in the form of a graph or tree structure. This is beneficial due to its capacity to represent multiple levels of detail in local geometry. Lastly, hybrid 3D shape descriptors combine several algorithms to produce better quality 3D shape retrieval and analysis.

2) Projections

Projecting 3D information onto a 2D plane is a unique representation of 3D data where the projection converts the object's information into a 2D grid with descriptive features. Image-based

techniques are convenient when analyzing large-scale data for pre-training (i.e., ImageNet). The proposal for spherical and cylindrical projections have been frequent in the literature (Shi et al., 2015). This method efficiently produces a well-structured grid array in the Euclidean world; however, with a limited capacity to store complicated 3D vision tasks and related information due to the information lost in projection (Leibe et al., 2016).

3) RGB-D

With the advent of commodity depth cameras such as the Asus Xtion Pro Live and Microsoft Kinect, combining 2D visual and 3D depth information is now being explored (Afzal et al., 2014). RGB-D data provide 2.5D information by illustrating a depth map along with the RGB color information. The use of this method is to reconstruct scenes and poses (Fanelli et al., 2011), recognize feature identities (Erdogmus and Marcel, 2013), and provide correspondence (Zollh et al., 2013). The use of many multi-camera techniques has been to capture incoherent geometry and human details at high resolution due to the technology's capacity to store much larger amounts of data; compared to other 3D datasets such as point clouds (Firman, 2016).

4) Multi-view

Multi-view data are commonly used in real-world applications due to their efficient structure. Data points' collection is via different methods and may require presentation as a combination of multiple 2D views. Multi-view learning can increase datasets by employing manually generated topologies and 2D views (Zhao et al., 2017). A simple way to generate such representations is by creating multiple 2D image descriptors per view, and then using that collection for learning (Su et al., 2015). However, the view numbers adequate to model a 3D shape are still questionable. Despite this uncertainty in multi-view modeling, Su et al. (2015) showed that well-represented multi-view data performed better than 3D volumetric data when recognizing shapes. Due to the

minimum storage required for 2D data: the use of the multi-view technique is to recognize and segment objects in the current DL practice.

5) Voxels

A regular grid in a 3D matrix can be used to represent and reconstruct 3D. Voxels are useful to represent 3D data by mapping and describing how is the distribution of 3D objects through an entire 3D scene. Detailed information about 3D shapes can reside in voxel structures as a form of an array or set of lists. Detailed information about objects and contextual details can be estimated without losing the original properties using 3D voxelated object representation (Xiang et al., 2015). Therefore, it has been used in mapping point clouds, human features, video sequences, and MRI scenes (Qi et al., 2017; Maturana and Scherer, 2017; Kamnitsas et al., 2017). Voxel-based methods rasterize 3D shapes as an indicator or sampled distance function over dense voxels and apply a 3DCNN over the entire 3D volume. Because the memory and computation costs grow cubically as the voxel resolution increases, these methods can become computationally expensive. That is why voxel grids are not suitable for representing high-resolution data or densely packed areas on a sparse matrix (Abdul-Rahman and Pilouk, 2008; Tatarchenko, Dosovitskiy, and Brox, 2017).

6) Octrees

Octree-based 3D volumetric representation is more efficient than voxels. Octree 3DCNN significantly reduces the computation time with a sparsely sampled matrix (Wang, et al., 2017). In Wang's model, octrees provide a generic and efficient 3DCNN solution for 3D shape analysis. Because octrees grow quadratically as the octree depth increases, they are suitable for analyzing high-resolution 3D models. Voxels and octrees have the same structure, but octrees are more

robust in representing the fine details of 3D objects (compared to voxels) and have the ability to share the same values for large regions of space (Ahmed et al., 2018).

2.3.2.2 3D non-Euclidean

Many scientific studies use data with underlying non-Euclidean spaces, such as sensor networks in buildings, functional networks in brain imaging, and meshed surfaces in computer graphics. Three main types of non-Euclidean data are point clouds, 3D meshes, and graphs.

1) Point Clouds

The increasing use of 3D data acquisition devices such as structured light scanners, Kinect, and time of flight, has made point clouds increasingly available. Point clouds classification can be non-Euclidean geometric data due to their sparsity and relatively large scale. Applying DL to point clouds comes with many challenges, including irregularity, a lack of structure, and point misalignments. Despite such limitations, point clouds' uses are in multiple computer vision tasks such as 3D reconstruction, object recognition, and vehicle detection, with efforts to filter 3D point clouds from noise while preserving original geometric data (Han et al., 2018).

2) Meshes

Meshes are one of the most popular topics in computer graphics when discussing 3D shapes. Meshes consist of a set of polygons called faces with a set of vertices that describe their associated connectivity. Meshes classification can be as Euclidean space on a local scale; however, on a global scale, meshes are non-Euclidean due to certain characteristics such as shift-invariance, operations in vector space, and global parametrization systems. Learning global meshes can be challenging because DL methods cannot easily adapt irregular representation as input.

3) Graphs

The representation of 3D mesh can also be a graph structure, where directing of the graph nodes and corresponding vertices of the mesh and edges represent the connectivity between these vertices. Because graphs contain complicated structures containing rich underlying information, traditional DL architectures have been used for analyzing graph data (Zhang, Cui, and Zhu, 2020). The combined use of mesh and graph structure is promising to tackle the multi zone BPS tasks.

2.3.3 DL architecture for 3D vision tasks

Over the last decade, 3D DL has enabled considerable advancements the field of computer vision. Unlike 2D vision tasks (Farabet et al., 2013; He et al., 2016; Krizhevsky et al., 2012; Shelhamer et al., 2017; Noh, Hong, and Han, 2015), 3D vision tasks are not straightforward due to the geometric complexity of 3D objects and data loss and distortion resulting from different representation types. The previous section discussed various representation types for 3D vision tasks, including 2D and 3D representations. However, 2D representations of 3D objects sometimes omit comprehensive information about the object and associated parameters.

Further discussions of the intrinsic nature of 3D building data and related building attributes will be below, including applications of 3D DL to geometric data in volumetric, mesh, and graph formats. Table 2.2 summarizes the advantages and limitations of each data representation for DL.

Table 2.2: Summary of 3D data representation for DL modeling and training

3D Euclidean		3D non-Euclidean		
Voxel	Octree	Point Cloud	Mesh	Graph
Features				
Volumetric data in grid format	Varying size of voxels data	3D points	Patched filters	Nodes and edges

Advantages				
Full volumetric description and information delivery of the 3D object	Efficient 3D volumetric representation	Applicable to any type of sensor data	Mostly used for preserving graphical properties of 3D object	Models for graphs can be used for mesh representation
Limitations				
Large memory requirements and high computation time	Geometry data does not preserve intrinsic part of 3D object	Irregular sampling and potential noise and missing parts	Difficult to train because of irregularity and complexity	DL models for meshes cannot be used for graphs

2.3.3.1 DL architectures for 3D volumetric data

Unlike real-world data (i.e., point clouds), most simulated data tend to be populated in a grid-like structure. Regularly populated points can be represented in the form of an array, such as of pixels or voxels. Rasterized images consist of pixels. A volumetric pixel (i.e., a voxel) is the 3D equivalent of a pixel, a distinguishable element of a 3D object. It is a volume element representing a grid structure, and its use in 3D DL is to exploit the full geometry of an object. ShapeNet (Zhirong Wu et al., 2015) was the first DL model to use a 3D geometry represented as voxels. The form of input is a 30 x 30 x 30 binary tensor indicating whether the voxel is a part of the target object or not. VoxNet (Maturana and Scherer, 2017) adopted a 3DCNN to recognize 3D objects in different 3D data representations, including RGB-D, LIDAR point clouds, and 3D CAD models. The convolution in VoxNet uses and follows 3D filters, and the network architecture is composed of two 3D convolutional and two fully connected layers. The input construction is as a volumetric occupancy grid of 32 x 32 x 32 voxels. Because of the training, VoxNet outperformed ShapeNet when tested on ModelNet10. (Sedaghat et al., 2017); enhanced the architecture of VoxNet to add orientation estimation as an auxiliary parallel task. For this experiment, they used a slightly

deeper network that equipped four 3D convolutional layers. This fortified the classification results for the ModelNet10 datasets.

An important contribution was the use of an auto-encoder structure in unsupervised learning to learn the embedding of 3D objects. (Sharma, Grau, and Fritz, 2016) developed a convolutional volumetric auto-encoder (VConv-DAE). The proposed networks learn 3D volumetric representations efficiently by estimating the occupancy grid of 3D data. VConv-DAE is promising for shape completion and high-performance shape interpolation. (Brock et al., 2016) proposed Voxception-ResNet (VRN), a deep 3DCNN. VRN is composed of 45 layers of deep CNN that require data augmentation to avoid overfitting. VRN shows a 51.1 percent performance improvement over VoxNet in ModelNet classification. However, deep neural networks are susceptible to overfitting and computational inefficiency in training.

A 3DCNN converts 3D shapes to sample representations and regularly applies a CNN to them. Voxel-based methods rasterize 3D shapes as an indicator or sampled distance function over dense voxels and apply the 3DCNN over the entire 3D volume. Because the memory and computation costs grow cubically as the voxel resolution increases, these methods can become tremendously expensive.

To solve this problem, LightNet (Zhi et al., 2017) was developed as a real-time volumetric CNN designed for ModelNet recognition. LightNet can achieve a faster convergence with fewer parameters and includes two main learning tasks: class labels and orientations. LightNet outperformed VoxNet by about 24.25 percent on ModelNet10 and ModelNet40 with 67 percent fewer parameters than VoxNet. According to Wang et al. (2017), Octree 3DCNN significantly reduced the computation time with a sparsely sampled matrix. In Wang's model, octrees provide a generic and efficient 3DCNN solution for 3D shape analysis. Because octrees grow quadratically as the octree depth increases, they are suitable for analyzing high-resolution 3D models.

2.3.3.2 DL architectures for 3D mesh data

A convolution operation is performed by passing a filter on a spatial domain and recording the correlation between the template and filters. This is because of the shift-invariant and global parameterization properties of Euclidean spaces. Various non-Euclidean CNN frameworks have recently been proposed to overcome the lack of shift-invariance and global parameterization in meshes.

Masci et al. (2018) demonstrated the first CNNs on triangular meshed surfaces. Geodesic convolutional neural networks (GCNNs) are a generalization of the convolutional networks for application to non-Euclidean manifolds. They extract patches as local systems of polar coordinates and pass through a cascade of filters and operators (Masci et al., 2018). This approach is innovative but has some drawbacks. It is applicable only to triangular meshes, the radius of the constructed geodesic patch is small for the actual shape, and rotations of the convolution filters are expensive. Anisotropic CNN (ACNN) extends the applicable geometry to deformable shapes and graphs (Boscaini et al., 2016). ACNN frameworks construct simple local patches independent of the size of the mesh. The MoNet framework proposes a general construction of patches by defining a local system of pseudo-coordinates around each vertex with weight functions (Monti et al., 2016).

The most recent achievement in geometric DL is spline-based convolutional neural networks (SplineCNNs). Proposals for SplineCNNs have been for irregularly structured geometric data because they eliminate the need to define patches on meshes or graphs. This framework outperforms other models on correspondence tasks due to the local support of the B-Spline basis. This framework allows for the use of any dimensionality and geometry on training input.

2.3.3.3 DL architectures for 3D graph data

3D data are not always available in a grid-like format (i.e., modeling meshes in CFD simulations, LIDAR point clouds). Non-Euclidean approaches convert irregularly populated 3D data for main DL operations such as convolution. Graph-based 3D data leverage the spectral properties of graphs to define intrinsic descriptors for DL training. With the success of CNNs in computer vision tasks, efforts have been directed towards generalizing CNNs to irregular structures. For instance, meshes in 3D geometries such as surface tension or temperature measured from nodes on discretized meshes can be converted to graphs and then used for training CNNs, instead of 4D tensors or more layers with spatial coordinates. Graphs can generalize natural frameworks for low-dimensional grid-like structures, allowing efficient forward propagation with large numbers of datasets.

Geometric DL is an effort to generalize DL models to non-Euclidean structures such as graphs and manifolds (Bronstein et al., 2017b). GCNN consists of two main methodological approaches: spectral and spatial filtering methods. Both are categorized based on how filtering is applied and how locally processed data are combined. Bruna et al. (2014) proposed spectral filtering methods operating on graph structures using CNNs. These define a convolution-like operator using the spectral eigen decomposition of Laplacian graphs (Bruna et al., 2014). This filtering operation implies that convolution is a linear operator that communicates with the Laplacian operator. However, Bruna's basis-based learning methods are limited to specific bases and can be computationally expensive. To overcome the limitations of Laplacian eigen bases and limited incompatibility of different shapes, quasi-harmonic bases have been introduced by constructing a compatible orthogonal basis across various domains through a joint diagonalization (Kovnatsky et al., 2013). However, this method requires prior knowledge about the correspondence between two distinct domains.

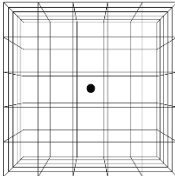
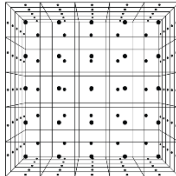
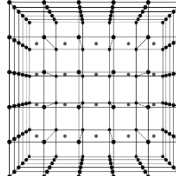
Local spectral filtering of graphs using a polynomial expansion was proposed by Defferrard, Bresson, and Vandergheynst, (2016) and Kipf and Welling, (2017). With the success of local spectral filtering methods, Wang, Samari, and Siddiqi (2018) proposed a local graph convolution with a novel graph pooling strategy to process unordered point clouds. They applied the spectral filtering method to pointNet++ and then replaced traditional max pooling with recursive cluster pooling. This method does not require pre-computation and can be trained in an end-to-end manner to achieve better performance than existing techniques.

Traditional ML applications cope with graph structures by mapping 3D information to a simpler representation such as vectors of reals (Singaravel et al., 2018a). Graph neural networks (GNNs) utilize spatial filtering to preserve graph-structured data via the weights of the graph (Scarselli et al., 2009). Learning graphs with the spatial filtering method are dependent upon each vertex's neighborhood. The graph topology (i.e., the spatial structure of the input graph) is maintained when the feature vectors from the neighborhood nodes are aggregated. Scarselli et al. (2009) used a simple diffusion function on the recurrent neural network (RNN) structure to repeatedly propagate node representation until it was stable and convenient. Next, gated graph neural networks were proposed to alleviate the computational cost of GNNs by performing updated states to learn the optimal graph representation using gated recurrent units (Li, Zemel, Brockschmidt, and Tarlow, 2016). To overcome the limitations of graph-based RNNs, graph convolutional neural networks were proposed, where the nodes of the graph are mapped to points in Euclidean space, for the compact representation of graphs (Bruna et al., 2014; Zonghan Wu et al., 2019). Following the work of Bruna et al. (2014), hyperbolic graph convolutional neural networks have been delineated, showing improved performance with less distortion when applied to real-world graphs (Chami et al., 2019).

2.3.4 3D data representation for BPS tasks

3D data representation methods are categorized and compared in this section (Table 2.3). Understanding the proper categorization of 3D data and relevant DL methods is essential for applying DL in performance simulation and architectural modeling.

Table 2.3: 3D data representation for building performance simulation

	BPS 3D representation		
BPS task	Energy	Daylighting	Airflow
			
Data type	Point	Grid	Mesh/Graph
Representation	Point coordinates	3D voxels	Nodes and edges

In practice, three main types of data exist in fields related to building performance simulation: building sensor data, building stock data, and building simulation data (Westermann and Evins, 2019). Building sensor and stock data usage is to optimize buildings' operations, and the simulation data usage is in the building's design and documentation. Sensor networks can be represented as a form of Point Clouds, and the building's stock data can be coded in 2D arrays. Both data require complicated input layers such as system configurations, occupancy information, and other engaging factors. Therefore, the development of a proper input representation method for DL on those kinds of data is essential.

Especially, during the early design stage, the use of building simulation tools has become popular. These devices have been widely adapted and evaluated. Daylighting, energy, and airflow are the commonly used performance simulation metrics to assess the built environment. Each simulation tool deals with the different 3D data formats (Table 2.3). Since building geometry is mainly

computed as 3D coordinates in CAD interface, DL is required to delineate the 3D representation of building geometry.

This survey revealed that the 3D representation methods played a crucial role in evaluating the performance of neural networks. For example, voxel (Maturana and Scherer, 2017; Sedaghat et al., 2017; Shafiei and Harris, 2019; Wu et al., 2015) is a common way to represent the 3D object; thus, its use can be as an input structure to train 3DCNNs. This type of representation has validated its efficiency in predicting grid-based simulations such as radiation intensities on building facades. However, the high computation time of a voxelated matrix causes the quadratic growth computation time during model training. Therefore, octree (Tatarchenko et al., 2017; Wang et al., 2017) structure can be mapped into 3DCNNs to increase training efficiency. Another method is applying graph neural networks (Chami et al., 2019; Li et al., 2016; Scarselli et al., 2009) to mesh-like geometry. This method is suitable to simulate the airflow around buildings since it effectively incorporates mesh representation of a 3D input.

The 3D representation methods discussed in this survey are especially beneficial for modeling a 3D-built environment and the physical phenomena around buildings. Although key features and advantages of using those networks over the other were highlighted, there is no absolute winner among the techniques introduced. Unlike 2D, 3D DL is still developing and maturing. Therefore, a series of experiments on the same BPS tasks is recommended for future researchers. There have been many methods proposed across different disciplines. VoxNet, Octree, and GCNNs outperformed others for shape analysis tasks but have an immense potential to expand to other 3D-related tasks. Therefore, proposed methods can be applied to the geometric DL for BPS tasks.

Chapter 3

Methodology

This chapter discusses the main strategies for integrating data-driven BPS tools in building modeling and simulation workflow and the complementary strategy for the deployment.

3.1 Development of new BPS workflow with ANN model

Traditionally, many stakeholders are involved in the BPS consulting process including architects, BPS modelers, and BIM modelers. During the design consulting process, architects manage building design options in CAD software based on the concept and programs of the buildings. With the increased use of BIM documents in the design process, the BIM modeler mainly focuses on conveying the appropriate information for future documentation. The BPS modeler takes the model information and creates a BPS workflow for the analysis results for the different performance metrics. Figure 3.1 shows the traditional workflow and data transfer between different experts in performance-driven design and consulting. Due to the expertise-dependent nature of physics-based BPS tools, the discrete workflow is inevitable in practice.

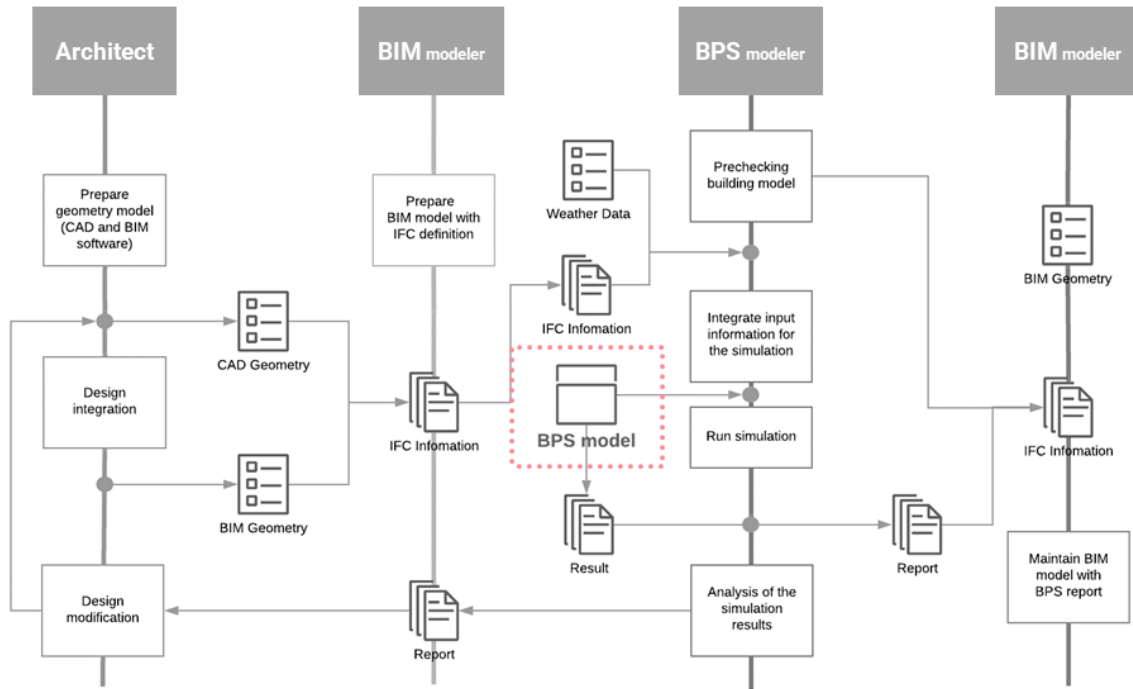


Figure 3.1: Traditional framework for BPS design-consulting process

Replacing physics-based BPS models with data-driven ANN models can potentially reduce the number of steps between each expertise in the early design decision-making process. Recent ANN models can easily integrate into the CAD platform with less dependency. Therefore, they can be directly integrated into the modeling workflow for architects.

Figure 3.2 represents the new workflow with ANNs models used as an alternative to existing physics-based models. To integrate ANNs models into the modeling workflow, it is necessary to develop data exchange protocol and methods for CAD and BIM software using their API. The different type of data processor to complete this workflow in this new framework will be discussed in this and following chapters. To realize this, it is necessary to look at the required data format for CAD and BIM with the existing BPS models. BIM-based geometry includes the detailed definition of buildings and related information not fully used by other BPS tools, such as EnergyPlus, and CFD.

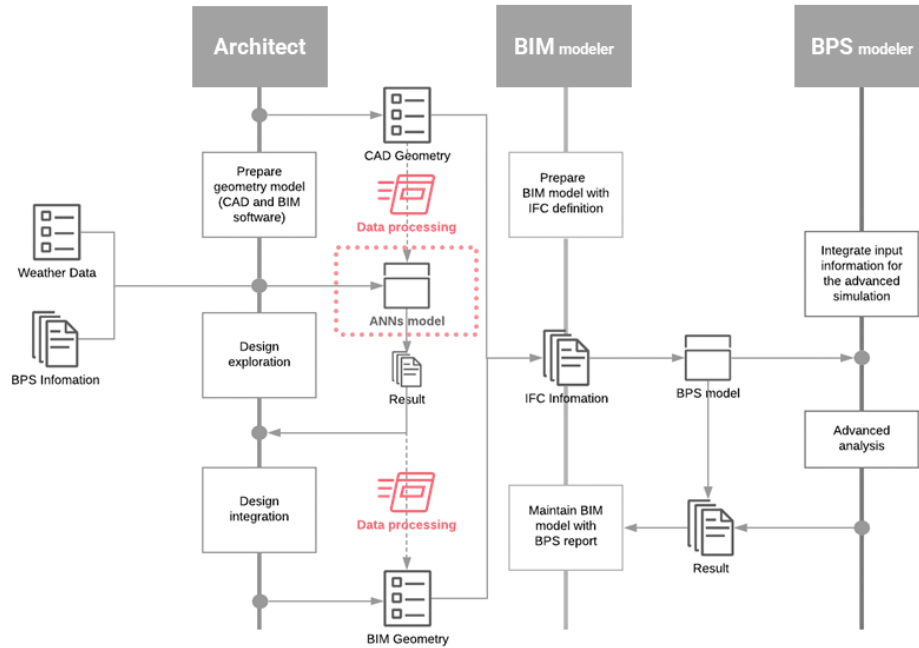


Figure 3.2: Proposed framework for BPS design-consulting process with ANN models

IFC-based building energy modeling tools define building geometry as a system of surfaces, also referred to as “space boundaries.” Examples of such surfaces include walls, slabs, roofs, columns, and beams (Bazjanac, 2010). Research by Autodesk (2015) has illustrated a mesh conversion process from Revit to CFD. Its use can verify the newly added CAD geometry and help convert a building’s geometry into meshes. The current practice of modeling for BPS simulation usually involves the manual recreation of a building’s geometry to represent various physical properties of buildings (such as thermal, acoustic, structural, and airflow), producing the ontology for BPS inputs of the building’s attributes for that simulation.

Conventional BPS tools require space boundary definitions or gridded arrays or mesh boundary conditions as input (Figure 3.3). Airflow and radiation mainly require the simulation parameters with point and mesh type representation as input; however, ANNs require new rules and topologies to explain a building’s attributes and related information. The proposed methods

generalized the specific representation of 3D building geometries for each BPS; with this effort, the general rule to create ANNs for taking buildings' geometry with performance metric can be illustrated.

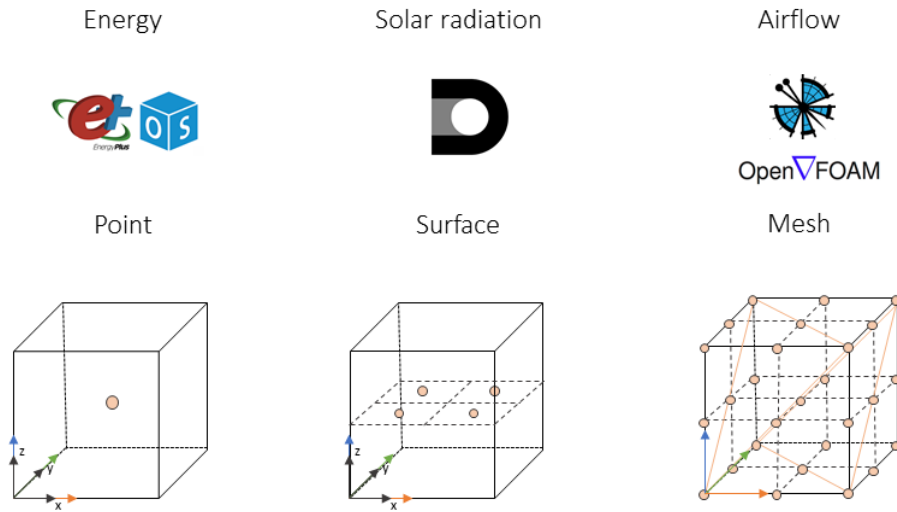


Figure 3.3: Inputs for modeling different performance assessment tools

The Euclidean data category mainly includes the following 3D representations: descriptors, projections, RGB-D, volumetric information, and multi-view data. The representations that fall into the non-Euclidean category are point clouds, graphs, and meshes. Three main types of data exist in fields related to BPS, building sensors, building stock, and building simulation information (Westermann and Evins, 2019). Building sensor and stock data usage is optimizing building operations, and their representation is as forms of non-Euclidean data such as point clouds. The utilization of simulation data in a building's design is in the form of Euclidean data such as volumetric grids (voxels).

3.2 Development of data exchange methods for proposed BPS workflow

The design of high-performance buildings using the CAD interface and interacting tools and surrogate models are complex and nonlinear. The sustainable nature of the process requires comprehensive guidelines to derive the necessary information efficiently throughout the design decision-making process. Figure 3.4 explains the ANN-driven BPS data processing flow for sustainable building design and compares it to the conventional method. Since data-driven building simulation software require different types of data representations as input, it is essential to find the adequate data conversion methods.

