Faculty of Science and Engineering

Reducing Computational Fluid Dynamics Simulation Cost by Intelligently Reducing Geometric Complexity

by

Stuart Ross Porter

This thesis is presented for the Degree of
Doctor of Philosophy - Chemical Engineering
of
Curtin University

November 2020

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made. This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Many of the Figures in this thesis have undergone minor editing to remove white space, increase visual contrast, or highlight important features. These edits were performed to improve the visibility of relevant information, and no data of significance was deliberately removed or obfuscated in the process.

Abstract

Computational Fluid dynamics (CFD) allows simulation of fluid flows including those relating to emergency scenarios. It also permits analysis of a wide range of emergency scenarios, with the ability to simulate them realistically from a scale of mm to metres/kilometres. Scenarios that can be simulated with CFD include chemical spills, personnel evacuations, hazardous gas clouds, dangerous weather, uncontrolled fires, and explosions.

But conducting large-scale CFD simulations for emergency scenarios is not only computationally expensive but also challenging because of lack of integration between Computer Aided Design/Building Information Model (CAD/BIM) and CFD. (BIM) or (CAD) geometries that are suitable for design and visualization often require extensive rework before they are usable in a CFD simulation, and sometimes must be rebuilt entirely. Once the geometry is CFD ready, the process can still require a great deal of expert knowledge and computational time to produce useful results.

This thesis examines some of the current challenges in using existing BIM/CAD geometry for CFD simulations. It identifies the manual rework of models as a key limiting factor in the wider use of CFD simulations. To overcome this limitation, a semi-autonomous algorithm to quickly produce fit-for-purpose CFD ready geometry (or abstractions) has been developed. CFD simulations are then carried out on these "abstractions", and a comparative analysis of the results with reference to the original geometry has been conducted. In summary, this research paves the way for integration of BIM/CAD with CFD.

Authorship

The following paper has already been published based on this research:

Porter, S., Tan, T., Wang, X., & Pareek, V. (2018). **LODOS - Going from BIM to CFD via CAD and model abstraction.** Automation in Construction, 94(January), 85–92. https://doi.org/10.1016/j.autcon.2018.06.001

Statement on Attribution of Contribution

The following parties have contributed to this research, and are satisfied this work duly acknowledges their contributions:

Stuart Porter, Candidate

Vishnu Pareek, Primary Supervisor

Tele Tan, Co-Supervisor

Xiangyu Wang, Contributor

Acknowledgments

Thank you to the Australasian Joint Research Centre for Building Information Modelling and Woodside Energy for the use of the LNG Module model. Thank you to Dr Jingde Li for the use of the LNG Train model.

Thank you to Curtin University, the West Australian School of Mines, and my supervisors for their support.

Thank you to Dr Biao Sun for his assistance and patience, this work would of been a great deal harder and taken a great deal longer without his help. I will be eternally grateful.

Thank you to Dr Richard Palmer for providing a sounding board, and an extra pair of eyes when my own were weary from proof reading.

Thank you most of all to my friends and family, who were there for me when I needed them to be, not there when I needed space, and understanding of the many times I couldn't be there due to my study commitments. This has been a longer and rougher road than anyone expected, but I'm proud of the work I've done, and looking forward to spending more time with all of you on the road ahead.

And thank you, whoever you are, for taking the time to read this. I hope you find my work interesting, accessible, and in some small way helpful in your own.

This research was supported in-part by an Australian Postgraduate Award (APA) Scholarship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computational Fluid Dynamics (CFD) is a technology that offers amazing versatility. Practitioners can create highly accurate fluid-flow simulations, which among other things are capable of predicting how a fire will progress, the parts in an engine will wear, and even how a virus might spread. CFD Simulations have the ability to aid in the design of nearly any manufactured object or structure, as well as to help with the understanding of complex natural systems, but they are also incredibly resource intensive to perform.

The process of setting up a simulation, from creating or importing the geometry, to producing a high quality mesh, to choosing how and what to simulate, can take hours or days of person time before the actual simulation process even begins. Once the simulation is ready, even simulating a relatively small area, such as a few metres cubed with a fluid flowing around a few objects, can take several hours of processing time to simulate just a few seconds of actual fluid flow. This means detailed simulations of a large-scale complex geometry that might be required to examine the safety of certain buildings or facilities, could be prohibitively expensive to perform.