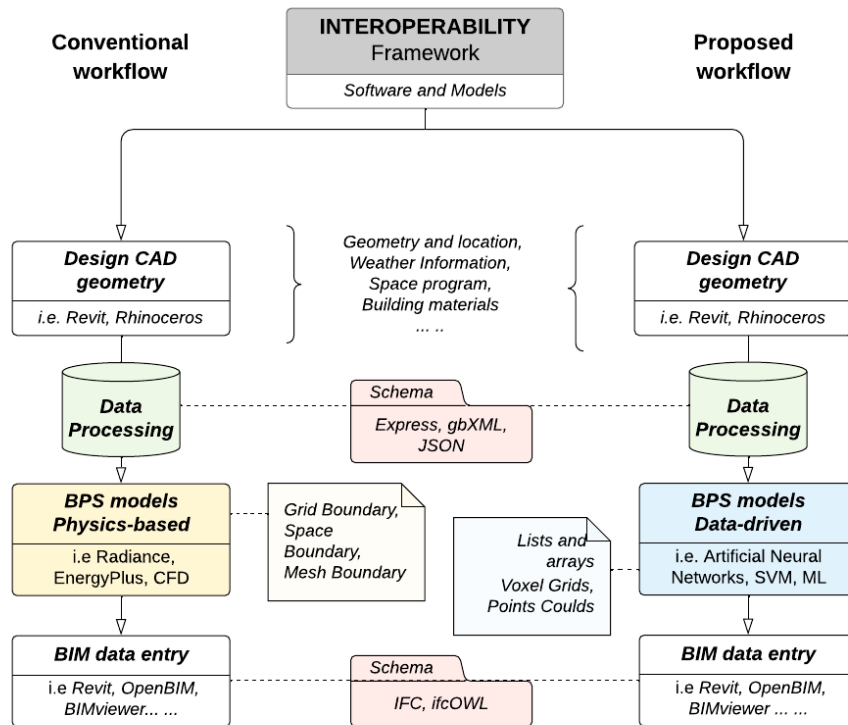


Figure 3.4: Comparison of BIM and BPS frameworks

Data conversion consists of two parts: pre-processing and post-processing. Pre-processing includes refined BPS requirements and geometry definitions for different data types (i.e., simulated and sensor data), and post-processing incorporates BIM requirements and performance metrics for entering the data into the BIM software. The ANN-based BPS interoperability framework covers data processing methods from the CAD software to ANN models (data pre-processing) and from ANN models to BIM documents (data post-processing).

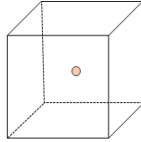
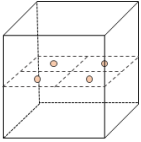
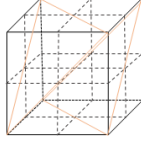
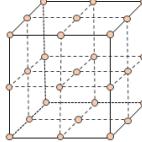
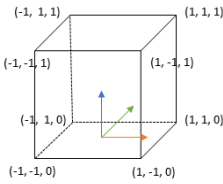
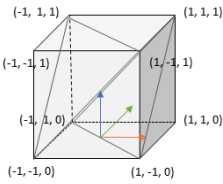
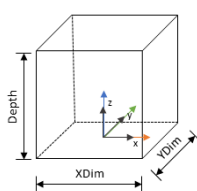
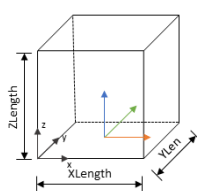
The application of ANN-based BPS tools with the data processing components during early design stages has become a promising approach. For these applications, such data modeling tools need to fulfil the basic functional requirement for BPS tasks for data transfer. Daylighting, energy, and airflow are the commonly used performance simulation metrics to assess the built environment. Each simulation tool deals with the different 3D data formats (Table 3.1).

Since building geometry estimation is mainly as 3D coordinates in the CAD interface, the rules for 3D representation of a building geometry for ANNs needs to be delineated. Table 3.1 lists representative data forms for buildings' geometry for the simulation tasks. For example, energy simulation requires a point-based descriptor for single-zone simulation or point-cloud type data for multi-zone simulation. For the radiation and airflow simulation, voxelated metrics that can interpret information from the mesh and grid-type geometry are required to further model ANNs to train networks.

In addition to converting the original CAD geometry for ANNs modeling, it is essential to preserve the original modeling properties for BIM documentation. Converting and preserving the buildings' information in a flexible workflow can demonstrate a sustainable design and consulting environment. As seen from the different geometry representations in IFC in 3.1, there are specific requirements for converting CAD geometry for BIM information. Therefore, in the new proposed

workflow, the integrated platform should have the capability to convert the buildings' model for ANNs analysis and preserve its modeling schema and rules simultaneously.

Table 3.1: 3D data representation for building performance simulation

BPS 3D representation			
Descriptor	Voxel/PointCloud	Mesh/Graph	Voxel/Octree
			
IFC geometry representation			
Brep/Surface model	Tessellated surfaces	CSG primitive	Swept solid
			

In general, the 3D representation methods played a crucial role in evaluating the performance of neural networks. For example, voxels (Maturana and Scherer 2017; Sedaghat et al., 2017; Shafiei Dizaji and Harris 2019; Wu et al., 2015) are commonly used to represent 3D objects; thus, they can be used as input structures to train 3DCNNs. This type of representation has validated its efficiency in predicting grid-based simulations such as radiation intensities on building façades. However, the high computation time of a voxelated matrix causes the quadratic growth in computation time during model training. Therefore, mapping the octree (Tatarchenko, Dosovitskiy, and Brox 2017; Wang et al. 2017) structure into 3DCNNs can increase training efficiency. Another method is applying graph neural networks (Chami et al., 2019; Li et al., 2016a; Scarselli et al., 2009) to a mesh-like geometry. This method is suitable to simulate the airflow around buildings as it effectively incorporates the mesh representation of a 3D input.

The 3D representation methods discussed are useful for modeling a 3D-built environment and the physical phenomena around buildings. Despite their individual advantages, there is no one best method among the techniques introduced. Unlike 2D DL, 3D DL is still maturing. Therefore, a series of ANN modeling on the same BPS tasks are recommended for future researchers. Many methods have been proposed across different disciplines. VoxNet, Octree, and GCNNs outperformed other tools for shape analysis tasks but have an immense potential to expand to other 3D-related tasks. Therefore, the proposed methods can be applied to the geometric DL for BPS tasks. Another possible direction is to continue to work on the mixed 3D representations with the described methods above.

To expand the scope of the ANNs architecture and improve the applicability of ANN models for 3D buildings, it is necessary to understand the structural properties of the different representations of 3D data; this is the focus of the present research. A comprehensive overview of ANN-based research on solar radiation and airflow patterns is provided in the next chapter to illustrate recent advances in ANN modeling of 3D data representations with an emphasis on the challenges emerging from the differences between various methods. In this study, experiments were conducted to develop an ANN-based BPS model and different 3D representation methods tested for solar radiation and airflow simulation. This stage led to the development of the data exchangers and ANNs models for the simulated data to calculate the annual radiation intensities on buildings façades and indoor airflow patterns with different input air conditions.

Chapter 4

Experiments

This chapter discusses two main experiments for learning models used for BPS. The first is solar radiation simulation, a method for developing ANN models for solar to simulate radiation mapped on the buildings' façade and indoor spatial daylight autonomy. The second is a version of airflow simulation to predict indoor airflow patterns using different ANN models. Followed by experiments, the algorithms implemented to converting the buildings' geometry data for ANN models are explained. These experiments and findings are used in Chapter 5 and 6.

4.1 Development of ANNs for different BPS tasks

4.1.1 ANN-driven solar radiation simulation

For over a decade, Radiance-based daylighting simulation, a state-of-the-art backward ray tracer, has been used to physically validate a range of building geometries and shading devices. Radiance is the most widely used daylighting simulation tool due to its accuracy, the data-validation capabilities, and the requirements for the industry standard. Using a physics-based Radiance

engine (i.e., DIVA), ANN models were trained to predict radiation intensities on building façades with thousands of reference geometries represented in CAD software (i.e., Rhinoceros). CAD-based tools enable quick creation of a building's geometry. Radiance-based tools require defining the geometry of a building in a 3D coordinate system. Interoperability between CAD and Radiance-based daylighting simulation tools is necessary to train the ANNs to serve as BPS engines. ANN architectures using 2D data illustrate the requirement for a considerable amount of training data. Thus, applying ANNs in a 3D domain is not as effective as in 2D. However, 3D data provide rich information about the full geometry of 3D objects and their environment. Therefore, 3D computer vision tasks require a way to incorporate 3D information when training neural networks. In the experiments outlined below, a new method of transferring the 3D coordinate system for CAD software into inputs for ANNs was investigated.

Objectives of ANNs-solar experiments are to analyze the use of ANN applications and BPS tasks achieved by 3D data representation (i.e., using voxels) and provide different data conversion techniques for predicting solar radiations on buildings façades; and to test the applicability of methods involving physics-based radiation simulation trained using 3DCNNs and different encoding methods of inputs for training 3DCNNs.

The solar radiation studies mainly consisted of DL methods as a means of representing the annual radiation intensity and the level of exposure of buildings without using physics-based engines with different input parametrizations. Such representations are obtained by combining information from all channels and are thus adequate for analyzing certain phenomena relating to indoor spaces. Many studies have used DL to estimate surface solar radiation on building façades (Mohandes et al., 1998; Yadav and Chandel, 2014; Voyant et al., 2017). In particular, CNN have recently been used in the 3D representations of building geometries. Due to the availability of both large 3D datasets and computational power, it is possible to apply DL to specific tasks related

to 3D data, such as segmentation, recognition, and correspondence (Ahmed et al., 2018). DL models are usually directly trained for the task of interest using a large amount of data with the weights updated for every iteration. The trained weights for 3DCNNs are used to predict the outcome, usually demonstrating superior performance in a short period.

In this solar radiation study, 3DCNNs were used to predict solar radiation on building façades. Because 3DCNNs require a voxelated matrix as input, different methods of mapping the relevant geometric information into the voxel format were investigated.

This section consists of three parts: data generation, data preprocessing, and 3DCNN modeling and validation. Conventional modeling and simulation software tools were used in all steps: Rhinoceros and Grasshopper for parametric modeling; DIVA in Rhinoceros for radiation simulation; Python3 packages, such as NumPy, SciPy, and Matplotlib, for data processing and visualization; and TensorFlow and Keras for modeling and validation. Figure 4.1 describes the general workflow along with the data structures and ANNs modeling methods used.

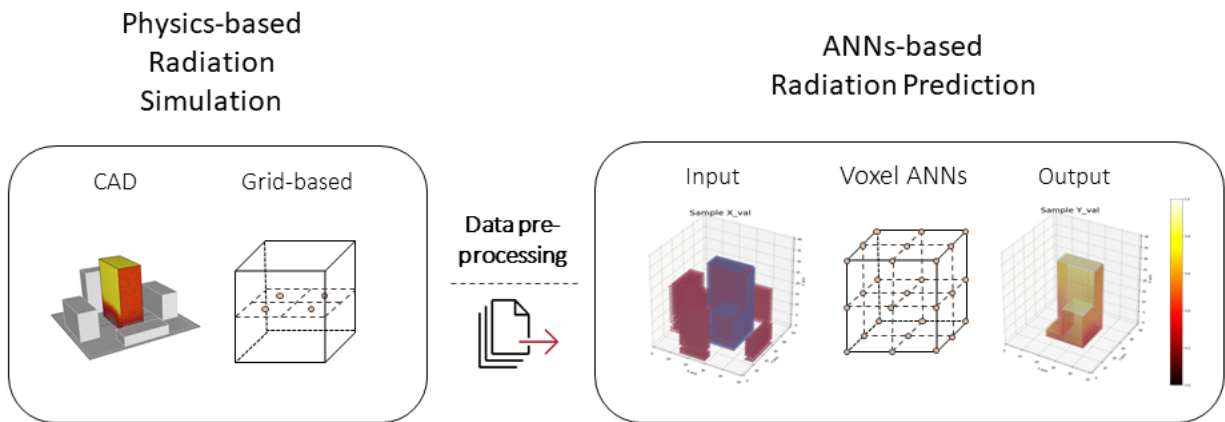


Figure 4.1: Generic workflow and related software

4.1.1.1 ANNs-solar experimental model: ARINet

The objectives of the development of ARINet (Han, 2020) were to design an ANN architecture using 3D building geometry and simulated data; determine a feasible method of encoding building geometry as input for ANNs, representing the physical properties of buildings such as radiation intensities; and define the voxelated properties of building geometry as input for ANNs.

The initial challenge when converting extracted data into a 3D voxel representation was to match the different coordinates to their boundary conditions. Because the output from DIVA (a physics-based radiation simulation engine) for Rhino's grid system cannot evenly distribute the local coordinates of building façades, pre-processing was necessary to control the points and values representing all radiation values equivalent to each sub-cube. Therefore, the edge values and voxel map created to represent the 3D information for radiation received were neglected. Figure 4.2 shows the process of input modeling for ARINet.

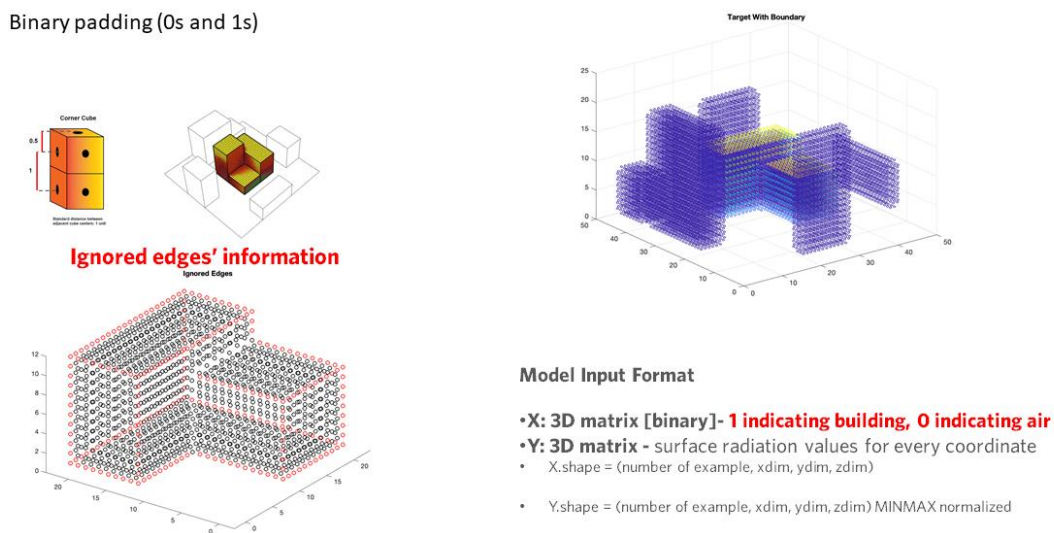


Figure 4.2: Binary padded voxel matrix (ARINet)

After processing the edge data, the model input, denoted with an X, was padded with binary information (i.e., 1s and 0s). In this voxel representation of the 3D space, one represented a

building, and zero indicated air. The predicted values that comprised the model output, denoted with a Y , could then be the surface radiation values for every coordinate. The structures of X and Y are as follows:

- X .shape = (number of example, xdim, ydim, zdim)

- Y .shape = (number of example, xdim, ydim, zdim) / (MINMAX normalized)

After processing each voxel shape, the boundary and target buildings were combined to serve as the input for the 3DCNN (Han, 2020). ARINet assumes the world to be a $51 \times 51 \times 51$ grid in which both the target and boundary buildings exist. Based on this assumption, superimposed binary output matrices for every building in the world were produced as part of the data processing stage. As a result, the performance of ARINet is generally good, yielding errors at 0.046; however, most errors are near the edges and in high intensity areas.

1) 3DCNN model architecture

This section describes the ARINet structure used to predict radiation intensity on a building's façade. The network output was the radiation intensity (i.e., a numerical value), thus, preferring the mean squared error (MSE) as the loss function. However, the simulation to generate the dataset can determine only the radiation intensity received by the building's surface. The loss function was then modified to account only for the MSE on the building's surface. Simulation datasets and physics-based loss functions were used to increase the accuracy of the ANN model and open to the possibility to gain more data in later stages of the model training for a more sustainable workflow.

When building ARINet, VoxNet, a 3DCNN for real-time object recognition (Maturana and Scherer, 2015), was referenced for the baseline architecture. Voxnet is a 3DCNN that can be

applied to create fast and accurate object class detectors for 3D point cloud data. Due to the simplicity of VoxNet’s architecture, a model for radiation simulation from the literature were created with the generated the dataset with voxel points (0s and 1s in space). As shown in Figure 4.3, ARINet takes a convolution and deconvolution-based architecture to preserve the original dimensionality of the 3D building objects for regression analysis outcome mapped on the voxelated matrix as inputs.

Using ARINet, the latent variables containing the hidden information from the input were obtained. By having latent variables as a part of the training models, it was possible to use the auto-encoder architecture to map the latent variables back to the 3D space in which the radiation results resided. To increase the range of captured information (i.e., handle the shadow issue), more layers were used in ARINet than in VoxNet.

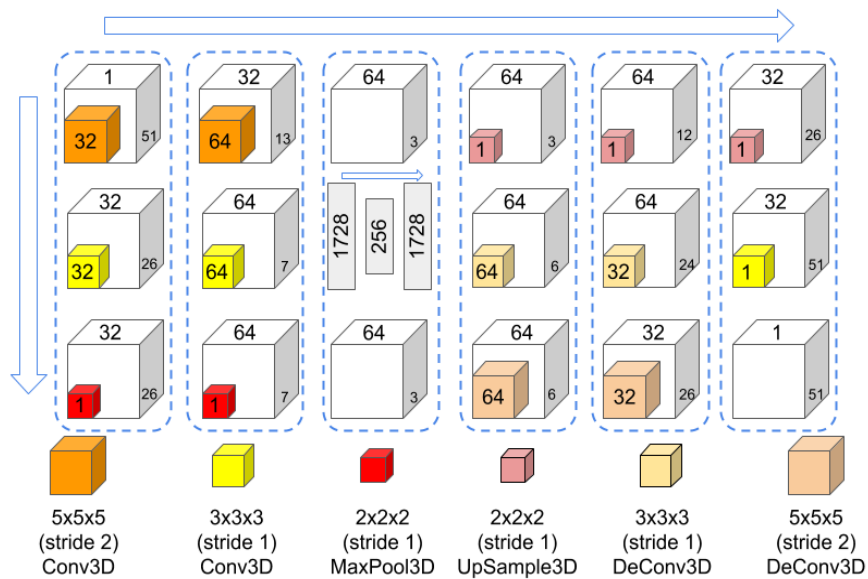


Figure 4.3: 3DCNN architecture of ARINet

ARINet assumed the world to be a 51 x 51 x 51 grid in which both the target and boundary buildings existed. Based on this assumption, superimposed binary output matrices for all of the buildings

in the world were produced as part of the data processing stage. Figure 4.3 illustrates the model architecture of the 3DCNN, beginning with the 51 x 51 x 51 voxel grid. The proposed ARINet consisted of two 3D convolutional layers before the max_pooling layer. The architecture retrieved the network by passing two additional 3D deconvolution layers. Basically, 3D image models were mapped onto latent spaces and later reshaped by calculating the difference values for radiation after passing into loss functions in the proposed 3DCNN architecture.

2) Results and Discussion

This section describes the results of the radiation received by the building façades in the new test sets. The selection of the three alternative buildings was to demonstrate the results and serve as the subject of a detailed analysis. Figure 4.4 illustrates three types of building geometry: rectangular with a horizontal overhang, round, and cube-shaped with an internal empty core.

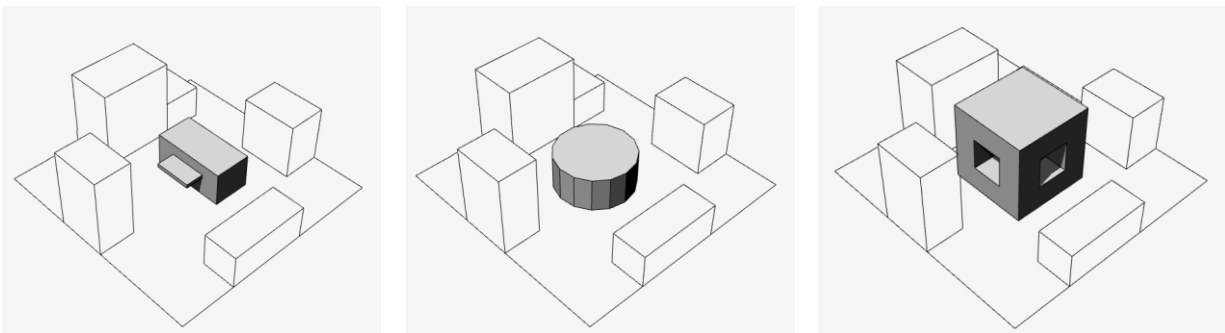


Figure 4.4: Reference geometries for the completely new buildings with boundaries

After providing these three options for the test sets, additional options with no boundary buildings were added. Total of six options to predict the radiation received were set. As ARINet training used building geometries of a specific type (i.e., box-shaped) with boundary buildings, the test sets were not expected to be predicted precisely. Figure 4.5 shows that the results for both the shading device and building offered relevant visualization output: leading to a low error rate (i.e., 0.0309) for the boundary buildings. Using the mapped radiation on the horizontal overhang,

ARINet could detect the shading device as an obstacle blocking the sunlight exposure underneath areas.

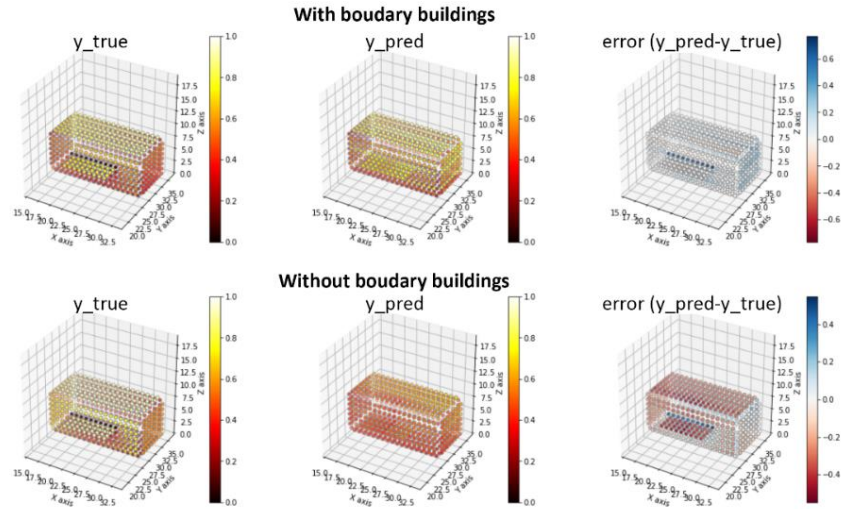


Figure 4.5: Analysis and comparison results (shading)

However, the absence of the boundary buildings resulted in a higher error rate (i.e., 0.061) and invalid estimation of the radiation intensities received by the buildings. This is mainly because the original training sets contained the information of the context buildings and never trained without the context buildings. This will be discussed in the next experiment by demonstrating the increasing accuracy of current issues by introducing more training datasets with different context building conditions.

In addition, ARINet predicted the values for the round-shaped building (Figure 4.6). However, a higher error rate was observed for the vertical façades on the round shapes. This is because there were no options for rotated façades in the training sets. Given that the model's initial training was with boundary buildings, the result with no boundary also produced a greater error (i.e., 0.0702) than did the other option (i.e., 0.0338). Additionally, by replacing L1 norm to L2 norm, the prediction accuracy could be increased.

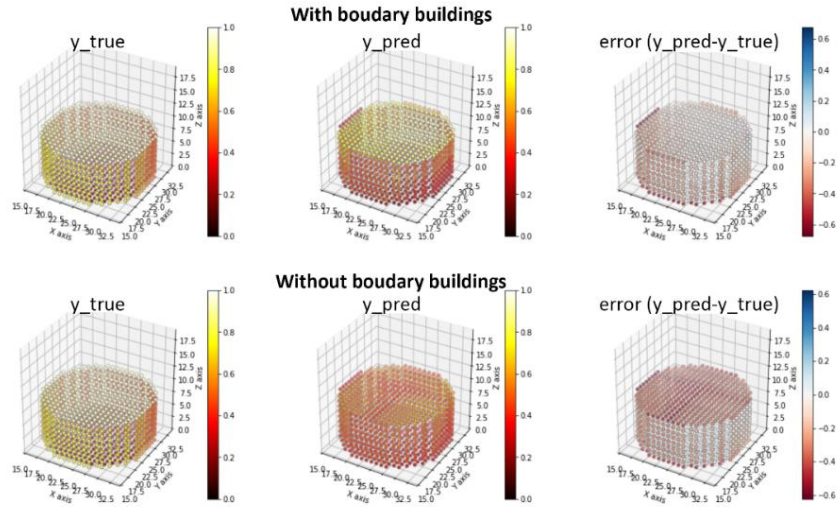


Figure 4.6: Analysis and comparison results (round)

The results did not realistically represent the radiation received by the building façades due to the information missing for the hidden properties of the surrounding buildings. However, this could be enhanced by training with different boundary buildings, which is a fixed property in the current training dataset. Increasing the rotation options for façades in the training dataset may improve the accuracy of predictions regarding vertical façades.

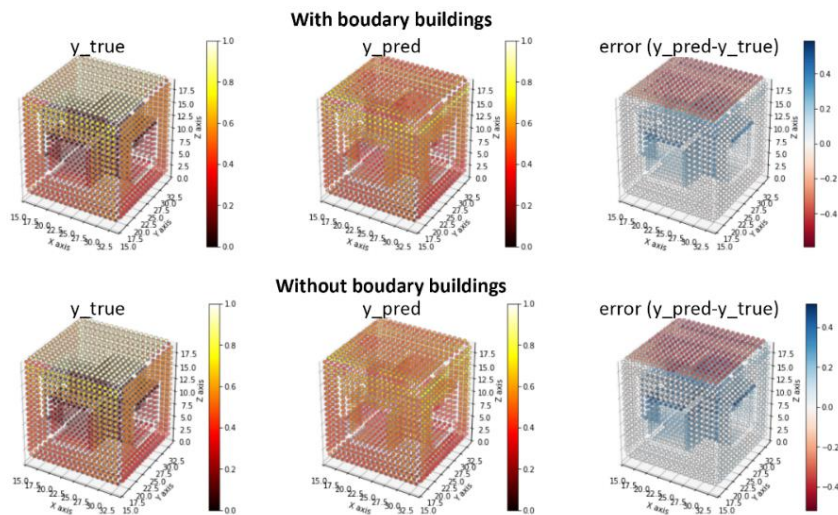
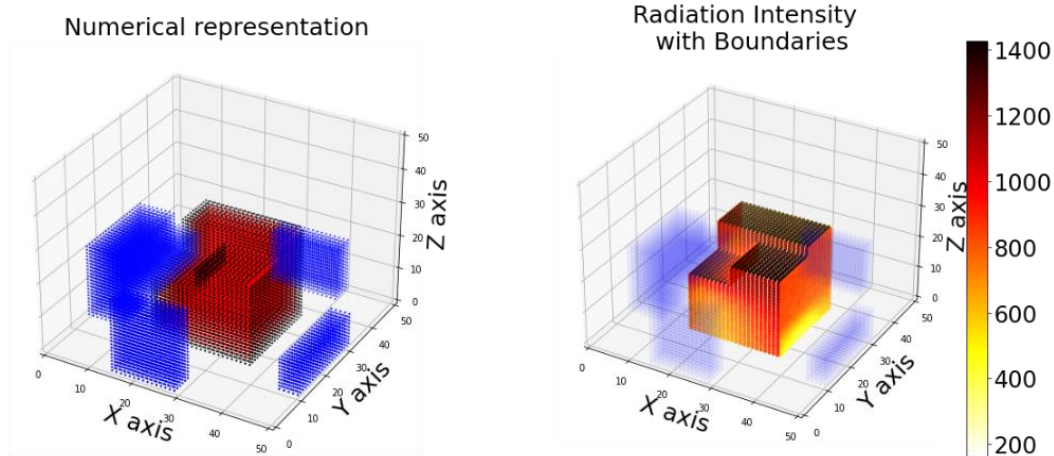


Figure 4.7: Analysis and comparison results (inner core)

The last test option was cube-shaped with an internal empty core (Figure 4.7). This option offered the lowest prediction accuracy, even with boundary buildings (i.e., 0.0538). In this case, the predictions for the horizontal rooftop and internal core surfaces differed from the expected results. This is because our model did not count the radiation bouncing off the opposite surface: thus, the model miscalculated the rays bouncing from the sun. As observed, the function of the ambient bounce in DIVA was very low, Level one or two for our model, due to the traits of the training sets. To overcome this limitation: radiation prediction for internal spaces such as floors with large windows can be used in training. In addition, the accuracy for this building without boundary buildings was better than for the round building (i.e., 0.0637). From the visualization, it can be observed that the vertical facade had a high level of accuracy since the buildings for the training set included these properties.