Moore's Law[1] suggests that processors double in complexity approximately every 18 months, with a roughly commensurate increase in computational power. Unfortunately the scale of computation required by large CFD problems means that even if Moore's law continues to hold for the next few decades, large scale problems featuring small-scale features will continue to be prohibitively expensive to process, with gains in hardware capability likely to be used for improving simulation complexity as much as for reducing processing times.

[1]https://ieeexplore.ieee.org/abstract/document/591665

Given the potential of CFD simulations to assist in so many aspects of everyday life, finding ways to reduce the cost of moving from geometry to finished simulation has massive potential to affect both current use cases as well as open up new ones. At the time of writing, the world is dealing with a pandemic in the form of COVID-19 (Ghebreyesus, 2020), with some researchers already using CFD to examine the dispersion of droplets or aerosols from a sneeze in a grocery store (Vuorinen *et al.*, 2020). If CFD can be made more accessible in terms of resource cost, it would enable wider uses as well as more dynamic responses from researchers to current and future problems.

This thesis will examine the process of moving from existing geometry in a BIM format to a CFD-ready mesh. It will discuss challenges in the current model conversion, import, rework, and mesh process, current methods for addressing those challenges, as well as potential new methods. It will detail the development and implementation of the proposed solutions, before presenting case studies with an analysis of their effectiveness.

1.1 Motivation

In Australia, safety is of paramount importance, where all operations are carefully governed by the laws of the country and the ethics of the companies involved – none more so than the country's Mining and Energy (M&E) industry. Any accident in M&E facilities can be incredibly expensive, where any unscheduled shut down of even a few hours could potentially lead to millions of dollars in lost revenue.

The equipment and substances involved in M&E can be extremely dangerous to human life, posing risks to the workforce, and in some rare circumstances even civilians. These factors lead to a strong focus on safety, redundancy and threat mitigation within the M&E industry. In spite of the inherent dangers, this focus in the industry on keeping personnel and assets safe, with oversight from Government bodies, ensures that significant incidents occur rarely.

But ensuring the safety of a large site, particularly ones dealing with volatile compounds like LNG facilities, can be a costly and difficult process. To verify the safety of facilities, companies must submit their designs to rigorous examination and simulation to ensure they do not pose a public risk. This process is typically undertaken by using CFD to simulate possible emergency scenarios and examine how far the LNG vapour cloud is capable of getting before dissipating to safe levels (Vílchez *et al.*, 2014).

The complexity and thus cost of a simulation varies greatly depending on a number of factors, such as to what degree aspects of physics such as fluid turbulence or energy transfer are modelled. The largest factor in determining a simulations complexity however is the size and complexity of the mesh, determined by dividing the overall area into cells between a given minimum and maximum size. A mesh with a few hundred thousand nodes is considered quite small but may take an hour of real world computing time to simulate a few seconds, with the computational costs increasing greatly for meshes with millions of nodes or more.

With LNG facility footprints sometimes representing hundreds of metres a side, the computational cost to examine an area that may measure thousands of cubic metres is immense, possibly requiring billions of nodes depending on the simulation resolution. The high resource cost of performing this work limits the number of scenarios a company is willing to examine, or the facility designs they can justify performing CFD analysis on. The complex geometry of an LNG plant, containing hundreds of pipes along with various structures and machinery scattered over large areas, makes it particularly costly for CFD analysis.

Engineering models in a Computer Aided Design (CAD) format or BIM format require significant manual rework before they can be used in a CFD simulation, as small gaps or complicated geometry can prevent meshing or require a much higher density mesh. There is also a lack of support for importing BIMs into CFD software. Therefore, it is necessary to explore ways to quickly and cheaply make an existing BIM model CFD compatible without extensive rework, while also examining and documenting the process of moving from a BIM or CAD model to a full CFD simulation.

1.2 Aims

The overall aim of this study was to examine and where possible improve the process of going from existing complex geometry to CFD Simulation results, with specific objectives to:

Research
: Identify challenges in the current methods for moving from existent models to CFD Simulations.

Design
: Study the problems identified to understand their cause, any existing processes for dealing with them, and new ways they might be addressed. Develop a solution or set of solutions based on this study, with an emphasis on extending and improving existing workflows.

Prototype
: A proof of concept system that partially or fully implements the designed solution(s) will need to be built. The implementation needs to be sufficient to allow for the evaluation of the proposed solution(s).

Test and Evaluate
: Finally, a set of simulation experiments need to be designed and carried out to examine the impact of the designed solution(s). Data from these experiments will need to be collected, analysed, and then presented as a thesis.

1.3 Problem Description and Scope

Building Information Models (BIMs) and Computer Aided Designs (CADs) are industry standard methods of representing 3D designs. Despite its name, BIM can be used in the design of anything from a wristwatch to an entire city. When compared to the more traditional CAD which represents designs with lines and 3D geometry, BIM represents its structures as 3D objects linked to a database which records details such as material type, manufacturer, or any other detail considered relevant.

These attributes make BIM a popular alternative to 3D CAD, the data-rich format allowing additional functionality and value adds. Being the industry standard, CAD is more widely supported, though the proprietary file formats implemented by CAD software companies often introduce challenges when moving between systems. BIM being effectively a successor format to CAD means there is strong interoperability between it and CAD, making the process of moving between them typically straightforward and robust.

One of the objectives of this study is to examine how these technologies integrate with CFD packages, which will be examined in part by using a BIM representation of an LNG module. A single LNG module provides an excellent test case for this, as the initial geometric complexity requires a fine mesh to capture accurately, but the domain volume required to simulate it results in an overall mesh that is computationally costly without being prohibitively so. Studying this test case allows the evaluation of current challenges in conducting CFD simulations on existent BIM/CAD geometry, which will be complimented by a literature review looking for known challenges and existing solutions.

One known challenge in conducting CFD simulations on existing architectural geometries that this work seeks to address is their multi-scale nature, where there can be features as small as millimetres and as large as kilometres in a single model. Performing CFD simulations against such models is inconceivable due to the enormous computational requirements, with the entire domain needing to be assessed based on its smallest features. In order to simulate such models, features must be removed, simplified, or even rebuilt in order to make a CFD simulation feasible. Currently, there is no tool able to automatically undertake this process, and doing it manually can require thousands of hours and valuable human resources.

A solution to this challenge will be especially useful for CFD simulations of large and complex systems, such as industrial facilities or a cityscape. Automating rework that would otherwise require manual effort will also reduce the cost to perform simulations, thus making it feasible to conduct CFD simulations during the early design stages where simulation could most inform decisions but the cost of a detailed simulation cannot be justified. A secondary application is in preparing models with peculiar geometry that has been stored in a proprietary format, for transfer to other software packages which lack support.

Finally, a set of suitable simulation experiments to examine the impact of the solution(s) will need to be devised and carried out. These experiments will be performed, comparative data analysed, then presented along with analysis.

1.4 Thesis Scope

The work presented in this thesis is of a multi-disciplinary nature, involving BIM – a tool used by artists and architects, and also CFD – a tool used by engineers and scientists. Therefore, a summary of the state-of-the-art of both BIM and CFD is presented to allow readers familiar with only one discipline to understand the potential of this work in its full context. The literature review pays particular interest to articles covering difficulties in moving from BIM/CAD to CFD, especially where solutions are offered, with articles covering best practices or particularly novel uses of one or more of the technologies also discussed.

Details on a proposed solution to existing difficulties with BIM/CAD to CFD are presented, along with a proof of concept system. In order to leverage existing software and reduce development overheads, this system is implemented within an existing architectural software package. A broad overview of how the system was implemented, along with lessons learned during implementation, is given.

Two test cases are presented, one simple and one that is reasonably complex, each designed to quantify the benefits and costs of the proposed solution. Details of the setup and results of each are presented separately, along with analysis of the solution based on the resulting data. The thesis concludes with a discussion and evaluation of the research to date, including proposals for possible future work.

1.5 Thesis Structure

This thesis is structured as follows:
Chapter 2 provides background on the subjects of CFD and BIM, as well as conducting a critical literature review. Because of the multi-disciplinary nature of this research, a brief background on each discipline is provided to assist those with knowledge of only one discipline to better understand the other. The aim is to provide an understanding of both the challenges and opportunities presented by BIM, CFD, and their integration.

Chapter 3 details the development of the proof of concept solution, the purpose of each algorithm, a high level description of how each algorithm works, and how they are implemented.

Chapter 4 describes a simple test case, demonstrating the implemented solution. Results from the test case are presented, quantifying the impact of Geometric Abstraction on CFD Simulation.