4.1.1.2 ANNs-solar experimental model: CoolVox

The problems of generating training datasets were discussed in the previous section. In the second solar radiation study, ensemble and fine-tuning methods were introduced to effectively predict solar radiation on building façades. Since the 3DCNNs performance was outperform than other architectures; however, there were some mispredictions on the edges, and for data without boundary conditions. To overcome the drawbacks of binary padding: CoolVox (Han, 2021) added one more categorical variable to the input matrix. The proposed method was to train another 3DCNNs model with ternary padding and preserved edge information. After pre-processing the initial data into the voxel grid (51 x 51 x 51), the model input, denoted with an X, was padded with ternary information (i.e., 0s, 1s, and 2s). In this voxel representation of 3D space, two represented the surface of a building, one the inside of a building, and zero air (Figure 4.8). The predicted values that comprised the model output, denoted with a Y, were surface radiation values in the voxel grid.



X: 3D matrix [0,1,2] - 0:Air , 1:Buildings' indoor, 2: Buildings' Surface

Y: 3D matrix - surface radiation values for every coordinate

Figure 4.8: Ternary padded voxel matrix (CoolVox)

The objectives of the development of CoolVox (Han, 2021) were to develop advanced 3DCNNs that accurately predict radiation intensities on façades; and explore a better way of encoding building geometry as ANN inputs representing the physical properties of buildings such as radiation intensities.

1) 3DCNN Model Architecture

This section describes the structures of the CoolVox1, CoolVox2, and CoolVox ensemble models used to predict the radiation intensities of the building façades. CoolVox1 consists of two 3D convolutional layers before the batch normalization and max_pooling layers (Figure 4.9). A 3D convolution layer convolves the scalable properties of different building geometries for a sparse matrix. A 3D convolution filter with learning parameters extracts low-dimensional features from the given matrix. After passing through another set of two 3D convolutional layers, the batch normalization, and max_pooling layers, the model retrieves the network by passing two additional 3D up-sampling layers. Stochastic gradient descent (SGD) was used as an optimizer (lr

= 0.01, momentum = 0.0), and CoolVox1 was trained for 100 epochs, with 32 batches in each epoch. CoolVox2 is proposed, which adds the dropout layer after the second convolution layer from CoolVox1 and applying a learning rate of 0.05 for the SGD optimizer.

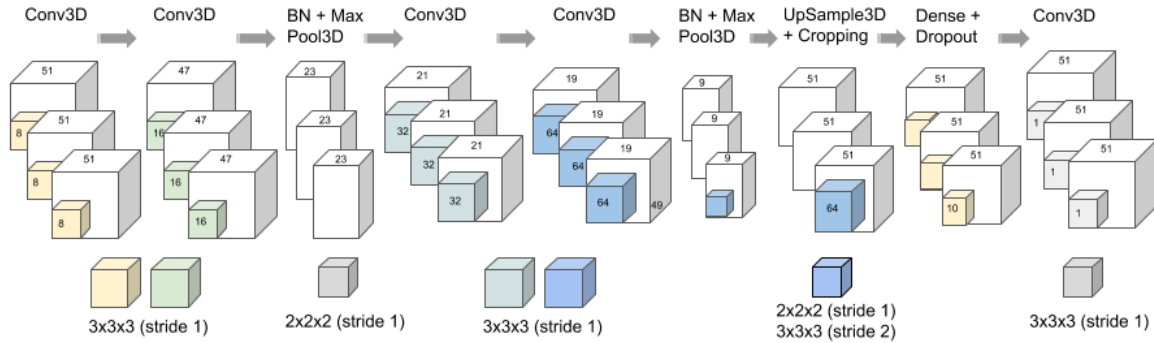


Figure 4.9: 3DCNN architecture for CoolVox1

Finally, the CoolVox ensemble model, which combines CoolVox1 and CoolVox2, was built by averaging the weights and output. An ensemble method was used because it usually results in better performance of models and has immense potential for associating different models. Both the training and validation errors of the ensemble model were reduced by 20 percent in comparison to the CoolVox2 model.

To calculate the error, the mean squared error (MSE) was selected as the loss function. In addition, the models are only concerned with the radiation intensity (kWh/m^2) on the building surface; thus, the loss function was modified to only account for the MSE on the surface and excludes those on the inside and outside of the building. The errors at the building's surface and boundary conditions were only computed and all the air space in the voxelated structure were ignored. Furthermore, all negative values were rejected and set the maximum value for radiation exposure. The customized loss function for training the models is in Table 4.1.

Table 4.1: Customized radiation loss computation

Python Code	<pre>def RadiationLoss(y_true, y_pred): <u>Inputs:</u> - <u>y_true: radiation of the target building. 3D Tensor with radiation value at target surface and others o.</u> - <u>y_pred: the prediction of the radiation.</u> <u>Returns:</u> - <u>scalar MSE loss, only calculated where radiation value not equal to o.</u> y_loc = K.cast(K.not_equal(y_true,K.constant(0)), 'float') return K.sum(K.pow(y_true-y_pred*y_loc,2))/K.sum(y_loc)</pre>
Error Metric	$\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$ <p>where Y_i is the estimated value and \hat{Y}_i is the simulated value, and n is the total number of data.</p>

2) Results and Discussion

Table 4.2 can be explained with the significance analysis. First, CoolVox1, CoolVox2, and CoolVox Ensemble models significantly outperformed the existing ARINet model, yielding validation errors of 0.0189, 0.0195, and 0.0157, respectively. Second, combining CoolVox1 and CoolVox2 by averaging the weights reduced the training and validation errors. Consequently, given that the CoolVox1 and CoolVox2 models had considerably smaller errors than ARINet (4.2), CoolVox2 further reduced the errors by combining the two; thus, the CoolVox ensemble model outperformed the others on the validation datasets.

Table 4.2: Performances of four 3DCNN models

	ARINet	CoolVox1	CoolVox2	CoolVox Ensemble
Training Error	0.0463	0.0180	0.0185	0.0146
Validation Error	0.0449	0.0189	0.0195	0.0157

Figure 4.10 depicts the prediction results by CoolVox1 and delineates the error plots on the same scale (from 0 to 1,600 kWh/m²) for all models. According to the error plot, although the CoolVox and ensemble models predicted the overall radiation accurately, they performed particularly well in predicting the rooftop conditions compared to ARINet. Furthermore, the error increased when the radiation on the north- and east-facing walls was predicted over the south- and east-facing ones. The models under-predicted the values near the edges of the north-facing façade owing to the limited sunlight on the north-facing façade. However, the CoolVox models still outperformed ARINet in predicting the annual radiation map on the north-facing façades.

South-East view of the validation building

North-East view of the validation building

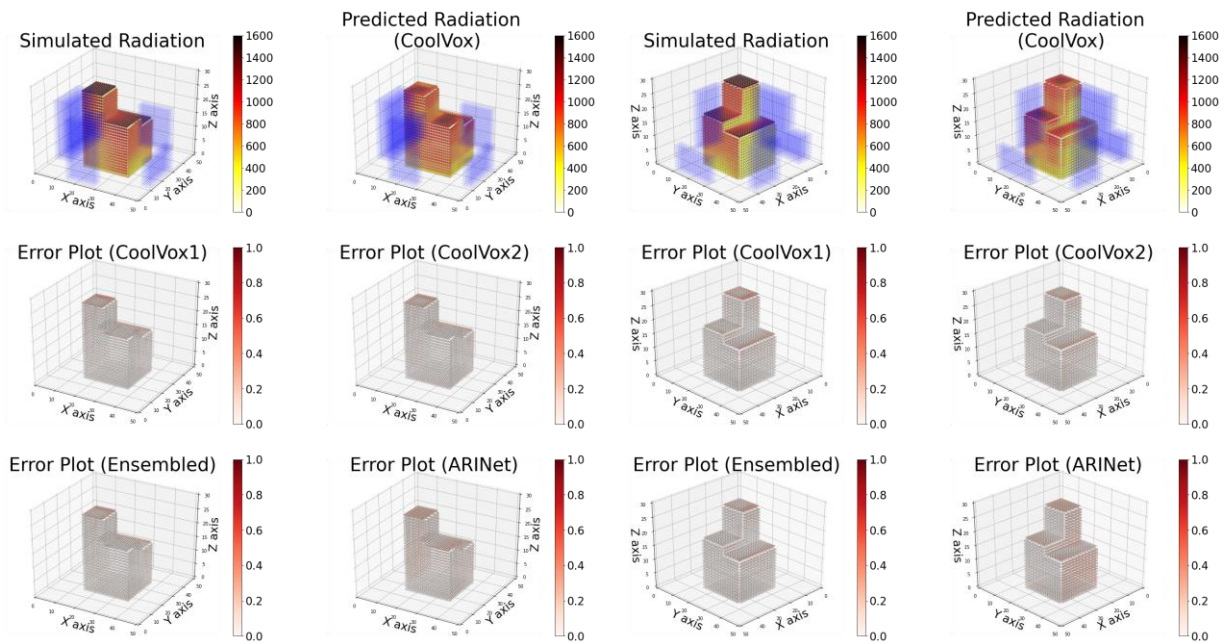


Figure 4.10: Sample results of actual values (top left), predictions (top right), and error plots of all models (middle left: CoolVox1; middle right: CoolVox2; bottom left: ensemble; bottom right: ARINet) respectively for the South-East and North-East views

The same building configurations by using the weights from the CoolVox ensemble model were illustrated in Figure 4.10. The simulated areas of the rooftop had the highest radiation intensity values for the three buildings, with values close to 1,600 kWh/m². Thus, the CoolVox model successfully captured the areas with the highest radiation intensities.

As four boundary buildings surrounded the target building (Figure 4.11, top left), the CoolVox model considered their effects when predicting the radiation intensities on the target building. The rightmost boundary from the southeast direction is a small building blocking radiation to the lower part but not on the elevated part of the target building. Therefore, the target building had relatively low radiation values at the bottom but high values on the upper-right side of the building. The top-right plot demonstrates this characteristic by the radiation predicted at the southeast façades.

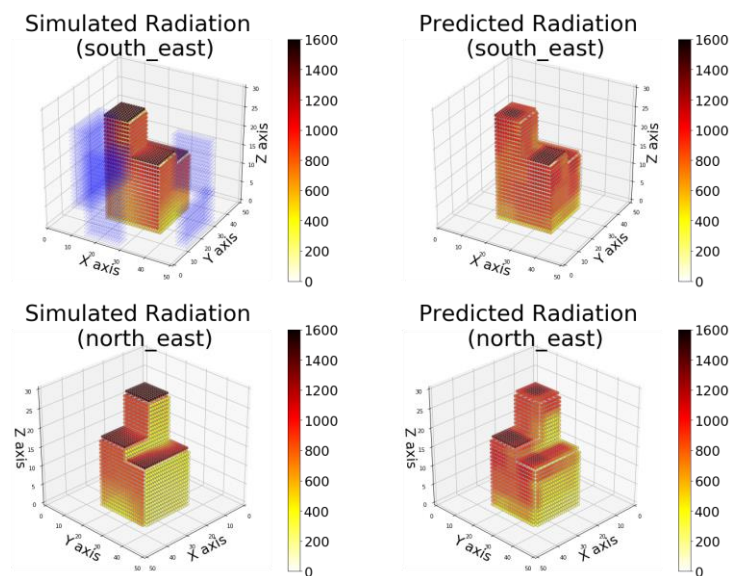


Figure 4.11: Simulated vs. predicted radiation intensities with boundaries (CoolVox)

The CoolVox model accurately predicted the areas with relatively low radiation intensities. The bottom-left image of the simulated radiation in Figure 4.11 demonstrates the low radiation intensities in the areas facing the north direction (x -axis) on the target building. In particular, the

region facing east (y -axis) at the bottom of the building had low radiation intensities. A few errors in predicted radiation intensities appear in the area facing the north (x -axis) as it is not entirely yellow. However, except in the areas near the edges, the model accurately predicted the relatively low solar radiation exposure. The target buildings were initially subject to boundary conditions and then exposed to the no-boundary condition to train the model on the different boundary conditions. The MSE for different boundary buildings were larger than in the buildings with no boundaries (Table 4.3).

Table 4.3: MSE by 3DCNN model and boundary condition

	ARINet	CoolVox1	CoolVox2	CoolVox Ensemble
With boundaries	0.054	0.0178	0.0183	0.0145
No boundaries	0.017	0.0079	0.0085	0.0066

Figure 4.12 and Figure 4.13 compare the results of the simulated and predicted radiation values from the training data, with and without boundaries. The simulated radiation is the radiation intensity values obtained from the original data, and the predicted radiation is the radiation intensity predicted by the CoolVox ensemble model. Figure 4.12 suggests that the CoolVox model trained well on the given dataset. The model successfully predicted the radiation intensities of the target building and captured the impact on the target building from all directions (north, south, east, and west). Because the data augmentation related to matrix rotation and translation was not implemented, the original directions of all datasets were maintained as global constraints.

The roof area of the target building in the simulation had the highest amount of radiation because no surrounding building obstructed it when absorbing solar radiation. The target building in the prediction (top- and bottom-right images in Figure 4.12) exhibited this trend; it also revealed the roof areas as experiencing the highest amount of radiation. In the top-left image, the right side of

the building facing the y -axis receives a relatively high radiation intensity on the top but a low radiation intensity at the bottom. The CoolVox model accurately predicted high radiations on the top and low radiation on the bottom, as seen in the region facing the y -axis in the top-right image.

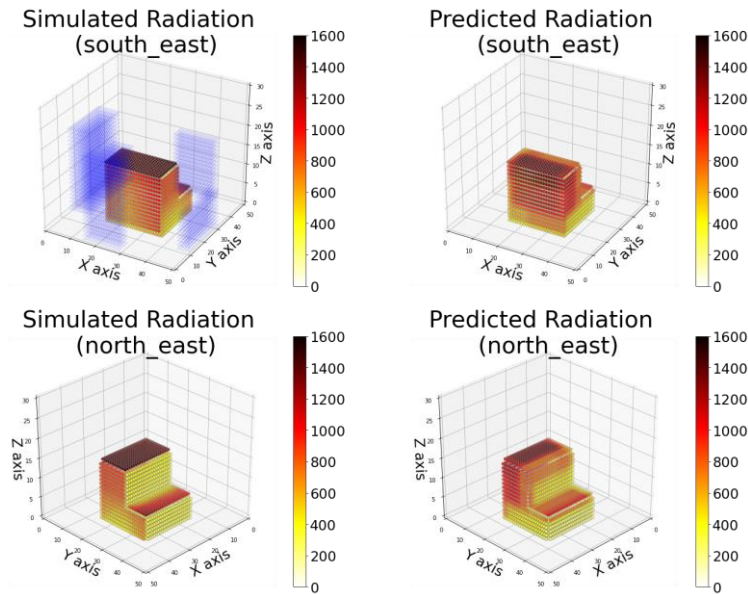


Figure 4.12: Simulated vs. predicted radiation intensities with boundaries (CoolVox)

The bottom-left image in Figure 4.12 shows that the two roof areas on the target building have high radiation intensity values but low values on the adjacent section facing the north (x -axis). The roof areas absorb heat from the Sun regardless of the presence of the boundary building. In contrast, the roof absorbs the least amount of heat at the north-facing façades. The target building (bottom-right image) received the highest radiation intensity in the roof area but yielded a few errors toward the x -axis where it was expected to absorb the least amount of heat, as its color was not entirely yellow.

When there are no boundary buildings (Figure 4.13), the target building has high radiation intensity values on different surfaces, except where the façade faces away from the sunrays. In the top-left image, the simulated radiation intensity is the highest on the roof, second-highest on the

area facing the south (x -axis), and relatively high on the area facing the east (y -axis). The area facing the x -axis also absorbed heat throughout the day as it faced the Sun. The area facing the y -axis did not adequately absorb heat when the Sun was in the west but absorbed a fair amount of it when the Sun was in the east. In the bottom-left image, the north-facing façade of the building did not absorb heat from the Sun and the other sections, as seen from the yellow area facing the x - and y -axes.

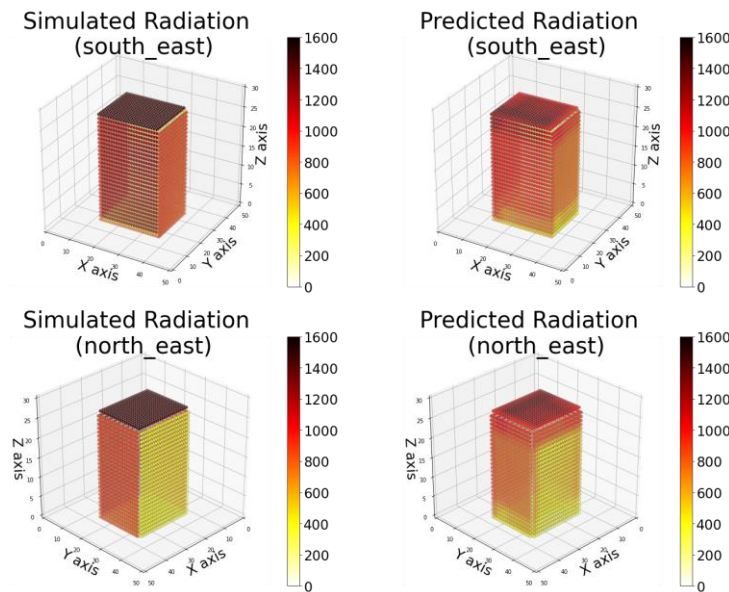


Figure 4.13: Simulated vs. predicted radiation intensities without boundaries (CoolVox)

For the no-boundary building scenario, the CoolVox model satisfactorily predicted the radiation intensities for the building, except in a few areas. For example, in the bottom-right image, the model makes errors when predicting the area not facing the Sun. However, in general, it performed better for buildings without boundaries than those with boundaries: as it was more challenging to train the model when there were different types of boundary buildings around the target building.

Finally, the circular geometry with boundary conditions was tested. Because the distance between the target and the boundary building was relatively long, the simulated values for building façades did not vary substantially. However, the trend of low radiation mapped on the north façade remained the same, as seen in the bottom two images in Figure 4.14.

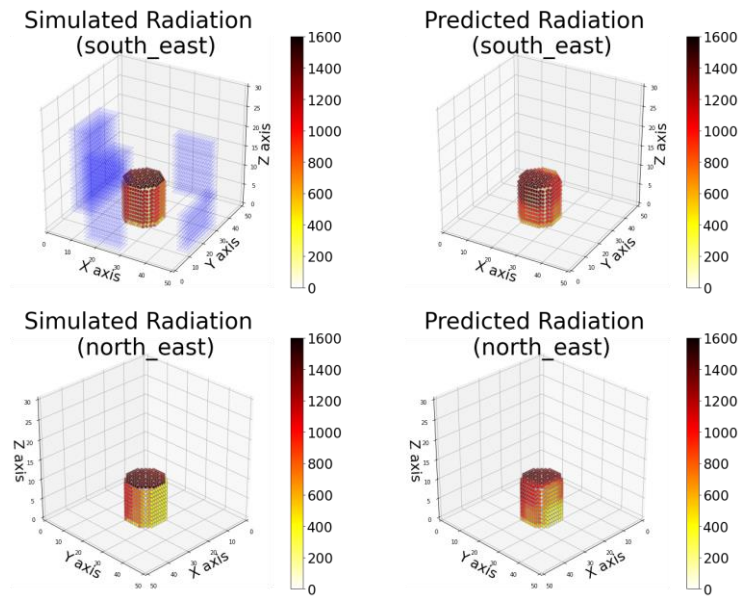


Figure 4.14: Simulated vs. predicted radiation maps for validation sets with boundaries

Unlike in the cubic geometry, the model prediction was better in the curved part of the building’s edges. Overall, the general trends in radiation intensities on the building façades were accurately captured in the case of the circular building.

3) Testing with new datasets

This section describes the amount of radiation received by the building façades in the new geometric configurations. Three new datasets were selected to demonstrate the results and serve as the subject of subsequent discussions. Figure 4.15 illustrates three types of building geometry: multiple small buildings, one canopy-style building, and one tall building with self-shading.

- Scenario 1: Increase in the number of adjacent buildings
- Scenario 2: Introduction of underexposed areas
- Scenario 3: Extension of building height

To avoid overfitting and overestimating the feasibility of the trained model, the new datasets for testing were created and tested. The testing process used different geometries and boundary conditions that did not exist in the training and validation datasets.

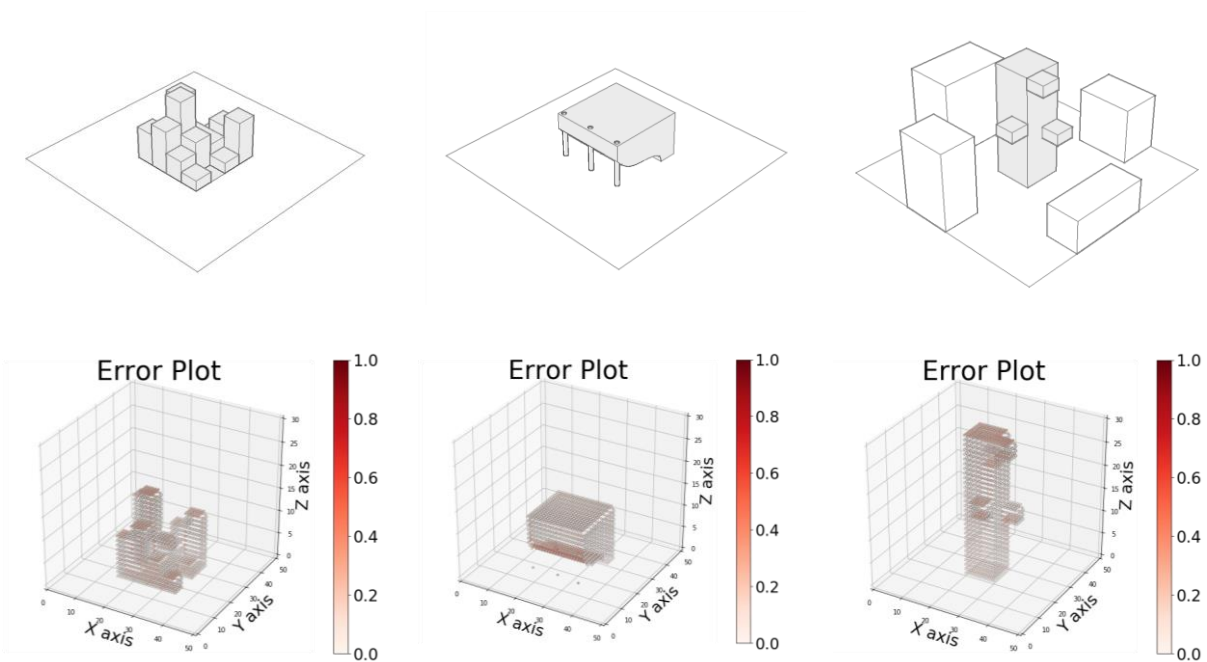


Figure 4.15: Error plots of multiple buildings (left), canopy-style buildings (middle), and tall buildings with self-shading (right)

For the three different datasets, CoolVox had an average error of 0.018, similar to the validation error in Table 4.3. In the case of multiple buildings with 16 different heights (left image in Figure 4.15), the prediction was accurate except for a few roof areas. The model performed well except in the underexposed area for the canopy-style building (middle image in Figure 4.15). The prediction

was also accurate for the self-shading building (right image in Figure 4.15), except for a few roof areas.

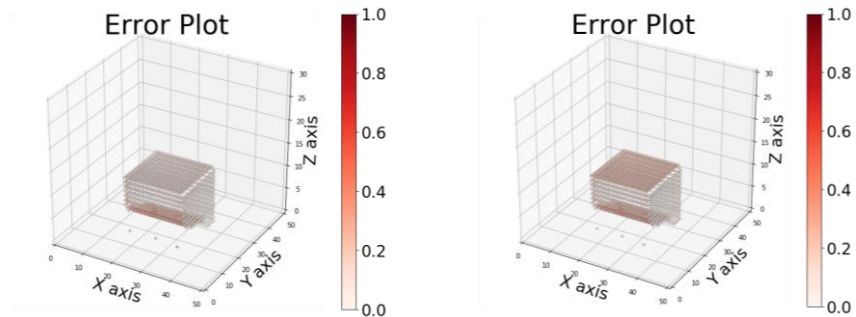


Figure 4.16: Error plots on the test set: CoolVox (left) and ARINet (right)

In the error plot of CoolVox in Figure 4.16, the error is relatively low for different surfaces compared to ARINet. CoolVox exhibited a relatively high error in the roof area (indicated by its color: which is darker than the other surfaces in the error plot), albeit less than that with ARINet.

This shows that the potential ternary methods outperformed when training the 3DCNN. The findings from the solar radiation studies are as follows:

- 3DCNNs can learn the orientation and boundary conditions of buildings.
- Ensemble and fine-tuning methods are promising means of advancing the models.
- 3D geometric data must contain edge information for 3DCNNs.
- Ternary padding outperforms binary padding.

These findings can be directly applied to the comprehensive framework as data pre-processing for radiation simulation with ANNs. The efficient data representation techniques for different ANN model transfer were developed in the dissertation. Which are the data post-processing techniques for connectivity to BIM software. All the presented work illustrates the data conversion technique for ANN-based BPS workflow.

4.1.2 ANNs-driven Airflow Simulation

4.1.2.1 ANNs experimental model: AirVox

Estimated flowrate and patterns have been used as buildings' performance indicators to predict the ventilation rate, air distribution, and mass transfer in buildings (Hopfe, 2012). Different computational models have been used to estimate airflow indoors, such as empirical formula, airflow network, and computational fluid dynamics (CFD) (Tan and Glicksman 2005; Zhai, El Mankibi, and Zoubir 2015). The numerical values calculated from the empirical or semi-empirical formulae are often too simplified compared with other simulation methods such as CFD (Blocken et al., 2011). The benefits of using CFD in early design consist of three parts: outdoor environment application, indoor environment application, and the volumetric air distributions and their visualization.

In the past few decades, CFD has been extensively used to represent the outdoor condition around buildings and the airflow distribution indoors (Gough et al., 2020; Li, Delsante, and Symons 2000; R. Widiastuti, M. I. Hasan, C. N. Bramiana 2020; Tong, Chen, Malkawi, Liu, and Freeman 2016). The early design practice has increasingly involved CFD for building designs, recently to optimize buildings' geometry and analyze local wind environments (Lee and Song 2014; Wang and Malkawi 2019). However, the computational time and efficiency have not been resolved quite well for applying CFD in instant design changes (Hensen, Djunaedy, Radošević, and Yahiaoui 2004).