Chapter 5 looks at a more complex test case, derived from a real world LNG Module. Some minor additions to the solution and experimental design are necessary to accommodate the greater complexity of this model, which are detailed in the chapter.

Finally, Chapter 6 presents a summary of the research to date, along with analyses and conclusions drawn from this initial work. Recommendations for future work are provided, highlighting areas where the existing work can be usefully expanded upon.

Chapter 2

Background: BIM and CFD

The research in this thesis is cross-disciplinary in nature, bringing together aspects from Built Environment and Chemical Engineering. To make the research broadly accessible and provide context for its potential, the literature review below will include items that provide background on the respective fields of study as well as examples of interesting applications of domain knowledge or technology. Building Information Modelling (BIM) from Built Environment will be described first, followed by Computational Fluid Dynamics (CFD) from Chemical Engineering, before a review of existing works that bridge the two domains.

2.1 Building Information Modelling

Building Information Modelling is a newer alternative to Computer Aided Design (CAD), which has been gaining prominence in the Architecture Engineering and Construction industry. CAD, which typically means 3D CAD, allows for design of anything from simple personal objects like rings up to complex machinery, buildings and cities. These designs use collections of lines and 3D objects to provide an accurate 3D geometry suitable for fabrication, or to provide an understanding of an object or space. As Cheng and Wang (2010) discuss, this process is inherently limited as the objects used in CAD have no real association with each other besides their position in 3D space, meaning the user must be relied upon to ensure any changes in the model's composition do not have unintended consequences for the rest of the model.

For large and complex CAD projects, it is easy for a physical conflict to be introduced into a model which can go unnoticed until the build crew are attempting to implement that design. This can lead to costly delays or ad hoc solutions leading to further design adjustments to accommodate the implemented changes. BIM improves on CAD in this regard by not simply representing objects as collections of lines, but by using a database back-end to make each 3D object an entity with known properties.

The concept of BIM began in the 1970s (Eastman *et al.*, 1974). It was originally known as Building Description System (BDS) and was intended to consolidate information in one place, so that changes in one area are reflected in other associated areas, keeping all documentation up to date. Since that initial inception, the concept has matured significantly and is gaining popularity in the Architecture, Engineering and Construction (AEC) industry with increased uptake (Bernstein *et al.*, 2014).

BIM has demonstrated extensive Returns On Investment (ROI) for projects of moderate to large scale (Azhar *et al.*, 2008), as can be seen in Figure 2.1. The ROI for BIM utilisation reported by Azhar *et al.* (2008) is exceptional, but the quality of their work suggests the numbers are likely to be accurate, though other projects might be expected to see more modest but still valuable returns. These returns can be attributed to the benefits of BIM, which can provide automatic clash detection, material quantity calculation and quicker design (Gao and Fischer, 2008; Cheng and Wang, 2010). Because of the database association providing details on the objects within the BIM, it also has the potential to be extended in significant ways (Porter *et al.*, 2014), providing many benefits and value adds for its users. For example, the material information stored in the BIM can be used to work out sound and heat insulation, environmental impacts, or even physical security vulnerability.

Due to these advantages, the usage of BIM is likely to continue to grow in comparison to traditional CAD methods. McGraw Hill Construction found that BIM use amongst respondents increased from 36% in 2013 to 56% in 2015, with high level implementations increasing over the same period from 12% of users up to 31% (Bernstein *et al.*, 2014). The increasing versatility and popularity of BIM makes it a logical choice as a format for complex models, and it is likely to become the dominant modelling system in industry.