DL has recently gained popularity for achieving state-of-the-art performance in different tasks, including text, image, and sound (Ioannidou, Chatzilari, Nikolopoulos, and Kompatsiaris 2017). Due to its wide applicability, such as 3D data processing and modeling, the efforts for solving different vision tasks are necessary (Gezawa, Zhang, Wang, and Yunqi 2020; Ioannidou et al.,

2017). DL can learn specific tasks on 3D data, such as image segmentation, recognition, and motion translation. Applying 3D vision tasks in fluid dynamic simulation has gained popularity with the advanced 3D vision techniques (Bouchiba et al., 2018; Fang, Sondak, Protopapas, and Succi 2018; Mohan and Gaitonde 2018). However, the direct application of ANNs-based CFD models in building design has not been explored yet. This study demonstrates an efficient way to model ANNs architecture for the airflow predictions and the potential application of such techniques in the architectural design practice.

This experiment uses the simulated data generated from the conventional CFD (i.e., OpenFOAM), trained with the model architectures discussed in the previous section. Figure 4.17 illustrates the overall workflow consisting of three parts: pre-processing of the simulated data and building geometry, ANNs modeling and prediction, and post-processing of the outputs in CAD software.

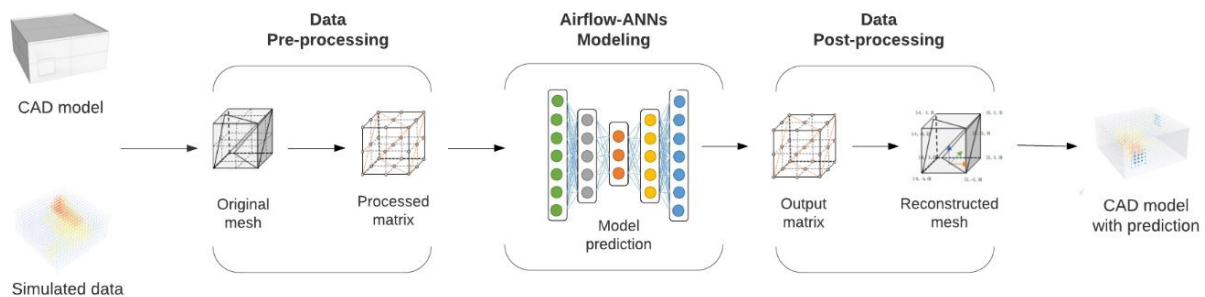


Figure 4.17: The workflow of modeling ANNs-based CFD

1) Data processing with simulated airflow results

The OpenFOAM-based CFD plug-ins for Grasshopper and Rhinoceros were adapted to generate training datasets. A total of 3,750 simulated data were collected with a fixed geometry and position variations of inlet and outlet on facades. The wind speed was set to 1, 3, and 5 m/s, and the wind direction varied from 30 to 90 degrees by 15 degrees.

The voxelated matrix represented each of the 2,750 datasets in the 3D grid of (21, 21, 11), as shown in Figure 4.18. The two images on the left-hand side of Figure 4.18 are a populated grid on the façade with inlet/outlet information, padded with ternary information (i.e., 0s, 1s, and 2s). Similar to categorical embedding for the radiation, the model input information for different design attributes can be translated into the categorical values for the simulation purpose. In this case, the interior space, inlet, and outlet are separately encoded into three values for training networks. The image on the right side of Figure 4.18 is a voxel representation of the output grids with the numerical information of the 3D wind vector and relevant wind pressure on simulated datasets. The input matrix of the 3DCNNs is defined by combining input and output points.

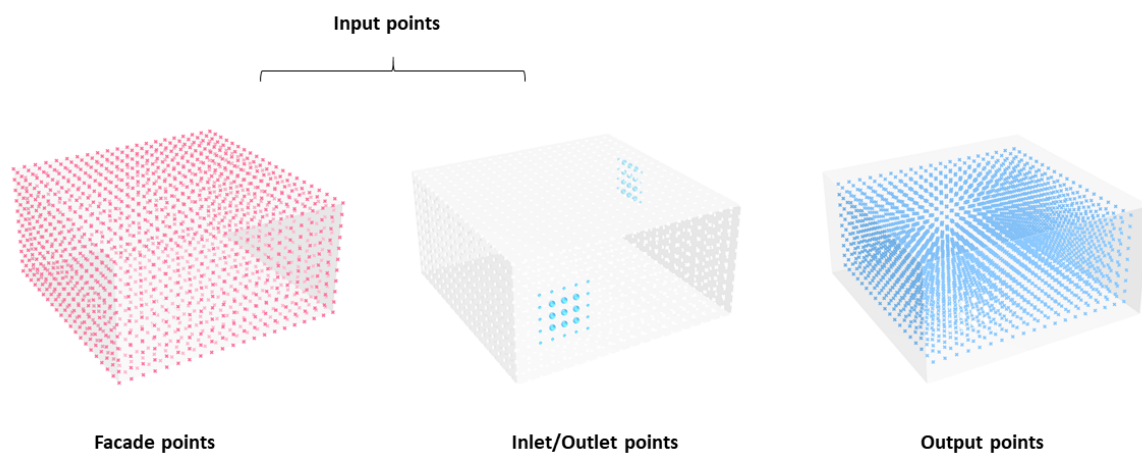


Figure 4.18: Input and output grids as inputs for 3DCNNs

To train the network, the output grid was used to calculate the relative position between inlet and output points and outlet and output points. Therefore, the relative positions of the inlet and outlet as cartesian coordinates of x, y, and z were embedded in the input matrix. Additionally, the coding of wind speed and direction were in the categorical values between zero and one, e.g., the coding for a wind direction of 90 degrees was one, and that of 45 degrees was 0.5.

2) 3DCNN model architecture

This section describes the architecture of the AirVox models used to predict the wind speed and pressure inside the given geometry. Because the validation of the high performance of Conv-Deconv architecture in BPS for radiation was in the CoolVox study, different ANN architectures currently used in the computer vision were added with the traditional 3DCNNs. Figure 4.19 illustrates four different 3DCNNs used for training voxelated datasets.

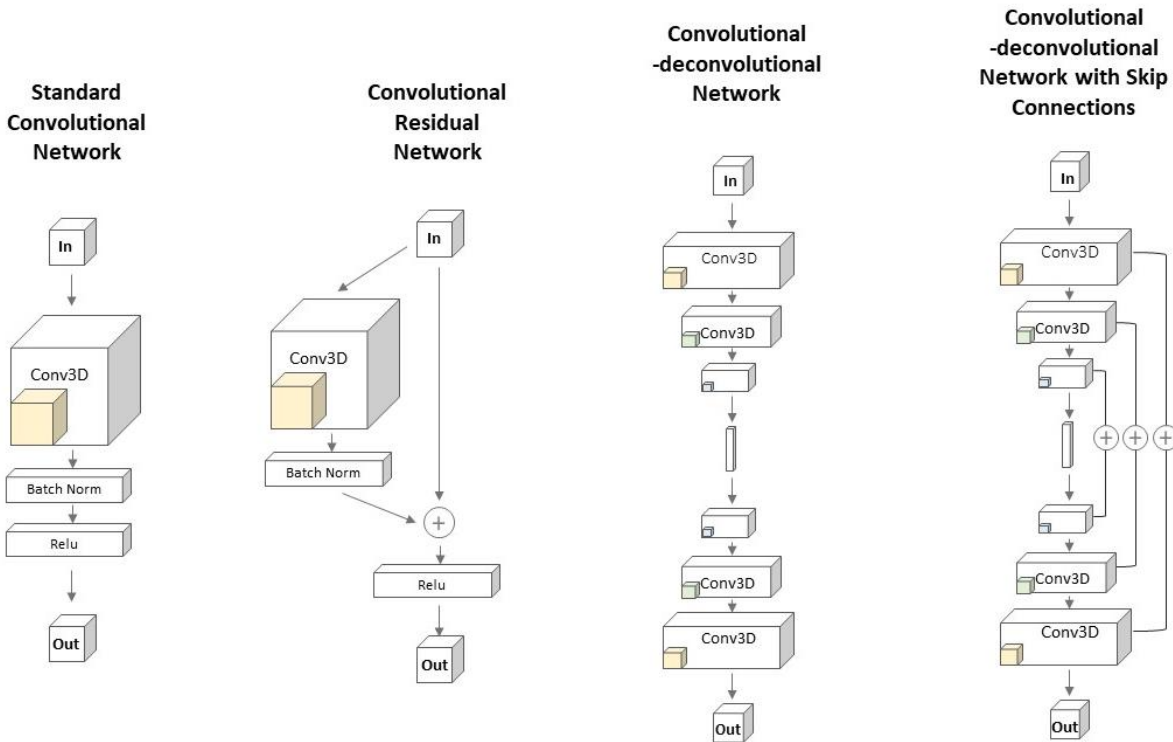


Figure 4.19: Different 3DCNNs architectures for training airflow network

After developing the standard CNNs, convolutional residual networks, and convolutional-deconvolution networks, the same input representation of a volumetric shape was adopted: a voxel matrix of resolution 32^3 . Therefore, each point in section 1 is first converted to a voxel representation with a regularly populated 3D matrix.

Standard convolutional neural network (S-Conv)

Standard convolutional neural network (S-Conv) is a well-validated approach for efficient training of multiple layers for various applications. The S-Conv architecture shown in Figure 4.19 on the left comprises 3D convolutional layers, batch normalization, activation function (i.e., ReLU) layers, and fully connected dense layers. The CNNs use filters to convolve the entire voxel preserving the 3D properties of inputs and generating different feature maps. Owing to the presence of local connectivity, CNNs preserve the correlation between neighboring pixels with the fixed object's location. The $3 \times 3 \times 3$ filters were used and the same number of filters for the same output-feature map size. Next, the batch normalization layers accelerate the training process by standardizing the inputs to a layer. Lastly, the fully connected layers provide meaningful outputs while maintaining the non-linear combinations of features from the input space.

Convolutional residual neural network (Res-Conv)

Convolutional residual neural network (Res-Conv) is formulated based on the hypothesis of residual mapping, which easily optimizes the network by mapping identities by a stack of nonlinear layers. Based on the S-Conv architecture, shortcut connections were inserted in this architecture, enabling the networks to have residual properties. In Res-Conv architecture for AirVox, the training outputs are added to the outputs of the stacked layers, and the networks are trained ReLU with backpropagation.

Convolutional-deconvolutional network (Conv-Deconv)

The voxel-to-voxel mapping on the 3D matrix by encoding numerical combinations of position and vectorized values was created to formulate learning architecture from a whole input voxel grid to the output a voxel grid. Convolutional-deconvolutional network (Conv-Deconv) takes the

convolution and deconvolution layers as the main architecture to maintain values in the voxel while training the networks. The top half of our network is an encoder structure that results in the condensed representation of the voxel grid that connects to a fully connected layer. The bottom half of the architecture reconstructs the network, which is the same size as the matrix of inputs layers. This stage uses deconvolution layers to expand the condensed representation of the inputs to output predictions.

Convolutional-deconvolutional network (Conv-Deconv-Skip)

The accuracy of predicting performance metrics using Conv-Deconv networks has been proven in previous experiments on the radiation intensities on the buildings' facades. However, the airflow required a finer resolution of the output distribution compared to radiation. While maintaining the Conv-Deconv structure, the skip-connection layers were added to feed more information. This network has similar architecture to the U-nets (Ronneberger et al., 2015). In addition, it is carefully optimized for training voxels for AirVox. To complete Conv-Deconv-Skip networks, the pooling operators and upsampling layers were used as baseline operators like Conv-Deconv networks but added fully connected layers to combine the localized data to the up-sampled output.

3) Results and Prediction

In this section, the airflow patterns with magnitude on the voxel grids using S-Conv, Res-Conv, and Conv-Deconv were illustrated. Figure 4.20 compares the mean-squared error (MSE) of the training and validating sets using S-Conv networks. The errors are calculated based on the exclusive areas such as buildings' indoors because of the customized loss function. The results of predicting angled flow show limitation of using S-Conv architecture.

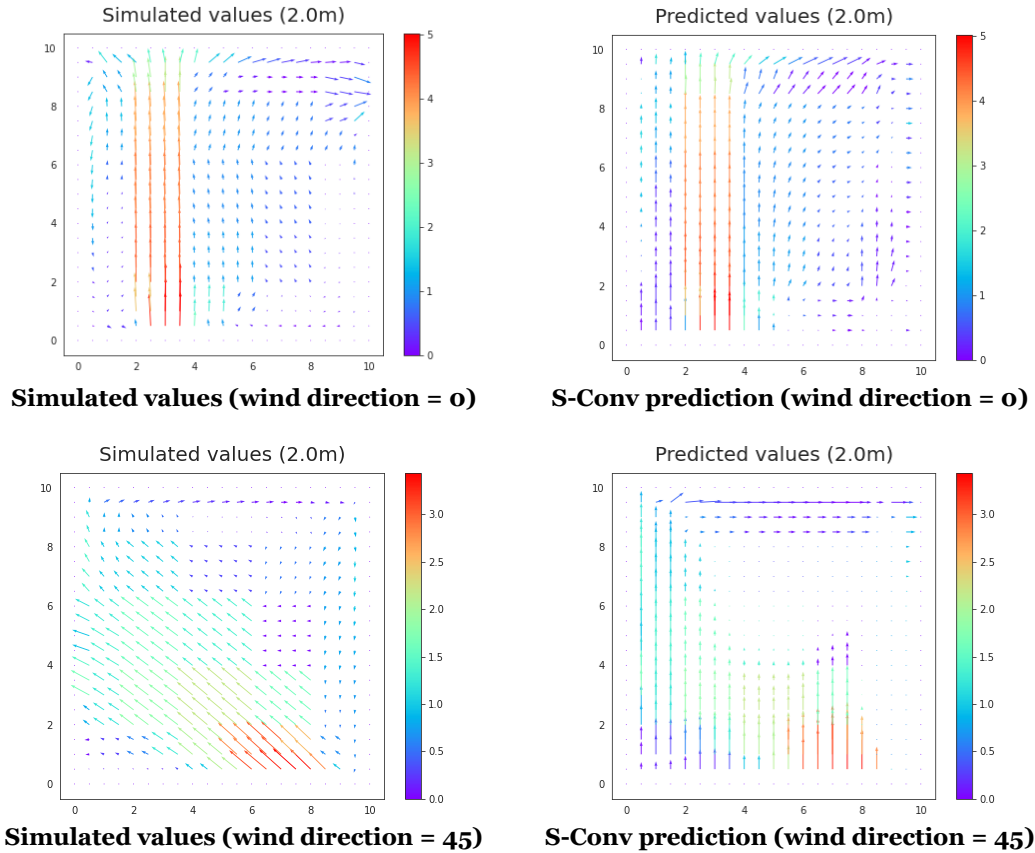


Figure 4.20: Visualized results comparison: Simulated vs CNNs predictions

As demonstrated from the plotted airflow pattern at 2.0 m height from the ground level, the predictions of different wind directions (i.e., 0 degrees and 45 degrees) and same wind speed, 3 m/s, are plotted in Figure 4.20. S-Conv model predicts the flow's magnitude well with the airflow perpendicular to the inlet (images in the first row in Figure 4.20); however, it fails to predict the directional airflow at 45 degrees from the inlet (images in the second row in Figure 4.20). This shows that the S-Conv is insufficient for predicting indoor airflow patterns with angled flow from the inlets. Therefore, the other three networks were trained to prove the feasibility of 3DCNNs for airflow simulation.

Firstly, Res-Conv was trained since it has more light architecture than Conv-Deconv and directly expanded from the original S-Conv model. Res-Conv captured the airflow patterns more

accurately than the S-Cov (images in the first row in Figure 4.21). However, the predicted values near the boundary areas show a higher level of errors than the Conv-Deconv model (images in the second row in Figure 4.21). Comparing Res-Conv and Conv-Deconv revealed that Conv-Deconv has a stronger ability to capture the general wind speed across interior space.

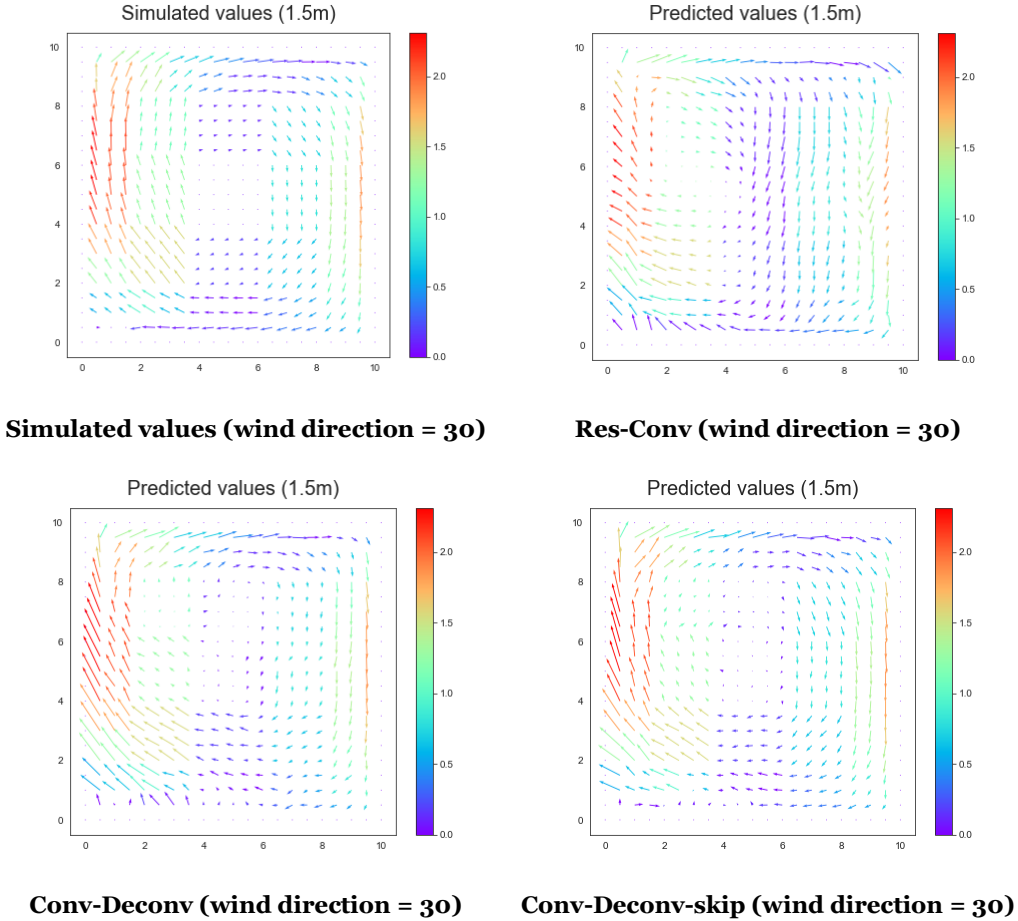


Figure 4.21: Visualized results comparison: Simulated vs. other models predictions

The Conv-Deconv model predicts the airflow magnitude most accurately, yielding the MSE errors at 0.036 for validation sets. Additionally, Conv-Deconv-Skip model was proposed to increase the accuracy of the Conv-Deconv networks. The Conv-Deconv-Skip model takes the base architecture of Conv-Deconv networks and adds additional skip connection layers to preserve more information during the training periods. This architecture was carefully designed to capture

detailed phenomena around the edges areas and opening areas when predicting airflow. Consequently, the Conv-Deconv network with skip connections outperformed the other networks for all three wind vector components in both magnitudes and directions.

Figure 4.22 shows the predicted and simulated values in the same plot with different vector components of x, y, and z for different architectures with height increments from the ground to the ceiling. The dotted line shows the original simulated data, and the solid line shows the predicted value as output from the ANNs. This figure shows more distinctive performance differences in vertical levels between different models. Conv-Deconv architecture demonstrates outstanding performance compared to the Res-Conv Networks in all wind vector components. In addition, the difference between Conv-Deconv and Conv-Deconv with skip connection is not very distinctive. Errors are generally low near the opening height, which is between 0.5 m to 2.5 m, and increase when it is close to the ceiling and with no opening areas.

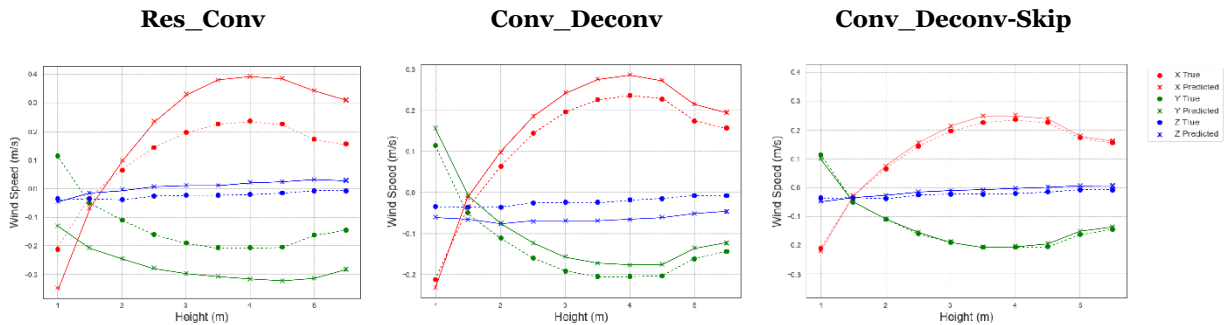


Figure 4.22: wind speed analysis for different height of the rooms among different models

Table 4.4 compares the mean absolute error (MAE) and the mean squared error (MSE) of the training and validation sets. Because of the customized loss function, the basis for calculating the errors was the exclusive areas such as the buildings' indoors. In addition, the training time increases proportionally due to more complex architecture with hidden layers. Consequently, it was found that the least MAE errors with Conv-Deconv architecture at 0.36 and the least MSE

errors with Conv-Deconv-Skip architecture at 0.0059. Conv-Deconv takes 1/3 less time for the training model than skip connection but yields similar errors.

Table 4.4: Loss comparisons of proposed CNNs.

	S-Conv	Res-Conv	Conv-Deconv	Conv-Deconv-Skip
Training time, s	65 s	74 s	195 s	256 s
MAE	0.357	0.139	0.036	0.048
MSE	0.064	0.05	0.008	0.0059

Buildings are 3D objects, thus their performance information can be formulated as a 3D matrix. Therefore, it is important to check the prediction accuracy across different levels in buildings. To discuss the prediction results further, the prediction results are plotted in Figure 4.23 from 1.5 m to 4.5 m at intervals of 1 m.

The Conv-Deconv-Skip model was used to output the results, and the MSE errors were plotted with the comparison results of the simulated and predicted values. The result plots in each plane show visually similar airflow patterns and magnitude for both simulated and predicted data across all heights. The error plot on the right shows the absolute difference of all three wind vectors from the simulated data. Among all the error plots, the lowest observed average MSE was for the 1.5-m outputs because of the relatively low wind speed across the lower plane. However, it is also well-captured at the height of the openings. Both planes with an inlet (3.5 m) and outlet (2.5 m) yielded an MSE of 0.003, which is a lower error rate than the average errors of the proposed models.

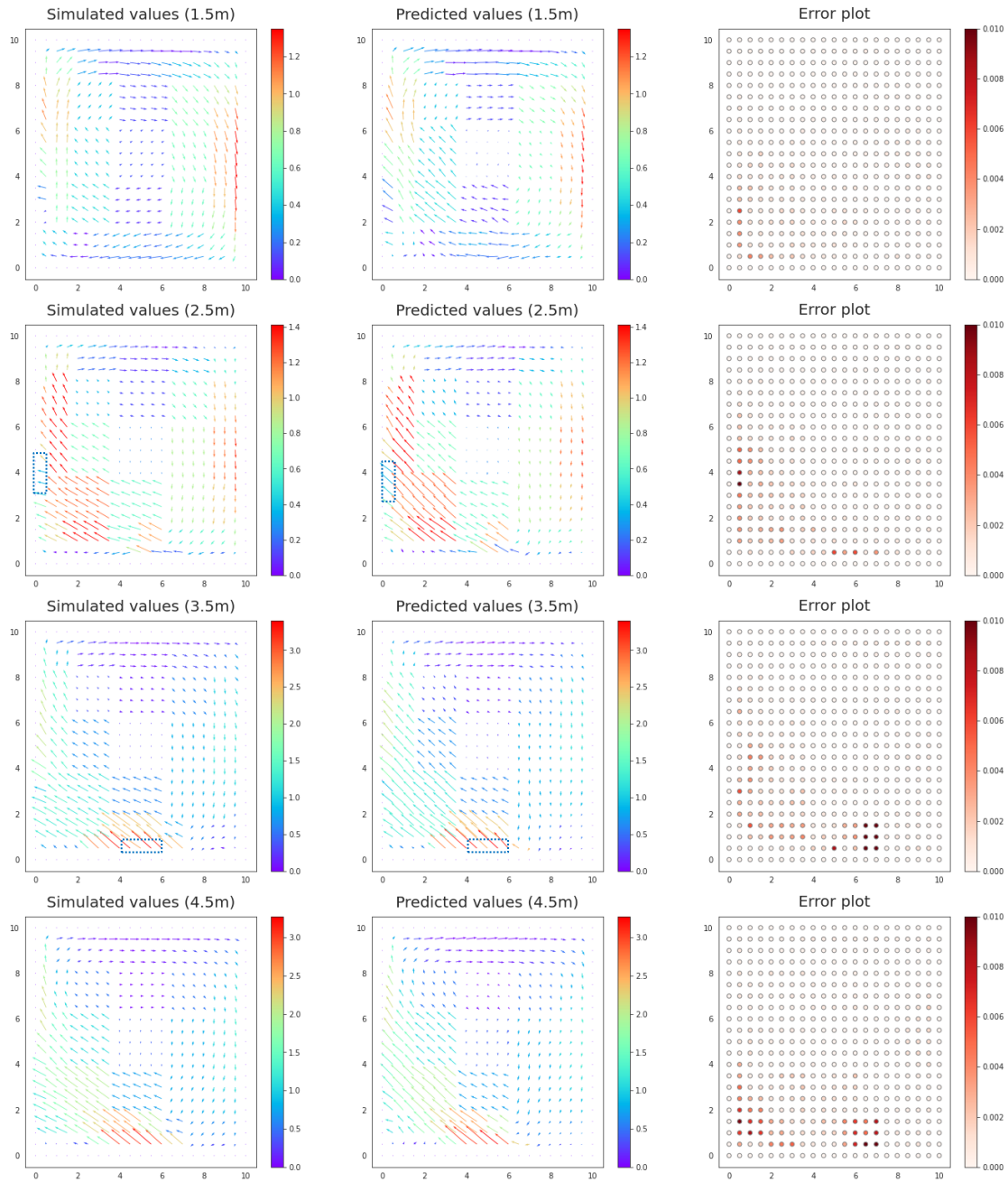


Figure 4.23: Result plots with different height levels (Conv-Deconv-Skip)

4) Discussions

Owing to the densely populated voxelated grids for the input boundaries, potential geometries, and inlet and outlet locations, the test sets to prove the generalizability in predictions. The way of computing the input matrix is identical to generating the training and validation sets. The only thing that should be of concern in modeling inputs is to make sure their boundary definition matches the ANN's boundary definition. To test the accuracy and adaptability of the networks on different geometry configurations with positions of inlet and outlet, a narrow rectangular-shaped room was introduced as a test case instead of the squared plan (Figure 4.24).