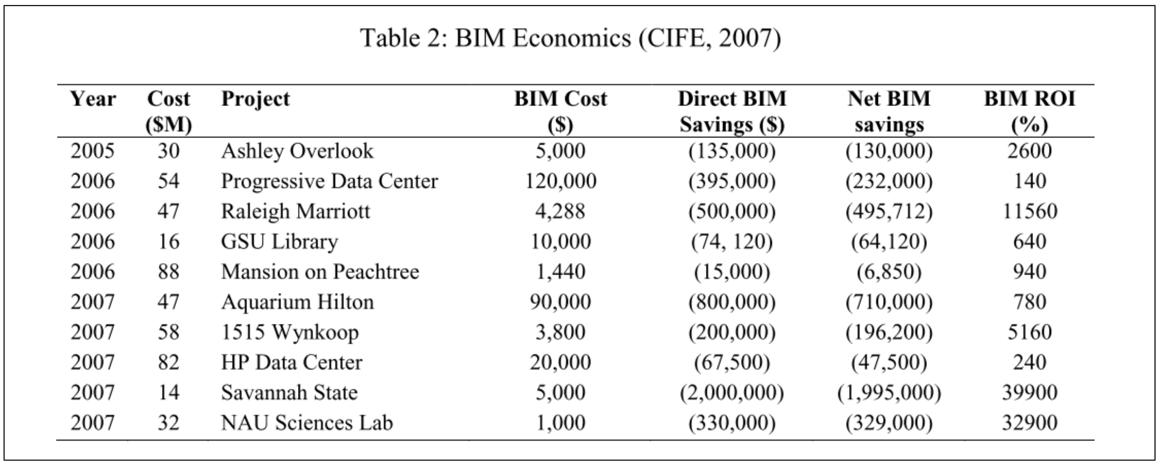


Table 2: BIM Economics (CIFE, 2007)

Year	Cost ($M)	Project	BIM Cost ($)	Direct BIM Savings ($)	Net BIM savings	BIM ROI (%)
2005	30	Ashley Overlook	5,000	(135,000)	(130,000)	2600
2006	54	Progressive Data Center	120,000	(395,000)	(232,000)	140
2006	47	Raleigh Marriott	4,288	(500,000)	(495,712)	11560
2006	16	GSU Library	10,000	(74, 120)	(64,120)	640
2006	88	Mansion on Peachtree	1,440	(15,000)	(6,850)	940
2007	47	Aquarium Hilton	90,000	(800,000)	(710,000)	780
2007	58	1515 Wynkoop	3,800	(200,000)	(196,200)	5160
2007	82	HP Data Center	20,000	(67,500)	(47,500)	240
2007	14	Savannah State	5,000	(2,000,000)	(1,995,000)	39900
2007	32	NAU Sciences Lab	1,000	(330,000)	(329,000)	32900

Figure 2.1: Table 2 from Azhar *et al.* (2008), showing ROI rates for investment in BIM

When integrating BIM with CFD, there are three primary factors: the time to model elements, the time to setup the simulation, and the computational requirements of the simulation. This study presents a process for importing BIM models into CFD. Specifically it details a process for moving from a BIM in Autodesk® Revit® to CFD Simulation in Ansys® Fluent®.

2.2 Computational Fluid Dynamics

Computational Fluid Dynamics refers to the study of fluid flows numerically. Simply, it is the computer aided simulation of fluids using mathematical models to approximate physical forces. Fluids in this context applies to anything which can flow, from water flowing in a pipe, to gasses mixing and combusting in an engine, to even large crowds.

Versteeg and Malalasekera (2007) provide an introduction to the topic of CFD, with an extensive discussion of available methods and the equations behind them. Their work provides an excellent reference for both new and experienced CFD practitioners, explaining how different CFD methods work and discussion on when they are best applied. Their chapters on “Errors and uncertainty in CFD modelling” and “Methods for dealing with complex geometries” are especially relevant to this research, with the provided references proving valuable in further expanding the understanding of the topics.

The potential of CFD to simulate fluid systems is limited by the cost of doing so. Even in situations where accurate physics models are available, the computational cost of simulating them perfectly – *i.e.*, close to absolute reality – is orders of magnitude beyond the computational power available even in modern super computers. This means that users need to limit the accuracy of their simulations, typically by limiting the level of details – *i.e.*, using coarser grids.

These limitations lead to a certain amount of unavoidable variance when comparing simulations with reality, requiring careful study of the simulation model in order to make meaningful conclusions. Ivings *et al.* (2013), discuss the difficulty of evaluating various competing LNG vapour models, before presenting details of the Model Evaluation Protocol (MEP), an effort to quantify the variance between a simulation model and a set of established values from experimental data. They highlight the effort involved in evaluating LNG models, a process requiring the simulation of 33 separate test cases under MEP. They also express concern over gaps in the available experimental data used for comparison, before advocating for further controlled spill experiments to allow greater validation, while acknowledging the difficulty and cost of acquiring such data.