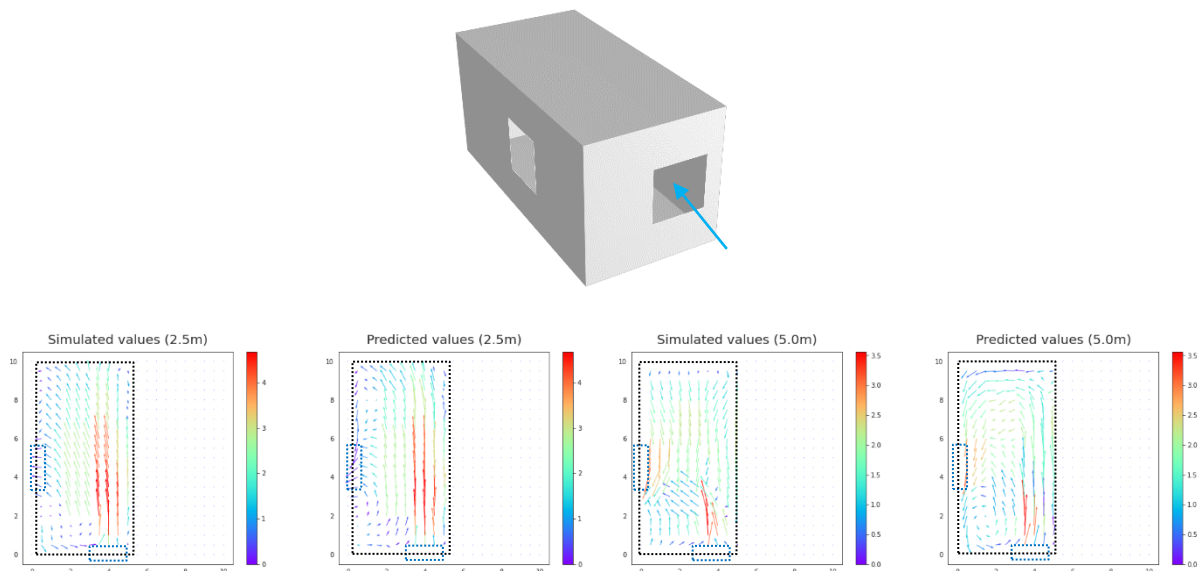


Figure 4.24: The comparison of results with simulated values in different section levels.

The bottom plots of Figure 4.24 show that the model predicts the airflow indoors with increasing height with an average MSE error of 0.07. The plots from different heights show the general indoor air velocity in reasonable amounts but fail to capture the air turbulence with errors lower than 1.3 degrees.

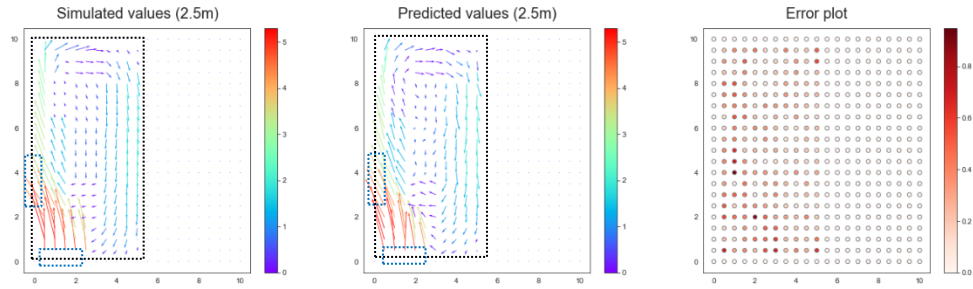


Figure 4.25: The comparison of results with simulated values and error plots.

Because the test sets include different geometry boundaries from the training sets, the location of the inlets is crucial. The experiments with different locations of inlets in voxels show different results, shown in Figure 4.26. In Figure 4.26, AirVox predicts the indoor velocity field better when the inlet location is on the boundary layers. The inlet positioned in the voxel boundary shows the MSE errors at 0.65, yet the inlet in the middle of the voxel shows errors of 3.2. When the test geometry's location was in the middle of voxels, the error increased significantly and failed to predict the airflow pattern accurately.

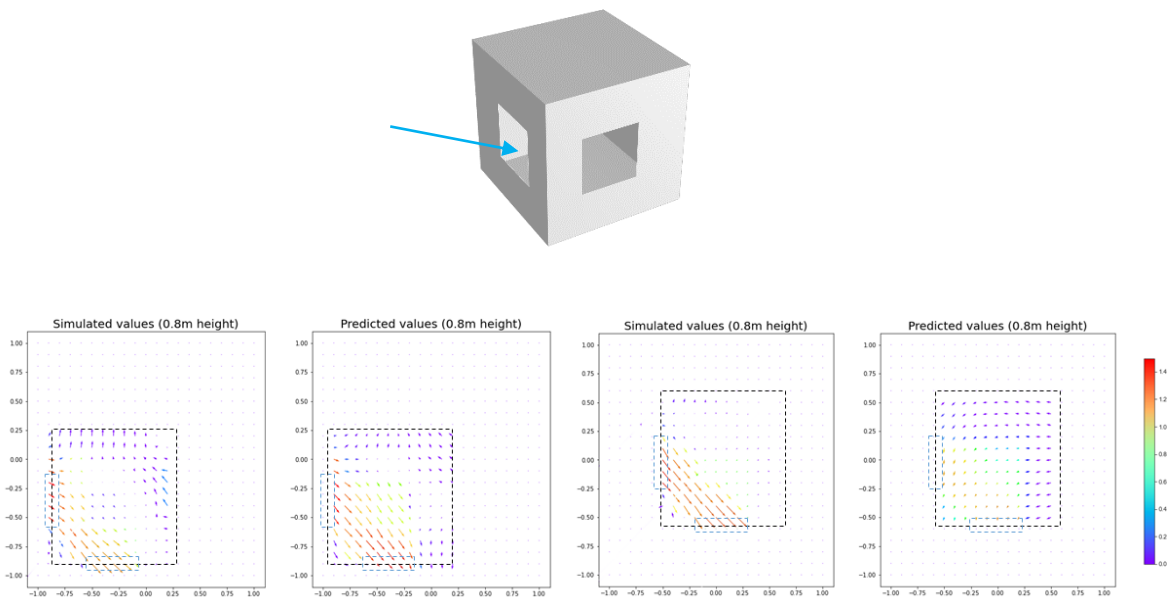


Figure 4.26: The result comparison of the different positions of building in the voxel.

Another finding from the test set experiments is that the inlet's location is not aligned façade of training sets. Since the training sets comprise the square-shaped buildings with various window locations, the AirVox fails to capture the airflow indoors when the inlet position is on the diagonal-shaped wall (Figure 4.27 on the left). However, when the inlet is on the boundary of buildings in a voxel with the same building shape, the AirVox predicts indoor airflow well, yielding errors of 0.06. For future works, to overcome the problem during this experiment, the data augmentation with the rotations of buildings geometry in given voxels is highly recommended.

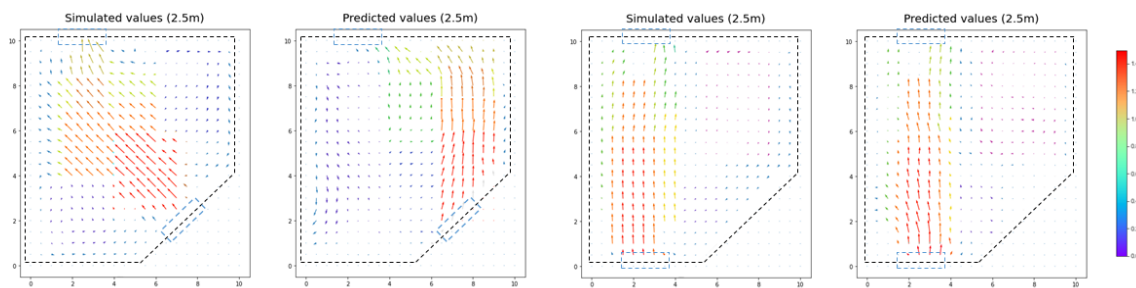


Figure 4.27: The result comparison of the different building's inlets location

The findings from the airflow simulation studies are as follows:

- 3DCNNs with convolution and deconvolution architecture predict airflow indoors well.
- Conv-Deconv model can be further improved by adding skip connection layers.
- 3D geometric data must contain the inlet and outlet information related to the voxels.
- The data augmentation with rotation works well by covering more diverse wind options with rotated geometries.

4.2 Algorithms for 3D data exchangers

The ANNs model for radiation and airflow were decided from the two main experiments in this chapter. However, implementing the ANNs model into CAD software requires the pre- and post-processing of the buildings' data and results mapped into the voxelated matrix. This section describes the algorithms implemented as a part of the workflow. Algorithms to convert geometry to voxel and map voxelated information back into the surfaces need to be developed. In computer graphic practice, there are several algorithms to convert a surface to a voxel grid, such as winding number and ray-casting. Marching cube is one of the widely validated techniques for retrieving the surface from voxels in practice. These are used for the core algorithms that convert CAD geometry to the inputs for ANNs and vice versa.

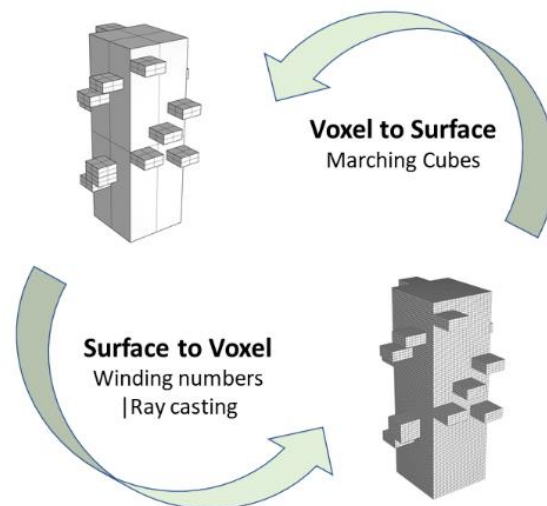


Figure 4.28: Diagram of converting both surface to voxel and voxel to surface

Figure 4.28 describes the relationship between implemented algorithms: (1) convert surface to voxel and (2) retrieve the 3D model from the voxelated matrix. The first step is designed for converting building geometry information to the input for the neural networks. The second step is to preserve the simulation properties of the original building geometry.

4.2.1 Robust voxelization

4.2.1.1 Ray-casting

Voxel-ray-cast algorithm is one representative algorithm to traverse voxel form (Figure 4.29). The projected ray detects whether the properties that penetrate are either the voxel boundary or not. Due to its simplicity and speed, many software use this method to create voxel matrix from the geometries.

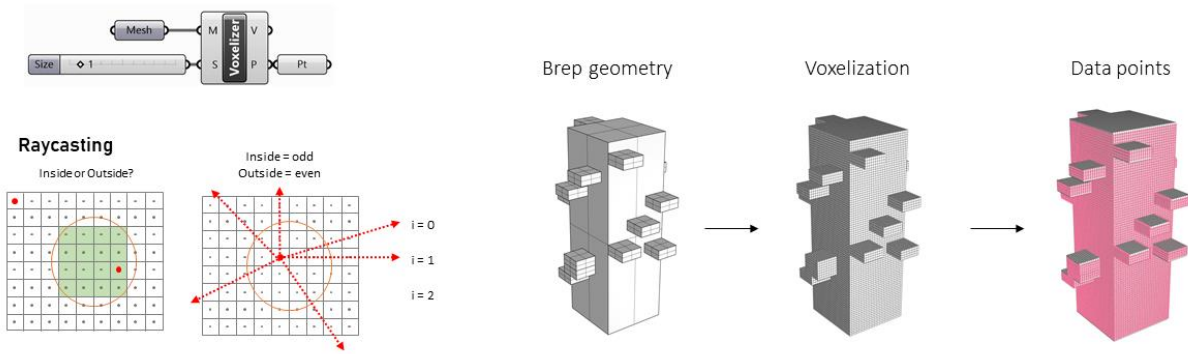


Figure 4.29: Ray-casting solution and algorithm application

4.2.1.2 Winding numbers

This algorithm was designed to compute its winding number by traversing over all triangles in meshes.

- To solve the problem of the open-boundaries, winding number methods were used to compute a sum of signed solid angles of each triangle (1).

$$w(p) = \frac{1}{2\pi} \oint_c d\theta \rightarrow \text{Naïve discretization} \rightarrow w(p) = \frac{1}{2\pi} \sum_{i=1}^n \theta_i \quad (1)$$

- Winding numbers: For 3D solid angle generalization (2)

$$w(p) = \frac{1}{4\pi} \iint_S \sin(\phi) d\theta d\phi \approx \frac{1}{4\pi} \sum_{f=1}^m \Omega_f \quad (2)$$

4.2.2 Surface and mesh generation

4.2.2.1 Marching cubes (3D)

Marching cube has 256 different situations as reference geometries, and those properties are generalized to 15 cases by rotations and symmetry. The stepwise example below shows the application of this algorithm.

- Step 1: Consider a cell defined by eight data values (Figure 4.30).

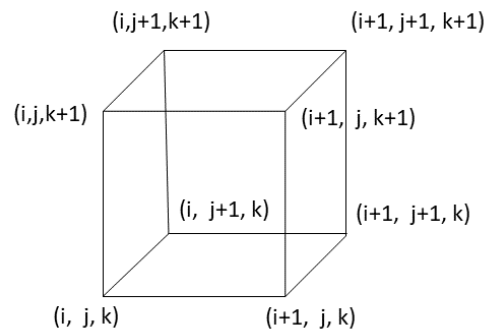


Figure 4.30: Marching cube points reference

- Step 2: Classify each voxel according to whether it lies outside the surface (value > isosurface value) and inside the surface (value ≤ isosurface value, Figure 4.31).
- Step 3: Use the binary labeling of each voxel vertex to create an index (256 possible different classification and finite cases, Figure 4.32).

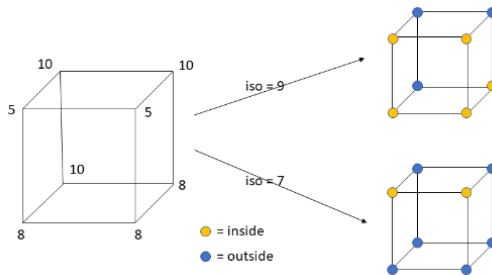


Figure 4.31: Voxel classification process

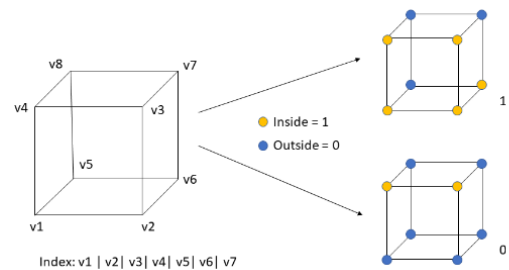


Figure 4.32: Voxel labeling process

- Step 4: For a given index, access an array storing a list of edges. All 256 cases can be derived from $1+14 = 15$ base cases due to symmetries (Figure 4.33). Edge list in the lookup table helps finding a optimal case (Figure 4.34).

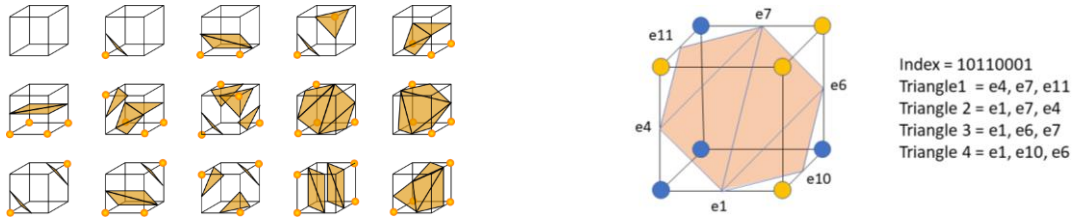


Figure 4.33: Symmetries from 256 cases

Figure 4.34: Get edge list from lookup table

- Step 5: For each triangle edge, find the vertex location along the edge using linear interpolation of the voxel values (Figure 4.35).

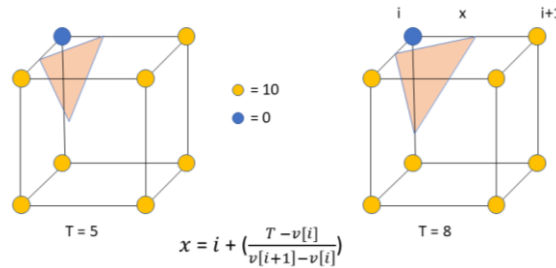


Figure 4.35: Find the location of the vertex along the edge

- Step 6: Calculate the normal at each cube vertex (central differences, Figure 4.36)

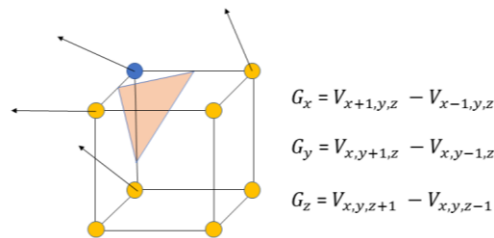


Figure 4.36: Calculate the normal at each cube vertex

- Step 7: Consider ambiguous cases (e.g., adjacent vertices – different states, diagonal vertices)

The algorithms explained above show the potential application of the middleware for surface-based simulation. This type of middleware software can be applied to grid-based simulations to help users convert buildings' geometry to a voxelated data structure for ANN inputs. Furthermore, the performance of the marching cube reveals the potential applicability to process ANNs outputs as the inputs of the BIM modeling interface.

4.2.3 Discussions and Findings

Often in early design, setting up one simulation case for one specific concept involves manually defining complex parameters (Nguyen et al., 2014). Furthermore, the runtime is relatively long and may interrupt the architect's train of thinking for the conceptual design; ideally, the software feedback time is less than 10 s (Miller 1968). As surrogates are evaluated instantly (< 0.1 s; Liesje et al., 2014), they can provide rapid feedback and an expeditious optimization. The ANNs-Solar ANNs-Airflow is the proper surrogate model, which is lightweight and can be embedded easily into any BPS software.

With the development of two middleware to process data, the use of ANNs-Solar and ANNs-Airflow algorithms in the CAD modeling interface could be completed (Figure 4.37). It was found that the 3DCNNs are the suitable ANNs architecture for estimating radiation intensities and airflow indoors on the buildings' facades yielding a high level of accuracy.

Therefore, the customized data exchangers for the 3DCNNs are essential for completing the modeling and consulting workflow. 3DCNNs are the best representation methods for voxelated data structures of the buildings' geometry and environments tested in the previous section. Using voxel-based CNNs as input, the proposed 3DCNNs (i.e., ARINet, CoolVox, and AirVox) can internally learn the underlying physics and geographical relationships.

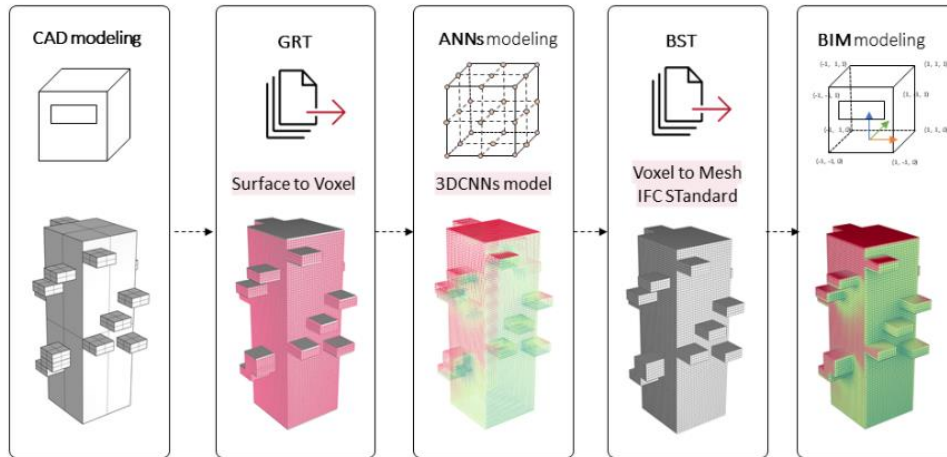




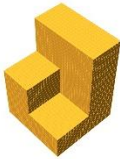


Figure 4.37: ANNs-solar workflow with 3DCNNs representation

Previous solar radiation studies used a voxelated matrix as input for the 3DCNNs, thus, including a large quantity of unnecessary information inside the matrix. This set of methods converts 3D shapes to sample representations and regularly applies a CNN to them. The voxel-based methods rasterize 3D shapes as an indicator or distance function sampled over dense voxels and then apply a 3DCNN over the entire 3D volume.

Defining a proper resolution for the voxel and maintaining the geometric properties of buildings are important. It can be shown that the voxelated surfaces for the inputs of the ANNs-Solar, 16^3 and 32^3 voxelated surfaces show distortions of the original geometries, mainly for the curvature of the facades and the shading elements and balconies on the facades (Appendix C). Starting from the 64^3 voxelated surface, the distortion percentage is reduced by maintaining the sophisticated surfaces of the buildings' geometries. However, increasing the size of voxels, increases the computation time significantly. Therefore, the accuracy and training time with different sizes of voxels on the CoolVox was tested. Table 4.5 lists the results of computation time for each iteration in seconds with validation errors. It was found that the model accuracy increases when training voxels with a 64^3 voxelated input matrix. Because the memory and computation costs grow

cubically as the voxel resolution increases, these methods can become tremendously expensive. According to Wang et al. (2017), Octree 3DCNN significantly reduced the computation time with a sparsely sampled matrix. In Wang’s model, Octree provides an efficient 3DCNN solution for 3D shape analysis. Because Octree grows quadratically as the Octree depth increases, it is suitable for analyzing high-resolution 3D models.

Table 4.5: The results comparison between the computation time and model accuracy

Basemodel	128³	64³	32³	16³
				
Iteration time, s	172	17	8	4
Error	0.141	0.158	0.221	0.286

One significant advantage of using 3DCNNs is that they maintain the orientation of the building because 3DCNNs preserve the voxel structure throughout the radiation prediction process. Consequently, the CoolVox model consistently predicts high radiation intensity on the south façade, low radiation intensity on the north façade, and varying radiation intensities on the east and west sides of the building. Therefore, Octree and other relevant methods will be tested in further studies designed to reduce the size of the different 3DCNNs. In addition, different types of ANN-based BPS models, such as airflow and building energy consumption, could be developed and tested using the proposed methods. Graph-based CNN models are other areas that need further investigation. Since 3DCNNs structures were optimized for both radiation and airflow predictions, the ways to reduce computational time are critical issues to be addressed. Along with the ideas of Octree CNNs and other efficient CNNs, graph CNNs should also be tested and developed to improve the proposed models

Chapter 5

Interoperability Framework

This chapter discusses the comprehensive interoperability framework for the data-driven BPS with a stepwise explanation. Figure 5.1 shows the proposed framework with essential components and models. It includes ANN models and two primary data exchangers, Data Exchanger 1 for the ANNs (for data pre-processing) and Data Exchanger 2 for IFC (for data post-processing), along with an ANN-based BPS workflow.

This framework illustrates the workflow and sub-processes for researchers trying to build a comprehensive interoperability framework for different CAD and BIM software packages with ANN models. When researchers develop their own ANN models for BPS research and practice, they can evolve the process and required algorithm for data conversion and implementation from CAD to BIM models.

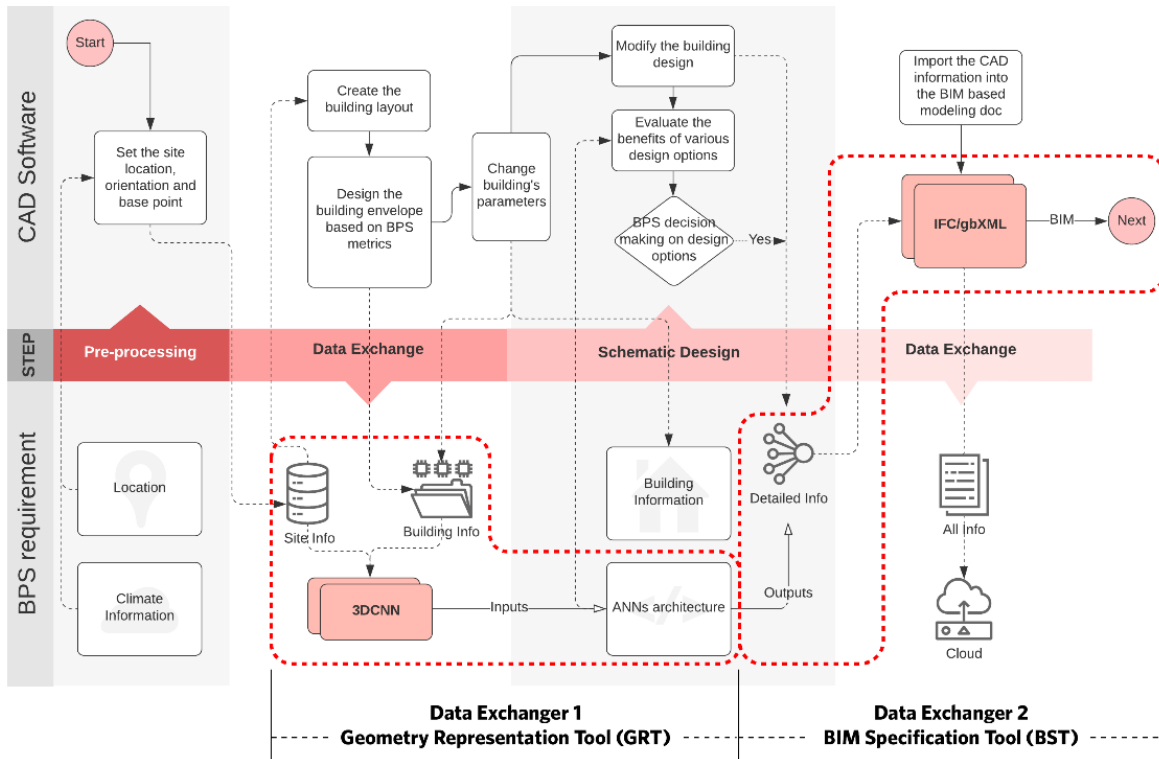


Figure 5.1: CAD/BIM to ANN-based BPS comprehensive framework.

Figure 5.2 illustrates the order of each sub-process and its connectivity. Section 5.1 explains the ANN models completed for BPS tasks related to solar radiation and airflow. Section 5.2 delineates the data conversion processes and details for developing two main data exchangers: the geometry representation tool (GRT) and the BIM specification tool (BST).

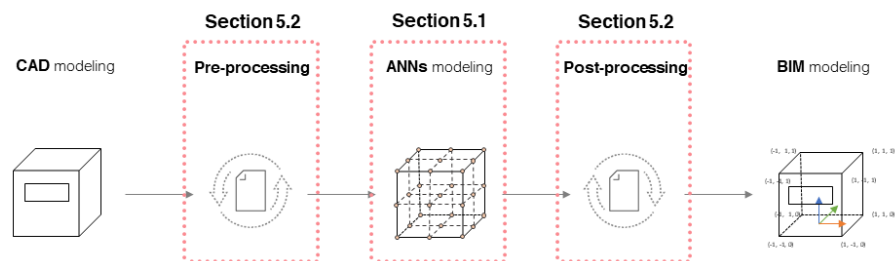


Figure 5.2: Detailed components for the chapter 5 and section numbers.

5.1 ANN and data modeling for different BPS tasks

Figure 5.3 illustrates the main contents for the section 5.1. As a result of chapter 4, 3DCNNs can be referred as the best method for representing building geometries and environments being tested in research on voxelated data structures. By using voxel-based 3DCNNs as inputs, they initially learn the underlying physics and geographical relationships.

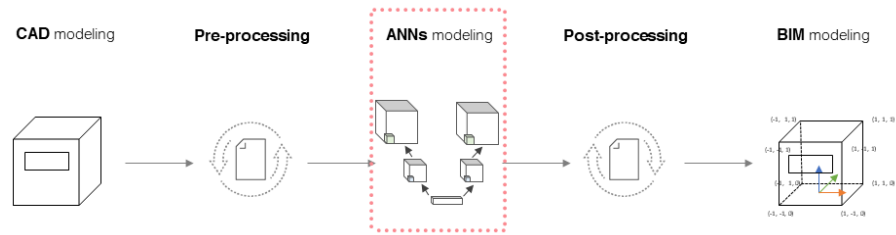


Figure 5.3: ANN models in the comprehensive framework.

Figure 5.4 shows the most suitable architectures of 3DCNNs for the solar radiation simulation and the airflow simulation respectively.

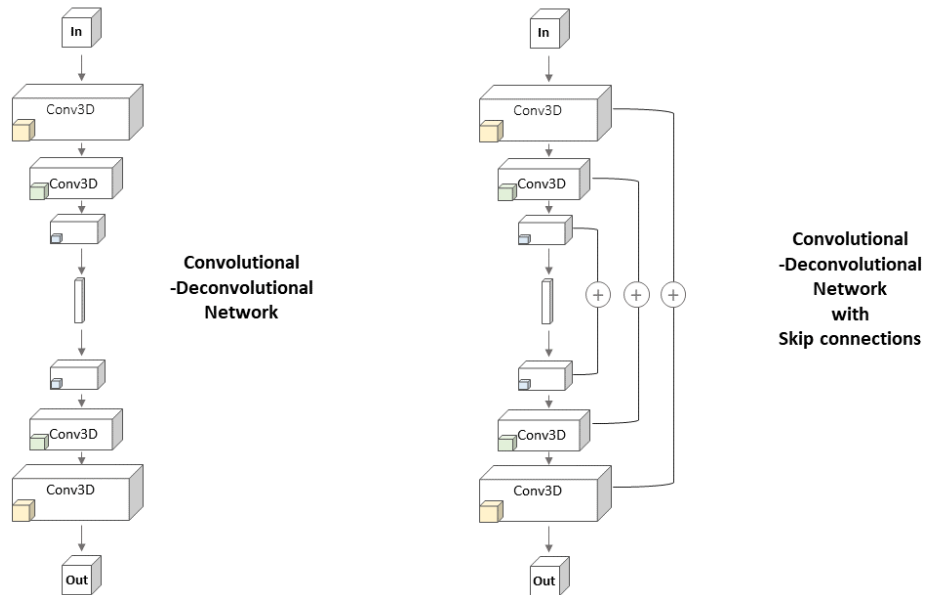


Figure 5.4: Final ANN models for different BPS tasks (left: solar radiation, right: airflow)

The previous section validated that the 3DCNNs outperformed conventional CNNs in predicting radiation and airflow. Among the various 3DCNN models, the convolutional and deconvolutional network types performed accurately for all the BPS tasks conducted in this study. The radiation results for the convolutional and deconvolutional (Conv-Deconv) networks had the lowest errors: 0.026 for the training and 0.033 for the validation sets (normalized total radiation, kWh). For the airflow indoors, it was found that the outlet airflow did not always follow the original pattern of Conv-Deconv networks. Thus, the skip connection layers were added atop the existing Conv-Deconv networks. The results from the Conv-Deconv skip model showed the closest pattern, specifically, inlet and outlet areas with the lowest MSE errors at 0.006 (normalized wind speed, m/s).

5.2 Development of data exchangers for different BPS tasks

The development of individually tailored plug-ins included a geometry representation tool (GRT) and a BIM specification tool (BST) (Figure 5.5). The GRT is middleware that transforms the building geometry and environmental information specified by CAD tools such as Rhinoceros, Revit, and AutoCAD into applicable geometric definitions through applications that support ANN training and prediction. The BST is a middleware that converts performance metrics and relevant data to BIM standard formats such as IFC. There are several rules for data translation and representation, including provisions for geometric conversion and numerical information transfer (as described in the previous section).

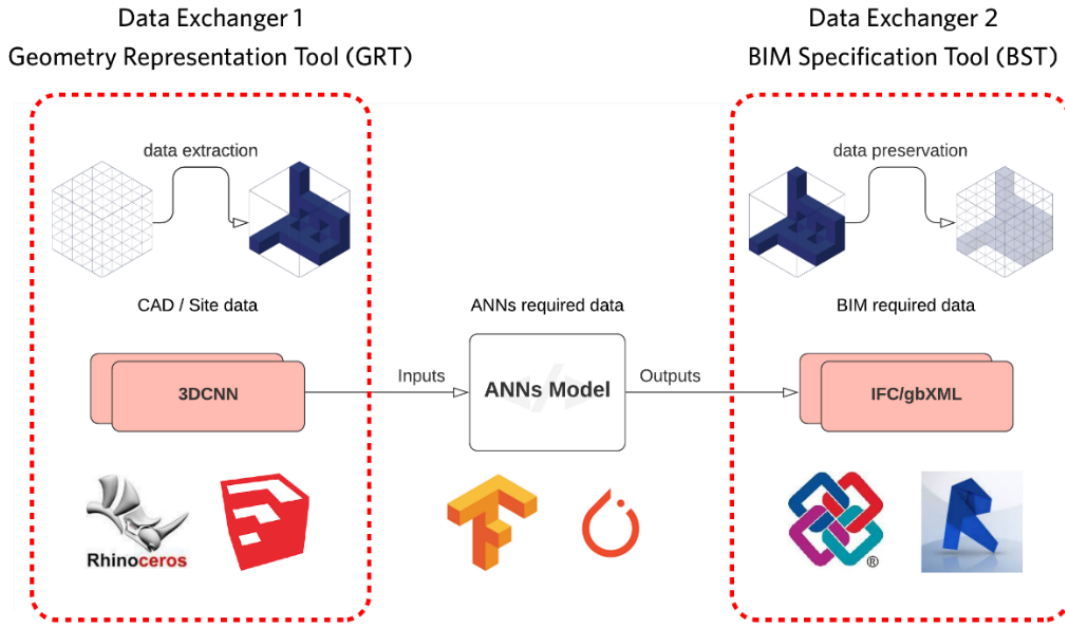


Figure 5.5: Main data exchangers: pre- and post-processing.

The GRT rules for data transformation consist of several steps already described in previous solar radiation studies. Some additional rules imbedded in the GRT include translating the cartesian coordinates of the CAD geometry into a voxelated 3D matrix, converting the performance metrics and environmental information into numerical values as arrays, identifying spaces and boundary conditions as different categorical values (i.e., binary or ternary padding), and encoding all numerical and categorical information as binary inputs for the 3DCNNs. The GRT can generate usable building information for ANN-driven BPS.

Additional rules imbedded in the BST include translating the coordinates of the geometry from the matrix to the IFC geometry definition (i.e., Brep/surface model, tessellated surface, Constructive solid geometry (CSG) primitive, and swept solid), transferring the BPS simulation results into a mesh-or graph-like IFC format, and preserving input and output for future BIM use. In this study, the IFC 2.3 and 4-example files were considered for BST prototype development and included lists of lists, binary representations, and tessellated geometry. Furthermore, the

IFC-OWL data schema was investigated to increase the reasoning, querying flexibility, and use of linked data in semantic web environments.

GRT- and BST- Solar and GRT- and BST- Airflow were developed and implemented to demonstrate the feasibility of using the required data structures. GRT-Solar takes voxelated 3D matrices as inputs to predict annual radiation intensities, while BST-Solar generates the boundary representation outputs of the given geometries to simulate future radiation for window control and operation. GRT-Airflow takes voxelated 3D matrices as inputs to predict indoor airflow, while BST-Airflow generates boundary representation outputs of given geometries to allow future airflow prediction for the buildings' opening design. Figure 5.6 explains the input for the GRT and output of the BST in the BPS simulation.

To accomplish the comprehensive framework, two middleware items were proposed to pre-process inputs and post-process the output data. The development of the specifically tailored plug-ins included the GRT and BST.

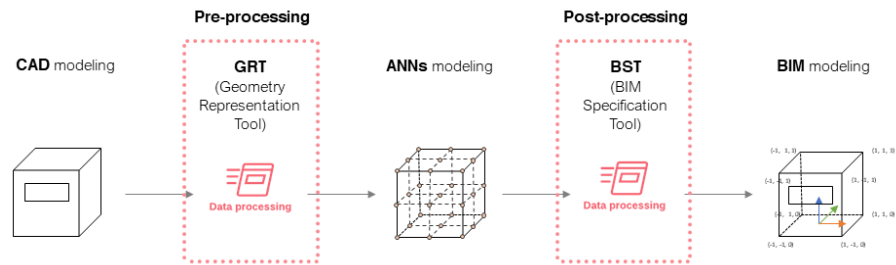


Figure 5.6: Data exchangers (GRT and BST) for ANNs-solar.

- **GRT** is middleware that transforms building geometry and environmental information specified by CAD tools such as Rhinoceros, Revit, and AutoCAD into applicable geometry definitions for applications that support ANN training and prediction.

- **BST** is middleware that converts performance metrics and relevant data into BIM standard formats such as IFC. There are several rules for data translation and representation, including rules for geometry conversion and numerical information transfer, as described in the previous section.

5.2.1. Data exchangers for solar radiation simulation

The study of solar radiation takes grid-based inputs as main reference points and produces visualized maps with values (Figure 5.7). The same simulation process was adopted to complete the workflow for solar radiation. In this section, the GRT and BST for solar are delineated through a stepwise process map and guidance.

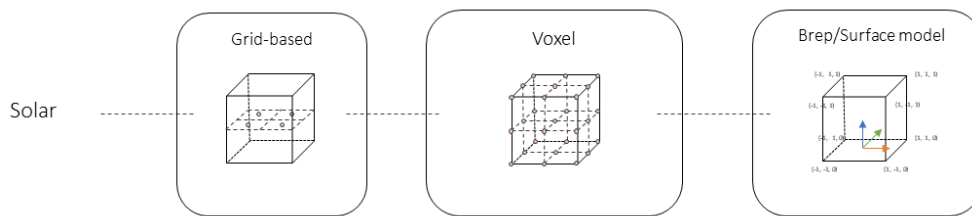


Figure 5.7: Data exchangers for solar diagram

5.2.1.1 GRT-Solar principles

GRT-Solar takes the Brep geometry as input and outputs voxelated grids mapped with pre-calculated distances and categorical values. The voxelated grids are populated with fixed parameters from the global coordinates in x-, y-, and z-dimensions. The 51 x 51 x 51 grids was used as a reference for the 3D voxel dimensions for GRT-Solar. Because most daylighting simulation standards require a 0.5 m or less grid-side as a minimum reference distance, the reference grid size was fixed to 0.5 m, i.e., the final voxel boundary dimensions in this study were 25 m x 25 m x 25 m equivalent grids. The 0-padded voxel grids were filled with values containing the different

characteristics of surfaces, such as voxel boundaries, voxel output grids, and voxel contexts. Figure 5.8 lists all the required inputs parameters for the ANNs-Solar.

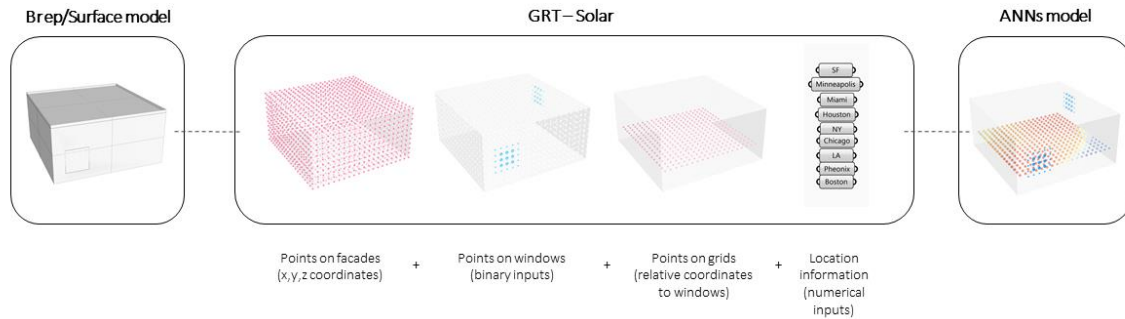


Figure 5.8: GRT-Solar input and output information.

Figure 5.9 shows the stepwise process for extracting building geometry for GRT-Solar. In Step 1, once users input Brep-type geometry that contains information regarding the walls, roof, slab, and openings, GRT-Solar automatically detects the different properties of the building blocks and separately tracks the geometry and property information. Step 2 separates the open and closed wall information with different categorical values. Lastly, GRT-Solar takes the internal layers of the closed walls to the indoor target plane and uses that surface to create the building voxel grids with output grids (Step 3).

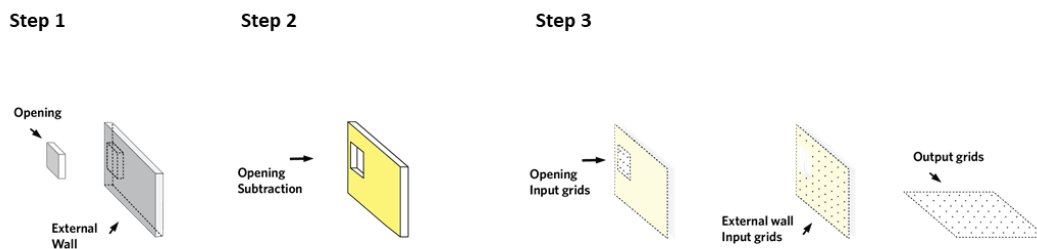


Figure 5.9: GRT-solar stepwise process.

Assuming all the populated points are located in the correct positions and match with the global voxel grids, the values were mapped through an input encoding process. Each voxel point in the

target building contains reference distance values from the openings and global reference points and weather information. Voxel grids outside the target building contain information indicating the existence of contextual buildings. Other than contextual buildings, all outside voxel areas are padded with 0s, further reducing the model size for training.

Figure 5.10 shows the process for encoding the input values for ANN modeling. For example, input information was designed with three different properties: air (indicating no building or no contextual building), façade (indicating a target building), and context (indicating contextual buildings). Therefore, air voxels are encoded as $[0, 0, 1]$, facades as $[0, 1, 0]$, and the context as $[1, 0, 0]$. Additionally, location information such as longitude and latitude are encoded with numerical values with tuple values (i.e., Boston = $[42.36, -71.05]$). Output grids are utilized later for the values predicted by ANN-Solar models.

Appendix D summarizes detailed information regarding the mapping methods for GRT-Solar. GRT-Solar vectorizes input lists into the voxel networks and uses both one-hot encoding and numerical methods to inform ANNs of the numerical relationships.

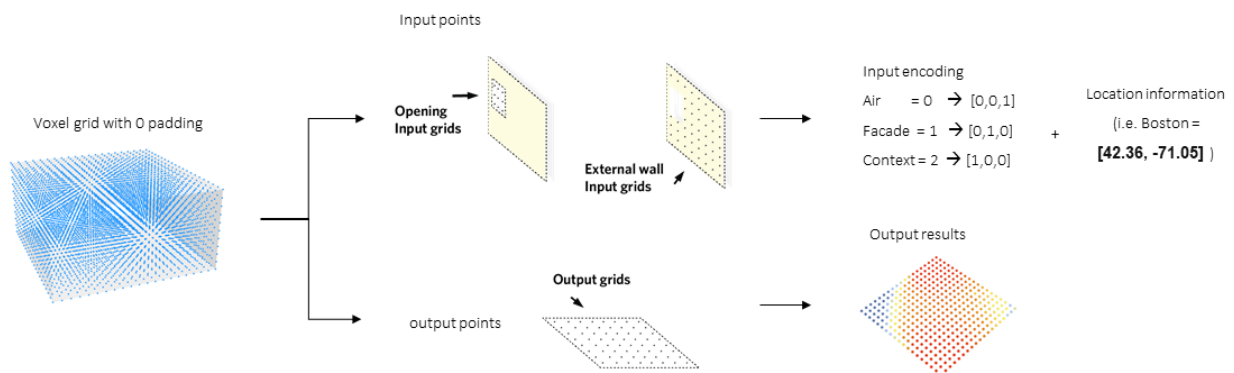


Figure 5.10: GRT-Solar value padding on each voxelated grid.

5.2.2.2 BST-Solar principles

BST-Solar communicates with software such as Revit. the IFC 2.3 and 4 data structures were chosen as a base model and processed the data to regenerate our model in both CAD and BIM software for further analysis. For a seamless workflow, all geometric information from the CAD models was tracked and stored in IFC data structures. Figure 5.11 shows how to accommodate each component in the relevant IFC data format.

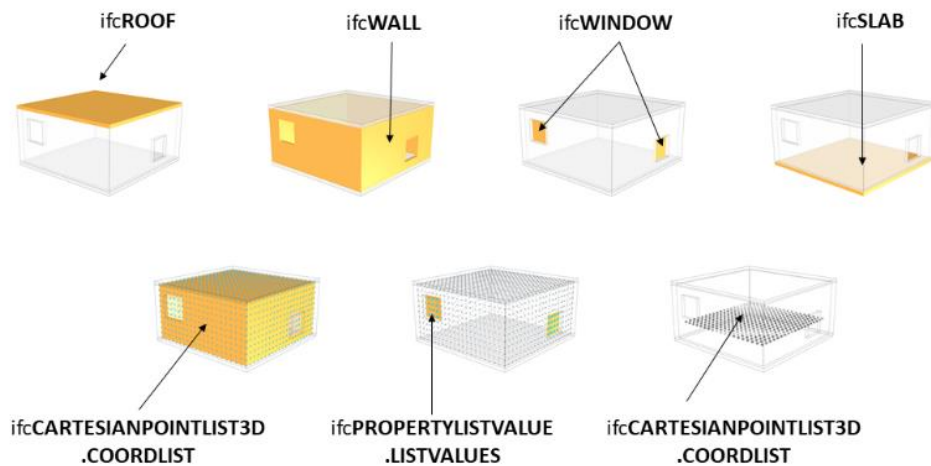


Figure 5.11: The IFC inputs for the CAD and voxel models.

Information delivery manual (IDM) provides guidance regarding the IFC framework. The IDM provides comprehensive references regarding information requirements to the AEC/FM industry; therefore, it is ideal to have an IDM for ANN-Solar to develop a consistent project-specific process and set of models. An information exchange requirement (ER) is a set of information from the information model that is applicable to the specific design stages. An ER should describe the information in non-technical terms. A functional part (FP) focuses on the individual actions within the delivery process, such as exchanging a building model. It is necessary to model walls, roofs, slabs, windows, etc. The action of modeling each of these elements is described within a

functional part. Business rules (BR) provide specific details about the business rules as textual expressions. BRs should contain informative notes and guidance. Appendix E lists the requirements for IDM-Solar with different functional and business parts. BST-Solar provides reliable information regarding geometric properties and functional parts to simulation practitioners for further analysis and certification. Appendix E lists all required modeling properties and simulation details needed to document any further analysis.

5.2.2 Data exchangers for airflow simulation

In practice, the study of indoor airflow distribution takes voxel-based inputs from the mesh-like geometry as main reference points and produces a visualized map with values (Figure 5.12). The same simulation process from the conventional CFD simulation was utilized to complete the workflow for airflow. This can be applied to different types of CFD software and various CAD software.

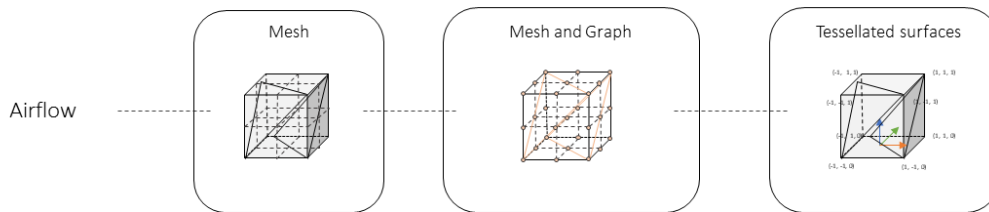


Figure 5.12: Data exchangers for airflow diagram.

The scripts for each data exchanger are coded in Python3 and will be made available to the community. In this section, the GRT and BST for airflow are delineated with a stepwise process map and guidance. The general workflow is similar to the data exchanger for solar, but the detailed information and voxel representation methods are differentiated to optimize training and the prediction process.

5.2.2.1 GRT-Airflow principles

GRT- Airflow mainly takes the mesh-like geometry as input and outputs a voxelated volumetric grid. Figure 5.13 illustrates the GRT workflows for airflow modeling and simulation. The voxelated grids are populated with fixed parameters from the world coordinates in the x-, y-, and z-dimensions. The 32 x 32 x 32 grids were created as references for the 3D voxel dimensions for GRT-Airflow. The os padded voxel grids are filled with values containing different surface characteristics, such as voxel boundaries with inlets and outlets and volumetric output grids.

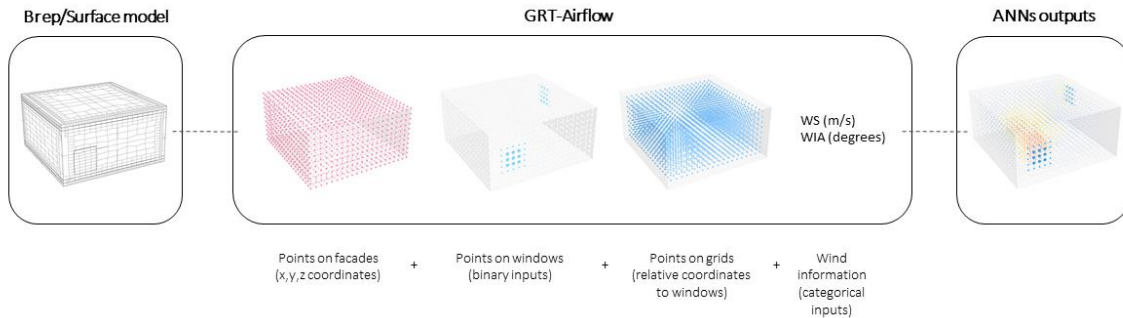


Figure 5.13: GRT-Airflow input and output information

Figure 5.14 shows the stepwise process for extracting building geometry for GRT-Airflow. In Step 1, once users input the mesh-type geometry containing information regarding the walls, roof, slab, and openings, GRT-Solar automatically detects the different properties of the building blocks and separately tracks the geometry and property information. Unlike GRT-Solar, GRT-Airflow asks users to decide the locations of inlets and outlets for flow simulation. Step 2 separates the openings and closed-wall information with different categorical values. Lastly, GRT-Airflow takes the internal layers of the closed walls to the indoor target plane and uses those surfaces to create an array of voxel grids with output grids (Step 3).

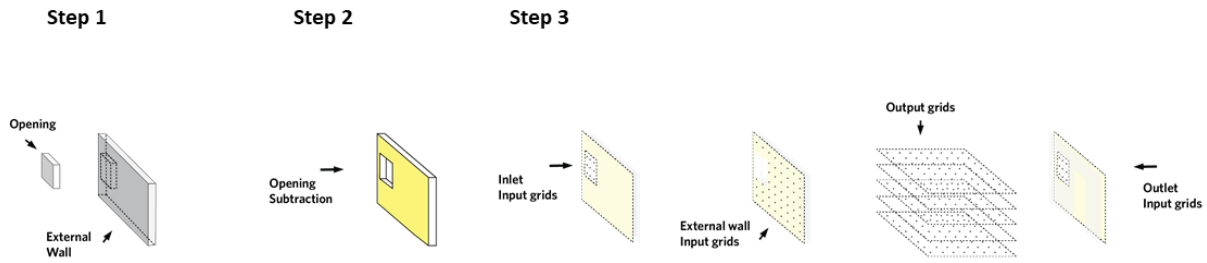


Figure 5.14: GRT-Airflow stepwise process diagram

With a given volumetric grid in a virtual box (i.e., $32 \times 32 \times 32$ grids), the locations of target buildings are mapped in the voxel grids with a value of $[1,0,0]$, representing building facades (Figure 5.15). In the next step, the values mapped on the facades are replaced with the openings information. GRT-Airflow requests replacing values for the inlets $[0,0,1]$ and outlets $[0,1,0]$ instead of $[1,0,0]$. This informs the ANNs, allowing them to understand airflow indoors with the relevant locations of inlets and outlets. Additionally, GRT-Airflow requires wind information to be input, such as the direction and speed from the inlet. Output volumetric grids are mapped with the relative cartesian coordinate values for analysis. Appendix F summarizes the required information and structure of the inputs for GRT-Airflow. GRT-Airflow vectorizes the input lists into the voxel networks and uses both one-hot encoding and numerical methods to inform ANNs, allowing them to learn the numerical relationships.

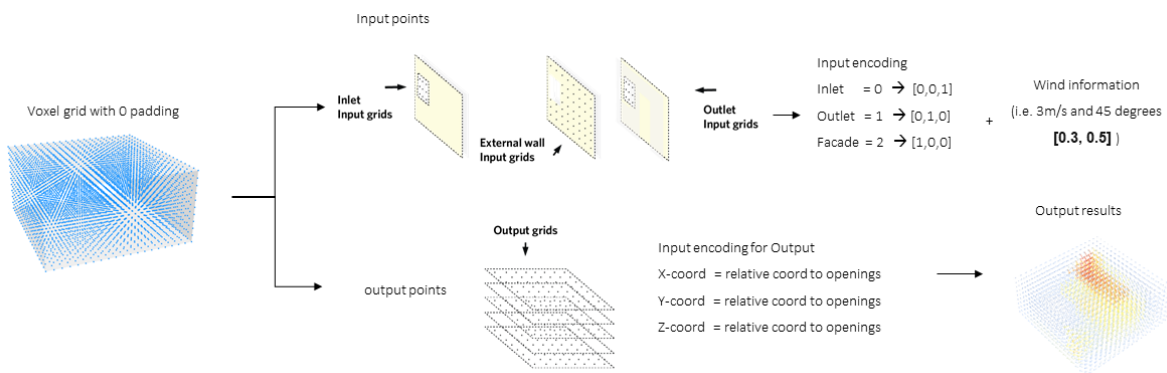


Figure 5.15: GRT-Airflow value padding on each volumetric grid.

5.2.2.2 BST-Airflow principles

BST- Airflow communicates with different software packages such as Revit. Similar to BST-Solar, BST-Airflow stores modeling properties and detailed information in the form of IFC data structures. Figure 5.16 shows that BST-Airflow preserves the information of the walls, roof, slab, and window openings. Additionally, it preserves the location information of the volumetric grids on the internal layers of facades. Later, it matches the populated voxel grids to the simulation results and sends information to the energy modeler and other engineers for reference regarding indoor airflow performance. All architectural geometries taken for the IFC 2.3 are in the form of Breps, meshes, and solid extrusions. BST-Airflow stores a mesh-type geometry after processing inputs for IFC-equivalent components such as IFC-Wall.

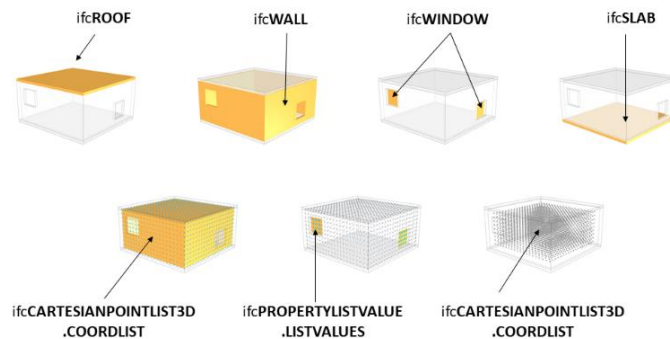


Figure 5.16: The IFC inputs for CAD and ANNs-based BPS models

In addition to architectural properties, IFC takes coordinates and property value lists as additional documentation. Therefore, BST-Airflow extracts all volumetric grids and equivalent values as lists of lists and delivers them to practitioners. Appendix G summarizes the type requirements for each property as different data structures that enable users to deliver modeling information for use in different software packages such as Revit. The structured coordinates and relevant numerical values can be stored as arrays and lists with a text description of simulation parameters (i.e., OpenFoam parameters) and detailed location information (Appendix G)

Chapter 6

Validation

This section discusses the application of the comprehensive interoperability workflow for the data-driven models. Two CAD modeling software packages (i.e., Rhinoceros and Revit) and two BIM software packages (i.e., Revit and OpenBIM) were used to demonstrate the feasibility of the proposed workflow (Figure 6.1).

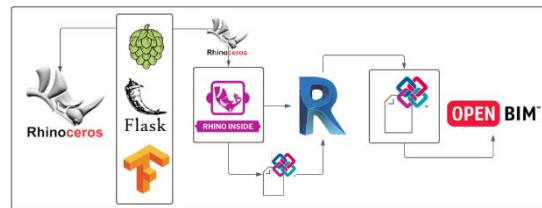


Figure 6.1: Software used for the workflow demonstration

The Python 3 package compatible with the DL models was coded and distributed via GitHub for use by researchers around the world. The Grasshopper script and Python-enabled interface in Rhinoceros were utilized and the building geometry in both Rhinoceros and Revit was manipulated interactively. The traditional CAD modeling tools inevitably incorporated the

6.1 Python packages and software integration

Python packages were developed for the GRT and BST. Both functions were coded using Python3 and released online as a package. Users can easily download this package and use it as a means of preparing to run ANNs for solar or wind. Figure 6.4 delineates the inputs and outputs represented in the modeling interface. To demonstrate the workflow, a simple box-shaped geometry with different opening sizes and locations was used. The Python package contains two main functions: GRT and BST. The GRT is the pre-processing component that allows network input preparation for ANNs. It takes the internal surface geometry as input and passes the voxelated networks to the ANN models. The predictions are plotted instantaneously in the Rhino view with a relevant color legend and given geometries. In this demonstration, the regularly populated voxel grids were used, which represent the positions for the SDA results and 3D wind vectors.

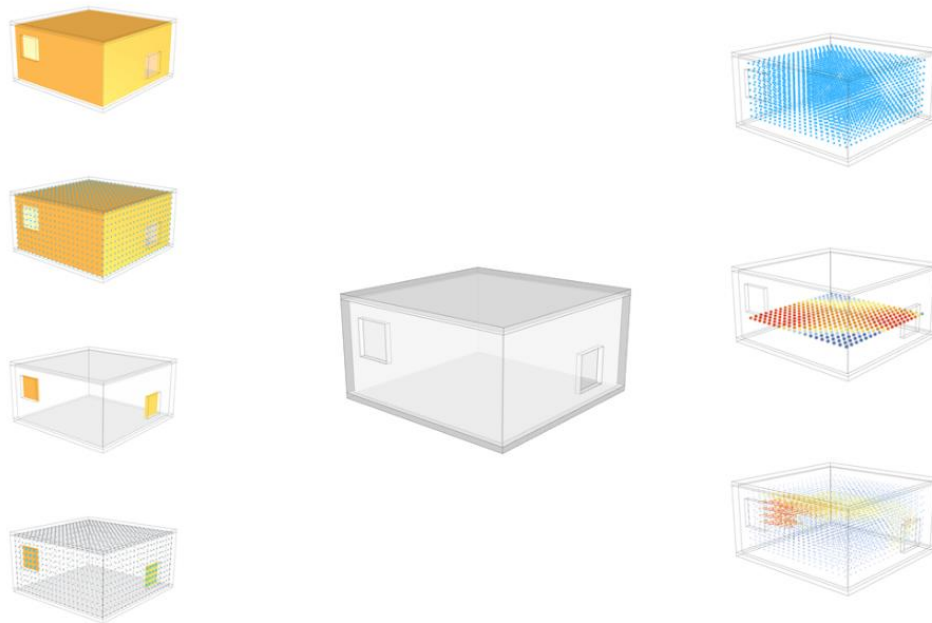


Figure 6.4: Analysis properties and representation for CAD software (GRT).

The BST is the post-processing component; thus, it converts IFC geometries and property information and translates coordinate information with IFC details to complete the IFC data file format. The BST mainly takes two types of inputs, extruded models such as Breps and surface models for populating the positions of the voxel grids with given geometries. BST includes the workflow for converting a CAD geometry into a BIM model, which can be directly shown in the Revit interface. In the Grasshopper canvas for Rhino, users specify the details for IFC geometries such as walls, roofs, and slabs. RhinoInside is a Revit plugin that directly imports, exports, or transfers the workflow from Rhino to Revit. When a user manipulates a geometry from Rhino, the user sees the changes in Revit.

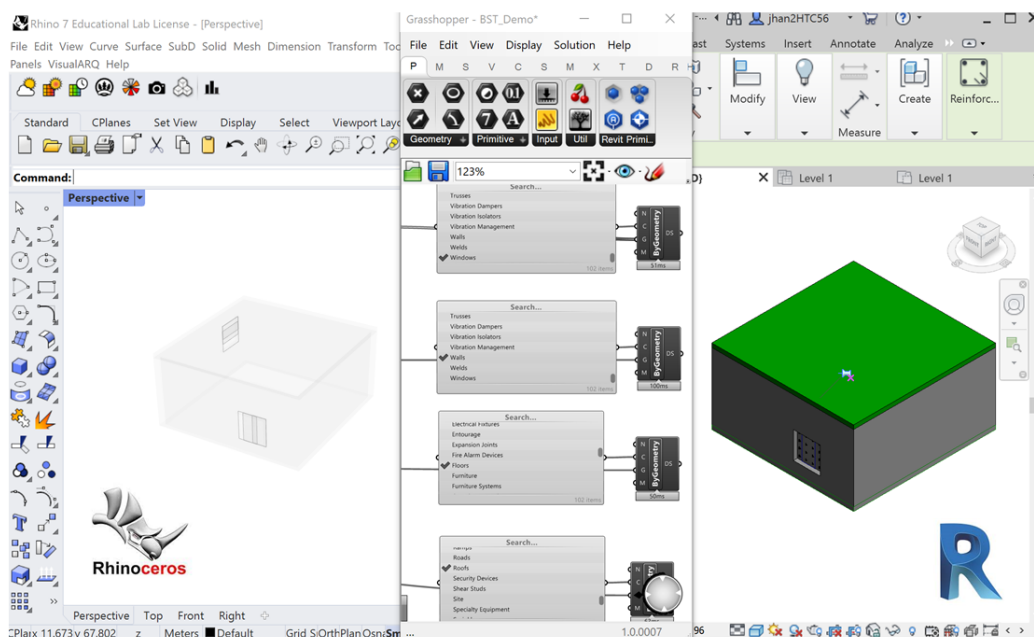


Figure 6.5: Analysis properties and representation for BIM software (BST).

After transferring models from Rhino and Revit, users must define the IFC properties of the models in .txt format. This exercise used IFC 4 as an example and transferred the converted information of walls, roofs, slabs, and the internal meshes with point coordinates and other

weather information to the IFC 4. Coordinates with values mapped on grids were converted into an array structure (a list of lists) along with reference points defining the building's indoors and location and use of functional windows. The BST has two different templates for IFC versions 2.3 and 4. These can be converted to IFC 4, but if the software only supports IFC 2.3, then the user can select the option for IFC 2.3 to store the ANN model information. Since the OpenBIM viewer supports both IFC 2.3 and 4, this demonstration used the most recent version of IFC. The last step of the demonstration involved importing the IFC file from the ANN models and visualizing its functional parts in the BIM viewer (Figure 6.5).

6.2 Discussion

This demonstration illustrates the comprehensive workflow for ANN models in a sequenced manner. This demonstration employed the software for CAD and BIM that is most widely used in the academy and industry. In addition, this demonstration utilized advanced ANN models for BPS tasks for future adaptations of the proposed tools. In addition, the applicability of the GRT and BST in real-world applications was demonstrated by introducing the currently available BIM and IFC software. Python3 packages can easily be integrated into Rhino and Revit by using Grasshopper and Dynamo (Python-enabled interfaces) for modeling tools. This package enables the direct use of models as ANN inputs and the conversion of important model properties into the IFC data structure for future use and model adaptation. Storing information is essential for communicating with different stakeholders at any stage of the design and consulting process and later submission of models for certification processes such as LEED and other relevant standards.

Chapter 7

Conclusion

This dissertation evaluated the interoperability between BIM-based and ANNs-based BPS models in CAD software. Based on the proposed interoperability framework, the future adaptability of the proposed software in the age of ANN-based BPS modeling was discussed.

The data-driven methods and models proposed in this study reduce the computation time and modeling effort, reducing gaps in modeling expertise. The proposed 3DCNN models can lead to better design decisions at early stages, resulting in fewer post hoc design modifications across different software packages and platforms. The key components that serve to resolve these issues are the geometry representation tool (GRT) and the BIM specification tool (BST). The GRT and BST are essential elements that help maintain sustainable modeling and consulting workflows. The GRT and BST were tested in different types of CAD and BIM software, and the workflow was delineated through a relevant case study. This research establishes comprehensive modeling and data conversion methods for BPS.

Future work will focus on developing advanced ANN models that combine physics-based models with a robust geometric representation algorithm for ANN-based BPS simulations. Combining classical physics-based models with ANNs is an exciting avenue for future research. Finding optimal datasets to train ANNs is difficult. Therefore, the combined use of both models in BPS is a promising direction because of the amount of available data, computational time efficiency, and increased accuracy. For example, an ANN model trained with locally available data and embedded in a physics-based sky model could generate a realistic local environment by substituting probabilistic models for deterministic equations. Conversely, the accuracy of ANN models can be increased by utilizing physics-based knowledge in their optimization. This new approach will help overcome the existing limitations of data-driven models and enhance their potential applicability to real-world problems, serving as a novel transition, converting physics-based modeling practices into future data-driven adaptations aligned with my current research.

Because the geometries of architectural buildings encompass more complex properties than BPS modeling, more efficient and robust geometric conversion techniques are required to complete ANN-based BPS frameworks. Currently, the BPS software deals with the simplified geometry needed to calculate performance metrics. Methods for modeling complex CAD-based geometries for ANNs have yet to be investigated. Therefore, the algorithms and techniques currently used by the computer vision industry should be further explored to reduce the model size and increase the potential applications of the complex geometry used in ANN modeling methods.

Appendix

Appendix A. Post-IFC Tools and Development

Integration	Tools Overview	Domain	Intelligence	Performance Analysis	Schema
Pre-Standard Integration					
IDD & GLIDE (Eastman and Henrion 1977)	Abstraction hierarchies with analysis and synthesis models	Standalone	Rule-based Expression-oriented	-	-
DIS & CAEADS (C. M. Eastman 1979)	Design exploration using GLIDE and abstract representations	Standalone	Rule-based	Structural, thermal, cost, piping, and distribution sizing analyses	-
ARMILLA (Gauchel et al., 1992)	Modular building approach	Standalone	Dynamic Constraint-based		-
AEDOT using ICADS (Pohl, J., LaPorta, J., Pohl, K. J., and Snyder 1992)		Standalone	Rule-based	Energy standards, building mass, daylighting	-
Post-STEP Standard Integration					
COMBINE COMBINE 2 (Augenbroe 1993)	Integrated environment for energy and HVAC tools in COMBINE; used Petri Nets concepts in COMBINE-2	Standalone		Energy, HVAC tools	EXPRESS, STEP
KNODES (Rutherford 1993)	Knowledge-based design framework	Standalone	Knowledge-based	Natural lighting, energy, energy design, spatial analyzer, structural, costing	EXPRESS, G
SEMPER (Mahdavi et al., 1997)	Active, multi-aspect design environment with dynamic links to performance valuation tools	Standalone	KBES for providing thermal comfort feedback Investigative project technique	Thermal (NODEM), airflow (Hybrid multi-zone, CFD), HVAC, thermal comfort (algorithmic routines, KBES), lighting	Shared Object Model































				(radiosity), acoustics (hybrid stochastic)	
Building Design Advisor (BDA) (Papamichael et al., 1997)	Process-logic control for automating activation processes	Standalone		Multi-criteria analysis based on light illuminance, energy use	BDA data meta schema
Post-IFC Standard Integration					
DAI (Augenbroe et al., 2004)	Four-layered process-centric workbench (design information, structure simulation models, analysis scenarios, and software tools)	Standalone	Process modeling and enactment (analysis)	Thermal (EnergyPlus, PMV), daylight autonomy (IDEA-L)	IFC, XML
SEMPER II (Lam et al., 2004)	Web-based active, multi-aspect design environment that uses XML for data transfer	Standalone	Same as SEMPER	Same as SEMPER	IFC, XML
DeST (Yi et al., 2007)	Locates an efficient and precise algorithm for mapping data between DeST and XML-formatted schema	Module-based		Climate analysis, lighting, ventilation systems, building loads, economic models	IFC
Dynamic Building Model (Grzybek et al., 2010)	Open, dynamic, and temporal building model for intelligent adaptable buildings		Inclusion of temporal databases in IFC	Thermal (test case)	IFC
Post-gbXML Standard Integration					
BCVTB (Wetter, 2011) SimModel (J. O. Donnell et al., 2011) Simergy (Alchemy, 2013)	BCVTB: integrated building energy and control systems software; SimModel: data interoperability services; Simergy: offers linkages to BCVTB and SimModel	Standalone	Matlab routines (e.g., optimization) is accessed in BCVTB)	Thermal (EnergyPlus, Modelica library), lighting (radiance), HVAC and controls (Modelica library), controls (Simulink)	IFC, XML, BIM gbXML
CBEMS (S. Wang et al., 2011)	Web- and BEMS-based four-tier architecture (data acquisition and interface,	Web-based	Policy learning (self-learning and self-computing),	Energy, lighting, plug-loads	XML

	automatic computing and execution, management and monitoring)		Nash equilibrium		
DYNAMIC-BIM (R. Srinivasan et al., 2012)	Open-source environment Ptolemy / Revit interface for data transfer / Revit plug-in		Process modeling	In progress, including integration of 3D heat transfer, energy analysis, daylighting	Enhances IFC for dynamic data acquisition
LadyBug (Roudsari and Pak 2013) (LadyBugTools 2018)	Plug-ins for Rhino and Grasshopper	Open-source Web-based		Load calculation and daylighting simulations	gbXML
BIM2BEM (Jeong et al., 2014)	BIM2BEM framework and prototype Revit2Modelica	Plug-ins	BIM and object-oriented physical modelling	Modelica-based building energy models	BIM API (i.e., Revit API)
IFC to IDF (Kim et al., 2012) gbXML to IDF (Dimitriou et al., 2016c)	IFC-based input data file converter		Inclusion of databases in IFC	Automated data converter/ EnergyPlus analysis	IDF / IFC / gbXML
FloorspaceJS (Macumber, Horowitz, Schott, Noland, et al., 2018)	Open-source, web-based geometry editor for BEM	Web-based		Users define building geometry story-by-story with custom 2D floor plans	JSON
IDF to JSON (New et al., 2018)	JSON-based IDF converter		Inclusion of databases in IFC to JSON		JSON
Design4Energy (Arayici et al., 2018)	Interoperability specifications for integrated BIM practice	Specification framework		Monitoring of carbon dioxide emissions (CO ₂)	IFC

Appendix B: Key terminology of neural networks

Term	Explanation
feed-forward neural networks	a basic type of neural network
convolutional neural networks	a neural network with the hidden layers that perform convolutions
layer	a vector-valued variable serving as input, output in neural networks
loss function	a scalar-valued function to be minimized during the training process
stochastic gradient descent	a commonly used optimization algorithm for training neural networks
learning rate	step size of the iterative gradient-based optimization algorithm
epoch	a full pass through all training data in stochastic algorithms
batch size	number of data points used to estimate gradients in one iteration
model hyper-parameters	external configuration of a network and the training process, such as the number of hidden layers, number of nodes per layer, activation function, learning rate, etc.
train, validation, test sets	the whole dataset is split into train and validation for training and tuning and newly create a test dataset for evaluating a model
early-stopping	a regularization technique that controls the training time in order to prevent overfitting

Appendix C: The different resolutions of voxelated surfaces for the ANNs-Solar

Basemodel	128^3	64^3	32^3	16^3
				
				
				
				
				
				

Appendix D. Summary of GRT-Solar

GRT-Solar		
Vox1	3D positioning of the reference points	3D matrix - X.shape = (number of examples, xdim, ydim, zdim)
Vox2	output mapped on the points	3D matrix - Y.shape = (number of examples, xdim, ydim, zdim) MINMAX normalized
Vox3	relationship of reference point to space	[0,1,0]: Indoor [0,0,1]: air
Vox4	relationship of surrounding surfaces to space	[1,0,0]: surface of buildings
Vox5	relationship of opening to space	os: air and customized loss function
List6	location information	[Lat-coord, Long-coord]

Appendix E. Requirements for BST-Solar

Requirement	Data type	Description	IFC class name
FP1	Surface	Walls, Roofs, Slabs and openings	IFC wall, roof, slab and so on
FP2	3D array	Input and output grids	IfcCartesianPointList3D.CoordList
FP3	Array	Relationship between opening and grids	IfcPropertyListValue.ListValues
FP4	Array	Distance between opening and grids	IfcPropertyListValue.ListValues
FP5	Text	Geographic location	IfcPostalAddress
BR1	Value list	Radiation	IfcPropertyListValue.ListValues
BR2	Text	Material information	IfcMaterialList.Materials
BR3	Text	Radiance parameters	IfcPropertyListValue.ListValues

Appendix F. Summary of GRT-Airflow

GRT-Airflow		
Vox1	3D position of the reference points	3D matrix - X.shape = (number of examples, xdim, ydim, zdim)
Vox2	output mapped on the reference points	3D matrix - Y.shape = (number of examples, xdim, ydim, zdim) MINMAX normalized
Vox3	relationship of reference point to inlet	[X-coord, Y-coord, Z-coord]
Vox4	relationship of reference point to outlet	[X-coord, Y-coord, Z-coord]
Num5	wind speed	[0-1]
Num6	wind direction	[0-1]

Appendix G. Requirements for BST-Airflow

Requirement	Data type	Description	IFC class name
FP1	Surface	Walls, Roofs, Slabs and openings	IFC wall, roof, slab and so on
FP2	3D array	Input and output grids	IfcCartesianPointList3D.CoordList
FP3	Array	Relationship between opening and grids	IfcPropertyListValue.ListValues
FP4	Array	Distance between opening and grids	IfcPropertyListValue.ListValues
FP5	Text	Geographic location	IfcPostalAddress
BR1	Value list	Radiation	IfcPropertyListValue.ListValues
BR2	Text	Material information	IfcMaterialList.Materials
BR3	Text	Radiance parameters	IfcPropertyListValue.ListValues

Bibliography

- Abdul-Rahman, A., and Pilouk, M. (2008). Spatial data modelling for 3D GIS. *Spatial Data Modelling for 3D GIS, January*, 1–289. <https://doi.org/10.1007/978-3-540-74167-1>
- Adelson, E. H., Anderson, C. H., Bergen, J. R., Burt, P. J., and Ogden, J. M. (1984). Pyramid Methods in Image Processing. *RCA Engineer*, 29(6), 33–41.
- Afsari, K., Eastman, C. M., and Shelden, D. R. (2016). Cloud-based BIM data transmission: Current status and challenges. *ISARC 2016 - 33rd International Symposium on Automation and Robotics in Construction*, 1073–1080. <https://doi.org/10.1177/0890334412444350>
- Afzal, H., Aouada, D., Font, D., Mirbach, B., and Ottersten, B. (2014). RGB-D multi-view system calibration for full 3D scene reconstruction. *International Conference on Pattern Recognition*, 2459–2464. <https://doi.org/10.1109/ICPR.2014.425>
- Ahmad, M. W., Mourshed, M., and Rezgui, Y. (2017). Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. *Energy and Buildings*, 147, 77–89. <https://doi.org/10.1016/j.enbuild.2017.04.038>
- Ahmed, E., Saint, A., Shabayek, A. E. R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., and Ottersten, B. (2018). *A survey on Deep Learning Advances on Different 3D Data Representations*. 1(1), 1–35.
- Akbarnezhad, A., Ong, K. C. G., and Chandra, L. R. (2014). Economic and environmental assessment of deconstruction strategies using building information modeling. *Automation in Construction*, 37, 131–144. <https://doi.org/10.1016/j.autcon.2013.10.017>
- Alchemy, D. (2013). *Simergy*.

- Andrew Witkin, D. T. and M. K. (1987). Signal Matching Through Scale Space. *International Journal of Computer Vision*, 144, 295–298. <https://doi.org/10.1109/DCABES.2014.74>
- Arayici, Y., Fernando, T., Munoz, V., and Bassanino, M. (2018). Interoperability specification development for integrated BIM use in performance-based design. *Automation in Construction*, 85(October 2017), 167–181. <https://doi.org/10.1016/j.autcon.2017.10.018>
- Ascione, F., Bianco, N., Stasio, C. De, Maria, G., and Peter, G. (2017). Artificial neural networks to predict energy performance and retro fit scenarios for any member of a building category : A novel approach. *Energy*, 118, 999–1017.
- Attia, S. G. M., and Herde, A. De. (2011). early design simulation tools for net zero energy buildings: a comparison of ten tools. *Ibpsa, February 2016*.
- Augenbroe, G. (1993). *An Overview of the COMBINE project*.
- Augenbroe, G., De Wilde, P., Moon, H. J., and Malkawi, A. (2004). An interoperability workbench for design analysis integration. *Energy and Buildings*, 36(8), 737–748. <https://doi.org/10.1016/j.enbuild.2004.01.049>
- Augenbroe, G., De Wilde, P., Moon, H. J., Malkawi, A., Brahme, R., & Choudhary, R. (2003). The design analysis integration (DAI) initiative. In 8th IBPSA Conference (pp. 79-86).
- Azar, S. P. E. (2016). *Integrating building performance simulation in agent-based modeling using regression surrogate models: A novel human-in-the-loop energy modeling approach*.
- Azeriel Rosenfeld, A. C. K. (1976). Digital picture processing. Academic press.
- Azriel Rogenfeld, J. L. P. (1966). Sequential Operations in Digital Picture Processing. 13(4), 471–494.

- Babovic, V., Cañizares, R., Jensen, H. R., & Klinting, A. (2001). Neural networks as routine for error updating of numerical models. *Journal of Hydraulic Engineering*, 127(3), 181-193.
- Baker, H. (1977). *Three dimensional modeling*. University of Illinois at Urbana-Champaign.
- Barrow, H. G., and Tenenbaum, J. M. (1981). Interpreting line drawings as three-dimensional surfaces. *Computer Vision*, 17(3), 75-116.
- Baumgart, B. G. (1974). *Geometric modeling for computer vision* (Issue October). Stanford University, department of computer science.
- Bazjanac, V. (2008). IFC BIM-based methodology for semi-automated building energy performance simulation. *CIB-W78 25th International Conference on Information Technology in Construction*, 292-299.
- Bazjanac, V. (2010). Space boundary requirements for modeling of building geometry for energy and other performance simulation. *Lawrence Berkeley National Laboratory, University of California, Berkeley, California, USA, November*, 16-18.
- Bazjanac, V., and Crawley, D. B. (1999). Industry Foundation Classes and Interoperable Commercial Software in Support of Design of Energy-Efficient Buildings. *5th IBPSA*.
- Beardsley, P., Torr, P., and Zisserman, A. (1996). 3D model acquisition from extended image sequences. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1065, 684-695.
https://doi.org/10.1007/3-540-61123-1_181
- Becerik-Gerber, B., and Rice, S. (2010). The perceived value of building information modeling in the U.S. building industry. *Electronic Journal of Information Technology in Construction*, 15(May), 185-201.

- Behrman, W. (2002). Best practices for the development and use of XML data interchange standards. *Center for Integrated Facility Engineering Technical Report, 131*, 27.
- Bell, H., & Bjørkhaug, L. (2020). A buildingSMART ontology. In *eWork and eBusiness in Architecture, Engineering and Construction* (pp. 185-190). CRC Press.
- Bijaoui, A., and Giudicelli, M. (1990). Optimal image addition using the Wavelet Transform. *Experimental Astronomy, 1*(6), 347–363. <https://doi.org/10.1007/BF00426718>
- Boscaini, D., Masci, J., Rodolà, E., and Bronstein, M. M. (2016). *Learning shape correspondence with anisotropic convolutional neural networks*. 1–13.
- Brice, C. R., Fennema, C. L., and Journal, A. I. (1970). *Scene analysis using regions*. April.
- Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2016). *Generative and Discriminative Voxel Modeling with Convolutional Neural Networks*.
- Bronstein, M. M., Bruna, J., Lecun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine, 34*(4), 18–42. <https://doi.org/10.1109/MSP.2017.2693418>
- Brown, M., and Lowe, D. G. (2003). Recognising panoramas. *Proceedings of the IEEE International Conference on Computer Vision, 2*, 1218–1225. <https://doi.org/10.1109/iccv.2003.1238630>
- Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2014). Spectral Networks and Deep Locally Connected Networks on Graphs, *21 May 2014*. 1–14.
- BuildingSMART. (2018). *buildingSMART*.
- Catalina, T., Iordache, V., and Caracaleanu, B. (2013). Multiple regression model for fast prediction of the heating energy demand. *Energy and Buildings, 57*, 302–312.

- Chami, I., Ying, R., Ré, C., and Leskovec, J. (2019). *Hyperbolic Graph Convolutional Neural Networks*. 1, 1–20.
- Cho, Y. K., Alaskar, S., and Bode, T. A. (2010). BIM-integrated sustainable material and renewable energy simulation. *Construction Research Congress 2010: Innovation for Reshaping Construction Practice - Proceedings of the 2010 Construction Research Congress*, 41109(May 2010), 288–297. (373)29
- Chong, A., and Menberg, K. (2018). Guidelines for the Bayesian calibration of building energy models. *Energy and Buildings*, 174, 527–547.
<https://doi.org/10.1016/j.enbuild.2018.06.028>
- Clarke, J. (2015). A vision for building performance simulation: a position paper prepared on behalf of the IBPSA Board. *Journal of Building Performance Simulation*, 8(2), 39–43.
<https://doi.org/10.1080/19401493.2015.1007699>
- Comaniciu, D., and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 603–619. <https://doi.org/10.1109/34.1000236>
- Corzo, G. A., Solomatine, D. P., Hidayat, H., De Wit, M., Werner, M., Uhlenbrook, S., and Price, R. K. (2009). Combining semi-distributed process-based and data-driven models in flow simulation: A case study of the Meuse river basin. *Hydrology and Earth System Sciences*, 13(9), 1619–1634. <https://doi.org/10.5194/hess-13-1619-2009>
- Daniel Maturana and Sebastian Scherer. (2017). VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. *The Positive Encourager*.
<https://doi.org/10.1109/IROS.2015.7353481>
- Darema, F. (2011). DDDAS computational model and environments. *Journal of Algorithms and Computational Technology*, 5(4), 545–560. <https://doi.org/10.1260/1748-3018.5.4.545>

- Davis, L. S. (1975). A survey of edge detection techniques. *Computer Graphics and Image Processing*, 4(3), 248–270. (75)90012-x
- Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). *Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering*. *Nips*.
- Dimitriou, V., Firth, S. K., Hassan, T. M., and Fouchal, F. (2016). BIM enabled building energy modelling: development and verification of a GBXML to IDF conversion method. *Proceedings of the 3rd IBPSA-England Conference BSO*, 1126.
- Dong, B., Lam, K. P., Huang, Y. C., and Dobbs, G. M. (2007). A comparative study of the IFC and gbXML informational infrastructures for data exchange in computational design support environments. *IBPSA 2007 - International Building Performance Simulation Association 2007*, 1530–1537.
- Donnell, J. O., See, R., Rose, C., Maile, T., Bazjanac, V., and Haves, P. (2011). SimModel: A domain data model for whole building energy simulation. *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association*, 382–389.
- Donnell, J. T. O., Maile, T., Rose, C., Morrissey, E., Regnier, C., Parrish, K., and Bazjanac, V. (2013). *Transforming BIM to BEM: Generation of Building Geometry for the NASA Ames Sustainability Base BIM*. January.
- Eastman, C., and Henrion, M. (1977). Glide: A language for design information systems. *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1977*, 24–33. <https://doi.org/10.1145/563858.563863>
- Eastman, C., and Jeng, T. S. (1999). Database supporting evolutionary product model development for design. *Automation in Construction*, 8(3), 305–323. (98)00079-X

- Eastman, C. M. (1979). The representation of design problems and maintenance of their structure. *Annals of Clinical Psychiatry Atomoxetine in African-Americans*, 21(1).
- Enkelmann, W., and Nagel, H. H. (1986). An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(5), 565–593.
<https://doi.org/10.1109/TPAMI.1986.4767833>
- Epstein, B. R., Hingorani, R., Shapiro, J. M., and Czigler, M. (1992). Multispectral KLT-wavelet data compression for Landsat Thematic Mapper images. *Data Compression Conference Proceedings, 1992-March* (May), 200–208. <https://doi.org/10.1109/DCC.1992.227461>
- Erdogmus, N., and Marcel, S. (2013). Spoofing in 2D face recognition with 3D masks and anti-spoofing with Kinect. *IEEE 6th International Conference on Biometrics: Theory, Applications and Systems, BTAS 2013*. <https://doi.org/10.1109/BTAS.2013.6712688>
- Fanelli, G., Weise, T., Gall, J., and Van Gool, L. (2011). Real time head pose estimation from consumer depth cameras. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6835 LNCS, 101–110.
https://doi.org/10.1007/978-3-642-23123-0_11
- Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013). Learning Hierarchical Features for Scene Labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions On*, 35(8), 1915–1929. <https://doi.org/10.1109/TPAMI.2012.231>
- Firman, M. (2016). RGBD Datasets: Past, Present and Future. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Section 3*, 661–673.
<https://doi.org/10.1109/CVPRW.2016.88>

- Freeman, W., Perona, P., and Schölkopf, B. (2008). International Journal of Computer Vision: Guest Editorial. *International Journal of Computer Vision*, 77(1–3), 1.
<https://doi.org/10.1007/s11263-008-0127-7>
- Gauchel et al. (1992). Armilla-Modular building models.
- Geman, S., and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6), 721–741. <https://doi.org/10.1109/TPAMI.1984.4767596>
- Geyer, P., and Schlüter, A. (2014). Automated meta-model generation for Design Space Exploration and decision-making – A novel method supporting performance-oriented building design and retrofitting. *Applied Energy*, 119, 537–556.
<https://doi.org/10.1016/j.apenergy.2013.12.064>
- Goldstein, E. B., and Coco, G. (2015). Machine learning components in deterministic models: Hybrid synergy in the age of data. *Frontiers in Environmental Science*, 3(APR), 1–4.
<https://doi.org/10.3389/fenvs.2015.00033>
- Goodfellow, I., Bengio, Y., and C. (2016). *Deep learning*.
- Gorissen, D., and Dhaene, T. (2010). *A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design*. 11, 2051–2055.
- Gratia, E. (2002). *A simple design tool for the thermal study of an office building*. 34, 279–289.
- Grzybek, H., Gulliver, S., and Huang, Z. (2010). *A Dynamic Intelligent Building Model Based on Industry Foundation Classes*.
- Han, J. M., Chang, C. K., & Malkawi, A. (2020). ARINet: Using 3D convolutional neural networks to estimate annual radiation intensities on building facades. *Building*

Performance Analysis Conference and SimBuild co-organized by ASHRAE and IBPSA-USA, 252-259.

Han, J. M., Choi, E. S., & Malkawi, A. (2021). CoolVox: Advanced 3D convolutional neural network models for predicting solar radiation on building facades. *Building Simulation* (Vol. 15, No. 5, pp. 755-768). Tsinghua University Press.

Han, J. M., Ang, Y. Q., Malkawi, A., & Samuelson, H. W. (2021). Using recurrent neural networks for localized weather prediction with combined use of public airport data and on-site measurements. *Building and Environment*, 192, 107601.

Han, X. F., Jin, J. S., Wang, M. J., and Jiang, W. (2018). Guided 3D point cloud filtering. *Multimedia Tools and Applications*, 77(13), 17397–17411. <https://doi.org/10.1007/s11042-017-5310-9>

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1002/chin.200650130>

Hensen, J., Djunaedy, E., Radošević, M., & Yahiaoui, A. (2004). Building performance simulation for better design: some issues and solutions. In *Proceedings of the 21st Conference on Passive and Low Energy Architecture (PLEA)* (pp. 1185-1190).

Heo, Y., Choudhary, R., and Augenbroe, G. A. (2012). Calibration of building energy models for retrofit analysis under uncertainty. *Energy and Buildings*, 47(2012), 550–560. <https://doi.org/10.1016/j.enbuild.2011.12.029>

Hien, W. N., Poh, L. K., and Feriadi, H. (2000). The use of performance-based simulation tools for building design and evaluation: A Singapore perspective. *Building and Environment*, 35(8), 709–736. (99)00059-1

- Hijazi, M., Kensek, K., and Konis, K. (2015). Bridging the gap: supporting data transparency from BIM to BEM. *Future of Architectural Research Centers Consortium*, 149-.
- Hopfe, C. J., Emmerich, M. T. M., Marijt, R., and Hensen, J. (2012). Robust multi-criteria design optimization in building design. *BSO12 - Building Simulation and Optimization Conference, citation (15)*, 19–26.
- Horowitz, S. L., and Pavlidis, T. (1976). Picture Segmentation by a Tree Traversal Algorithm. *Journal of the ACM (JACM)*, 23(2), 368–388. <https://doi.org/10.1145/321941.321956>
- Hygh, J. S., Decarolis, J. F., Hill, D. B., and Ranjithan, S. R. (2012). Multivariate regression as an energy assessment tool in early building design. *Building and Environment*, 57, 165–175. <https://doi.org/10.1016/j.buildenv.2012.04.021>
- Inventor, A., & Tooling, A. I. (2002). Autodesk®.
- Ioannidou, A., Chatzilari, E., Nikolopoulos, S., and Kompatsiaris, I. (2017). Deep learning advances in computer vision with 3D data: A survey. *ACM Computing Surveys*, 50(2). <https://doi.org/10.1145/3042064>
- Isard, M., and Blake, A. (1998). Condensation-conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1), 5–28.
- Jaffal, I., Inard, C., and Ghiaus, C. (2009). Fast method to predict building heating demand based on the design of experiments. 41, 669–677. <https://doi.org/10.1016/j.enbuild.2009.01.006>
- Jain, A., and Srinivasulu, S. (2004). Development of effective and efficient rainfall-runoff models using integration of deterministic, real-coded genetic algorithms and artificial neural network techniques. 40, 1–12.

- Jeong, W., Kim, J. B., Clayton, M. J., Haberl, J. S., and Yan, W. (2014). Translating building information modeling to building energy modeling using model view definition. *Scientific World Journal*, 2014(1).
- J.Gauchel, L. Hovestadt, S. V. W. and R. R. B. (1992). Modular Building Models. Carnegie Mellon University.
- Kalogirou, S. A., and Bojic, M. (2000). Artificial neural networks for the prediction of the energy consumption of a passive solar building. *Energy*, 25(5), 479–491. (99)00086-9
- Kamnitsas, K., Ledig, C., Newcombe, V. F. J., Simpson, J. P., Kane, A. D., Menon, D. K., Rueckert, D., and Glocker, B. (2017). Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. *Medical Image Analysis*, 36, 61–78.
<https://doi.org/10.1016/j.media.2016.10.004>
- Karlshøj, J., See, R., and Davis, D. (2012). An Integrated Process for Delivering IFC Based Data Exchange. 1, 53.
- Kazmi, I. K., You, L., and Zhang, J. J. (2013). A survey of 2D and 3D shape descriptors. *Proceedings - 10th International Conference Computer Graphics, Imaging, and Visualization, CGIV 2013*, 1–10. <https://doi.org/10.1109/CGIV.2013.11>
- Khemplani, L. (2004). Autodesk Revit: implementation in practice. Autodesk.
- Kim, I., Kim, J., and Seo, J. (2012). Development of an IFC-based IDF converter for supporting energy performance assessment in the early design phase. *Journal of Asian Architecture and Building Engineering*, 11(2), 313–320. <https://doi.org/10.3130/jaabe.11.313>
- Kipf, T. N., and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *ICLR*, 1–14.
- Kiviniemi, A. (1999). IAI and IFC - State of Art. *Cib W78 Conference, September 1995*.

- Kovnatsky, A., Bronstein, M. M., Bronstein, A. M., Glashoff, K., and Kimmel, R. (2013). Coupled quasi-harmonic bases. *32*(2).
- Krasnopolsky, V. M., & Fox-Rabinovitz, M. S. (2006). Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks*, *19*(2), 122-134.
- Krizhevsky, A., Sutskever, I., and Hinton, E. G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 1–1432. <https://doi.org/10.1201/9781420010749>
- Laakso, M., and Kiviniemi, A. (2012). The IFC standard - A review of history, development, and standardization. *Electronic Journal of Information Technology in Construction*, *17*(May 2012), 134–161.
- LadyBugTools. (2018). *gbXML Viewer R12*.
<https://doi.org/https://www.ladybug.tools/spider/gbxml-viewer/r12/gv-app/gv-app.html>.
- Lam, K. P., Karaguzel, O. T., Zhang, R., and Zhao, J. (2012). Identification and Analysis of Interoperability Gaps between Nbims/Open Standards and Building Performance Simulation Tools. *Greater Philadelphia Innovation Cluster for Energy-Efficient Buildings*, *February*, 25.
- Lam, K. P., Wong, N. H., Mahdavi, A., Chan, K. K., Kang, Z., and Gupta, S. (2004). SEMPER-II: An internet-based multi-domain building performance simulation environment for early design support. *Automation in Construction*, *13*(5 SPEC. ISS.), 651–663.
<https://doi.org/10.1016/j.autcon.2003.12.003>
- Lee, G., Sacks, R., and Eastman, C. M. (2006). Specifying parametric building object behavior (BOB) for a building information modeling system. *Automation in Construction*, *15*(6), 758–776. <https://doi.org/10.1016/j.autcon.2005.09.009>

- Leibe, B., Matas, J., Sebe, N., and Welling, M. (2016). Deep Learning 3D Shape Surfaces Using Geometry Images. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9906 LNCS, VII–IX.
<https://doi.org/10.1007/978-3-319-46466-4>
- Li, Y., Zemel, R., Brockschmidt, M., and Tarlow, D. (2016). Gated Graph Sequence Neural Networks. *1*, 1–20.
- Liebich, T. (2010). Unveiling IFC2x4 - the next generation of OpenBIM. *Proceedings of the 27th International Conference on Information Technology in Construction CIB W78*, 16–18.
- Liesje Van Gelder, Payel Das, Hans Janssen, S. R. (2014). *Comparative study of metamodeling techniques in building energy simulation: Guidelines for practitioners*.
- Liu, X., Wang, X., Wright, G., Cheng, J. C. P., Li, X., and Liu, R. (2017). A state-of-the-art review on the integration of Building Information Modeling (BIM) and Geographic Information System (GIS). *ISPRS International Journal of Geo-Information*, 6(2), 1–21.
<https://doi.org/10.3390/ijgi6020053>
- Longuet-Higgins, H. C. (1987). A computer algorithm for reconstructing a scene from two projections. In *Readings in Computer Vision* (pp. 61–62). <https://doi.org/10.1016/b978-0-08-051581-6.50012-x>
- Macumber, D., Horowitz, S., Schott, M., Nolan, K., Schiller, B., Horowitz, S., Schott, M., Nolan, K., and Schiller, B. (2018). FloorspaceJS - A New, Open Source, Web-Based Geometry Editor for Building Energy Modeling (BEM). *Simulation for Architecture and Urban Design (SimAUD) 2018, March*, 1–8.
- Mahdavi, A., Mathew, P., Kumar, S., and Wong, N. H. (1997). Bi-directional computational design support in the SEMPER environment. *Automation in Construction*, 6(4), 353–373.

- Malkawi, A., and Augenbroe, G. (2004). *Advanced building simulation*. Routledge.
- Malkawi, A. M. (2004). Developments in environmental performance simulation. *Automation in Construction*, 13(4), 437–445. <https://doi.org/10.1016/j.autcon.2004.03.002>
- Mallat, S. G. (1989). A theory for multiresolution signal decomposition: The wavelet representation. *Fundamental Papers in Wavelet Theory, I* (7), 494–513.
<https://doi.org/10.1515/9781400827268.494>
- Mann, S., and Picard, R. W. (1996). Video Orbits of the Projective Group: A New Perspective on Image Mosaicing. *MIT Technical Report*, 338, 1–31.
- Masci, J., Boscaini, D., Bronstein, M. M., and Vandergheynst, P. (2018). Geodesic convolutional neural networks on Riemannian manifolds. *ArXiv:1501.06297v3*.
- Miller. (1968). *Response Time in Man-Computer Conversational Transactions*. 33(pt. 1), 267–277. <https://doi.org/10.1145/1476589.1476628>
- Mingqiang, Y., Kidiyo, K., and Joseph, R. (2008). A Survey of Shape Feature Extraction Techniques. *Pattern Recognition Techniques, Technology and Applications, November*.
<https://doi.org/10.5772/6237>
- Mohd-Nor, M. F. I., and Grant, M. P. (2014). Building information modelling (BIM) in the Malaysian architecture industry. *WSEAS Transactions on Environment and Development*, 10, 264–273. <https://doi.org/10.1016/j.buildenv.2006.10.027>
- Monti, F., Boscaini, D., Masci, J., Rodol, E., Svoboda, J., and Bronstein, M. M. (2016). Geometric deep learning on graphs and manifolds using mixture model CNNs.
- Moon, H. J., Kim, B. K., and Choi, M. S. (2013). A BIM based data model for an integrated building energy information management in the design and operational stages.

Proceedings of BS 2013: 13th Conference of the International Building Performance Simulation Association, 3217–3224.

Mumford, D., and Shah, J. (1989). Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5), 577–685. <https://doi.org/10.1002/cpa.3160420503>

New, J., Ph., D., and Adams, M. B. (2018). *EnergyPlus Performance Improvements via JSON Input Refactoring*. *Yip 2016*, 320–324.

New, J. R., Ridge, O., and Parker, L. (2017). Constructing Large Scale Surrogate Models from Big Data and Artificial Intelligence Constructing Large Scale Surrogate Models from Big Data and Artificial Intelligence. <https://doi.org/10.1016/j.apenergy.2017.05.155>

Nguyen, A. T., Reiter, S., and Rigo, P. (2014). A review on simulation-based optimization methods applied to building performance analysis. *Applied Energy*, 113, 1043–1058. <https://doi.org/10.1016/j.apenergy.2013.08.061>

Nicolle, C., and Cruz, C. (2011). Semantic Building Information Model and multimedia for facility management. *Lecture Notes in Business Information Processing*, 75 LNBIP, 14–29. https://doi.org/10.1007/978-3-642-22810-0_2

Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. *Proceedings of the IEEE International Conference on Computer Vision, 2015 Inter*, 1520–1528. <https://doi.org/10.1109/ICCV.2015.178>

Nouidui, T. S., Wetter, M., and Berkeley, L. (2018). SimulatorToFMU: A Python Utility to Support Building Simulation Tool Interoperability. *In Proceedings of the 2018 Building Performance Analysis Conference and SimBuild, Chicago, IL*. 325–330.

- Papamichael, K., LaPorta, J., and Chauvet, H. (1997). Building Design Advisor: Automated integration of multiple simulation tools. *Automation in Construction*, 6(4), 341–352. (97)00043-5
- Paul Debevec, J. M. (1998). Recovering High Dynamic Range Radiance Maps from Photographs Paul E. Debevec. *ACM Transactions on Graphics*.
- Pauwels, P., Krijnen, T., Terkaj, W., and Beetz, J. (2017). Automation in Construction Enhancing the ifcOWL ontology with an alternative representation for geometric data. *Automation in Construction*, 80, 77–94. <https://doi.org/10.1016/j.autcon.2017.03.001>
- Pauwels, P., and Terkaj, W. (2016). EXPRESS to OWL for construction industry : Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, 63, 100–133. <https://doi.org/10.1016/j.autcon.2015.12.003>
- Pavlidis, T., and Liow, Y. T. (1990). Integrating Region Growing and Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3), 225–233. <https://doi.org/10.1109/34.49050>
- Penttila, H., Rajala, M., and Freese, S. (2007). Building Information Modelling of Modern Historic Buildings. *Predicting the Future, 25th ECAADe Konferansı, Frankfurt Am Main, Germany*, 607–614.
- Poggio, T., Torre, V., and Koch, C. (1985). Computational vision and regularization theory. *Nature*, 317(6035), 314–319. <https://doi.org/10.1038/317314a0>
- Pohl, J., LaPorta, J., Pohl, K. J., and Snyder, J. (1992). AEDOT Prototype (1.1): An Implementation of the ICADS Model.

- Ponce, J., Berg, T. L., Everingham, M., Forsyth, D. A., Hebert, M., Lazebnik, S., Marszalek, M., Schmid, C., Russell, B. C., Torralba, A., Williams, C. K. I., Zhang, J., and Zisserman, A. (2006). Dataset Issues in Object Recognition. 29–48. https://doi.org/10.1007/11957959_2
- Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 77–85. <https://doi.org/10.1109/CVPR.2017.16>
- Rackes, A., Paula, A., and Lamberts, R. (2016). Naturally comfortable and sustainable : Informed design guidance and performance labeling for passive commercial buildings in hot climates. *Applied Energy*, 174, 256–274. <https://doi.org/10.1016/j.apenergy.2016.04.081>
- Ravi Srinivasan, Charles Kibert, Paul Fishwick, Zachary Ezzell, Siddharth Thakur, Ishfak Ahmed, J. L. S. (2016). *Preliminary research in dynamic-BIM (D-BIM) workbench development. In Proceedings of the 2012 Winter Simulation Conference (WSC)*, 1–12. <https://doi.org/10.1109/WSC.2012.6465271>
- Ritter, F. (2015). Simulation-based Decision-making in Early Design Stages.
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 9351, pp. 234–241). https://doi.org/10.1007/978-3-319-24574-4_28
- Rosenfeld, A., and Davis, L. S. (1979). Image segmentation and image models. *Proceedings of the IEEE*, 67(5), 764–772. <https://doi.org/10.1109/PROC.1979.11326>

- Roudsari, M. S., and Pak, M. (2013). Ladybug: A parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design. *Proceedings of BS 2013: 13th Conference of the International Building Performance Simulation Association, December*, 3128–3135.
- Rutherford, J. H. (1993). KNODES : Knowledge-Based Design Decision Support. *Design*.
- Samuelson, H., Claussnitzer, S., Goyal, A., Chen, Y., and Romo-Castillo, A. (2016). Parametric energy simulation in early design: High-rise residential buildings in urban contexts. *Building and Environment*, 101, 19–31. <https://doi.org/10.1016/j.buildenv.2016.02.018>
- Scarselli, F., Gori, M., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *20(1)*, 61–80.
- Sedaghat, N., Zolfaghari, M., Amiri, E., and Brox, T. (2017). Orientation-boosted Voxel nets for 3D object recognition. *British Machine Vision Conference 2017, BMVC 2017*, 1–18. <https://doi.org/10.5244/c.31.97>
- Seitz, S. M., and Dyer, C. R. (1999). Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2), 151–173. <https://doi.org/10.1023/A:1008176507526>
- Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2014). Overfeat: Integrated recognition, localization and detection using convolutional networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.
- Shafiei Dizaji, M., and Harris, D. (2019). *3D InspectionNet: a deep 3D convolutional neural networks based approach for 3D defect detection on concrete columns*. April 2019, 13. <https://doi.org/10.1117/12.2514387>

- Sharma, A., Grau, O., and Fritz, M. (2016). VConv-DAE: Deep volumetric shape learning without object labels. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9915 LNCS, 236–250.
https://doi.org/10.1007/978-3-319-49409-8_20
- Shelden, D. (2009). Information modelling as a paradigm shift. *Architectural Design*, 79(2), 80–83. <https://doi.org/10.1002/ad.857>
- Shelhamer, E., Long, J., and Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 640–651. <https://doi.org/10.1109/TPAMI.2016.2572683>
- Shi, B., Bai, S., Zhou, Z., and Bai, X. (2015). DeepPano: Deep Panoramic Representation for 3-D Shape Recognition. *IEEE Signal Processing Letters*, 22(12), 2339–2343.
<https://doi.org/10.1109/LSP.2015.2480802>
- Shi, J., and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888–905.
<https://doi.org/10.1109/34.868688>
- Singaravel, S., Suykens, J., and Geyer, P. (2018a). Advanced Engineering Informatics Deep-learning neural-network architectures and methods : Using component-based models in building-design energy prediction. *Advanced Engineering Informatics*, 38(June), 81–90.
<https://doi.org/10.1016/j.aei.2018.06.004>
- Singaravel, S., Suykens, J., and Geyer, P. (2018b). Deep-learning neural-network architectures and methods: Using component-based models in building-design energy prediction. *Advanced Engineering Informatics*, 38(May), 81–90.
<https://doi.org/10.1016/j.aei.2018.06.004>

- Snavely, N., Seitz, M. S., and Szeliski, R. (2006). Photo Tourism: Exploring image collections in 3D. *Proceedings of SIGGRAPH 2006*, 1(212), 835–846.
- Srinivasan, P., Liang, P., and Hackwood, S. (1990). Computational geometric methods in volumetric intersection for 3D reconstruction. *Pattern Recognition*, 23(8), 843–857. (90)90131-4
- Srinivasan, R., Kibert, C., Thakur, S., Ahmed, I., Fishwick, P., Ezzell, Z., and Lakshmanan, J. (2012). Preliminary research in dynamic-BIM (D-BIM) Workbench development. *Proceedings - Winter Simulation Conference, BuildingSMART*, 594–605. <https://doi.org/10.1109/WSC.2012.6465331>
- Stavarakakis, G. M., Zervas, P. L., Sarimveis, H., and Markatos, N. C. (2012). Optimization of window-openings design for thermal comfort in naturally ventilated buildings. *Applied Mathematical Modelling*, 36(1), 193–211. <https://doi.org/10.1016/j.apm.2011.05.052>
- Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3D shape recognition. *Proceedings of the IEEE International Conference on Computer Vision, 2015 Inter*, 945–953. <https://doi.org/10.1109/ICCV.2015.114>
- Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.
- Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., and Rother, C. (2008). A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6), 1068–1080. <https://doi.org/10.1109/TPAMI.2007.70844>
- Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2017). Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs. *Proceedings of the IEEE*

International Conference on Computer Vision, 2017-October, 2107–2115.

<https://doi.org/10.1109/ICCV.2017.230>

Terzopoulos, D. (1983). Multilevel computational processes for visual surface reconstruction.

Computer Vision, Graphics, and Image Processing, 24(1), 52–96. (83)90020-8

Terzopoulos, D. (1988). The Computation of Visible-Surface Representations. *10(4)*.

Turk, M., and Pentland, A. (1991). Eigen faces for Recognition. *Journal of Cognitive*

Neuroscience, 3(1).

Volk, R., Stengel, J., and Schultmann, F. (2014). Building Information Modeling (BIM) for

existing buildings - Literature review and future needs. *Automation in Construction, 38,*

109–127. <https://doi.org/10.1016/j.autcon.2013.10.023>

Wang, C., Samari, B., and Siddiqi, K. (2018). Local Spectral Graph Convolution for Point Set

Feature Learning. *ArXiv Preprint ArXiv:1803.05827*.

Wang, G. G., and Shan, S. (2007). Review of metamodeling techniques in support of engineering

design optimization. *Journal of Mechanical Design, Transactions of the ASME, 129(4),*

370–380. <https://doi.org/10.1115/1.2429697>

Wang, J., and Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification

using Deep Learning.

Wang, P. S., Liu, Y., Guo, Y. X., Sun, C. Y., and Tong, X. (2017). O-CNN: Octree-based

convolutional neural networks for 3D shape analysis. *ACM Transactions on Graphics,*

36(4). <https://doi.org/10.1145/3072959.3073608>

Wang, S., Zhang, G., Shen, B., and Xie, X. (2011). An integrated scheme for Cyber-physical

building energy management system. *Procedia Engineering, 15(0), 3616–3620.*

<https://doi.org/10.1016/j.proeng.2011.08.677>

- Wei, T. (2013). A review of sensitivity analysis methods in building energy analysis. *Renewable and Sustainable Energy Reviews*, 20, 411–419. <https://doi.org/10.1016/j.rser.2012.12.014>
- Westermann, P., and Evins, R. (2019). Surrogate modelling for sustainable building design – A review. *Energy and Buildings*, 198, 170–186.
<https://doi.org/10.1016/j.enbuild.2019.05.057>
- Wetter, M. (2011). Co-simulation of building energy and control systems with the building controls virtual test bed. *Journal of Building Performance Simulation*, 4(3), 185–203.
<https://doi.org/10.1080/19401493.2010.518631>
- Witkin, A. P. (1984). *Scale-space processing*. 1–4.
- Wong, J., and Yang, J. (2010). Research and application of Building Information Modelling (BIM) in the Architecture, Engineering and Construction (AEC) industry: a review and direction for future research. *Proceedings of the 6th International Conference on Innovation in Architecture, Engineering and Construction (AEC)*, Loughborough University, UK, Pennsylvania State University, 356–365.
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Member, S., and Yu, P. S. (2019). A Comprehensive Survey on Graph Neural Networks. 1–22.
- Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3D ShapeNets: A deep representation for volumetric shapes. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June, 1912–1920.
<https://doi.org/10.1109/CVPR.2015.7298801>
- Xiang, Y., Choi, W., Lin, Y., and Savarese, S. (2015). Data-driven 3D Voxel Patterns for object category recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June, 1903–1911.
<https://doi.org/10.1109/CVPR.2015.7298800>

- Yi, Y. K., Lee, J., Malkawi, A. M., Wang, C., Jiang, Y., and Yan, D. (2007). Data-centric simulation interoperability using the Dest simulation platform. *IBPSA 2007 - International Building Performance Simulation Association 2007*, 1587–1594.
- Zarzalejo, L. F., Ramirez, L., and Polo, J. (2005). Artificial intelligence techniques applied to hourly global irradiance estimation from satellite-derived cloud index. *Energy*, 30(9 SPEC. ISS.), 1685–1697. <https://doi.org/10.1016/j.energy.2004.04.047>
- Zhang, L., João, M., and Ferreira, A. (2004). Survey on 3D shape descriptors. *DecorAR*, 1–28. <https://doi.org/10.1016/j.ultsonch.2010.03.006>
- Zhang, Z., Chong, A., Poh Lam, K., Pan, Y., Zhang, C., and Lu, S. (2018). *A Deep Reinforcement Learning Approach to Using Whole Building Energy Model For HVAC Optimal Control ASHRAE Multidisciplinary Task Group on Occupant Behavior in Buildings View project International Energy Agency Energy in Buildings and Communities Program. July.*
- Zhang, Z., Cui, P., and Zhu, W. (2020). Deep Learning on Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 14(8), 1–1. <https://doi.org/10.1109/tkde.2020.2981333>
- Zhao, J., Xie, X., Xu, X., and Sun, S. (2017). Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38, 43–54. <https://doi.org/10.1016/j.inffus.2017.02.007>
- Zhi, S., Liu, Y., Li, X., and Guo, Y. (2017). Toward real-time 3D object recognition: A lightweight volumetric CNN framework using multitask learning. *Computers and Graphics*, 71, 199–207.
- Zollhöfer, M., Nießner, M., Izadi, S., Rehmann, C., Zach, C., Fisher, M., ... & Stamminger, M. (2014). Real-time non-rigid reconstruction using an RGB-D camera. *ACM Transactions on Graphics (ToG)*, 33(4), 1-12.