Chang and Hanna (2004), identified a mean margin of error of +/- 30% to be the mark of a good model, with further model improvement limited by random turbulence. Elsewhere, Sun *et al.* (2013) compared the accuracy of a simulation carried out using the Dense Gas Dispersion model (DEGADIS) to one performed in Ansys® Fluent®, a common CFD system used in academia and industry. They found that DEGADIS had an error rate of 42.38% compared to 19.82% for Ansys® Fluent®, showing both the greater accuracy achieved by CFD when compared to integral based solvers, as well as the inherent margin of error in even a well formed CFD simulation.

These margins of error are important because CFD simulations are used to assess the safety of LNG plants. LNG is a dangerous substance not just because it can be flammable, but also because it can undergo Rapid Phase Transitions (RPTs), and can suffocate people by displacing air (Koopman and Ermak, 2007). Rapid Phase Transitions can occur when liquid LNG interacts with warmer materials, causing the LNG to go from a liquid to gas state with an explosive force comparable to TNT.

Koopman and Ermak (2007) discuss several experiments that have been carried out to study the safety of LNG processes over the years, such as the Falcon, Burro and Coyote series of tests. These experiments allowed for LNG behaviour to be studied and recorded under controlled conditions, with known quantities of LNG 'spilled' so that the resultant vapour clouds could be analysed. A significant challenge in these tests and other similar ones is the impact of RPTs, which can extend the hazard zone of a vapour cloud by explosively dispersing LNG over a large area, or damage the test facilities themselves, notably in in the case of one Falcon test to the point that further tests had to be cancelled, despite the experiment being specifically designed to study RPTs.

The hazard zone of an LNG leak is generally defined as the area between its origin and its Lower Flammability Limit (LFL), which is around 5% LNG to air. The Upper Flammability Limit (UFL) for LNG is 15%, LNG in air with concentrations higher than this being non-ignitable. For many experiments and simulations, Methane is substituted for LNG as LNG is 85-95% Methane by volume (Luketa-Hanlin *et al.*, 2007), meaning the two behave nearly identically, with pure Methane being computationally cheaper to simulate than the multi-species mixture of chemicals that makes up LNG.

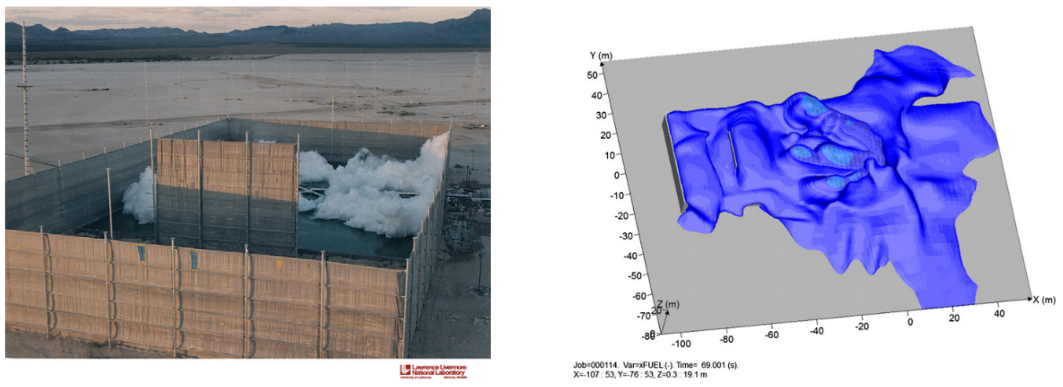
Because of the risk posed by LNG vapor clouds, the design of LNG facilities is subject to strict safety regulations. In the United States the Federal Energy Regulatory Commission (FERC) requires that as part of the design of an LNG facility, the facility must be demonstrated to be safe via modelling. Gavelli *et al.* (2008) discuss in their paper the findings by the FERC that SOURCE5, an integral based modelling system, is insufficient for the purpose of simulating safety.

Further, Gavelli *et al.* (2008) also discuss the use of CFD for modelling hazard footprints, as well as the suitability of CFD over competing systems for performing analysis on LNG facilities. They note that the Sandia National Laboratories strongly recommend the use of CFD for LNG Vapor Cloud modelling in a report they published in 2004 (Hightower *et al.*, 2004). Gavelli *et al.* (2008) also compare a SOURCE5/DEGADIS model to one they created using Ansys® Fluent®, concluding that the Ansys® Fluent® simulation more closely matches experimental data.

Qi *et al.* (2010) also compared CFD simulation to integral based systems. As with the work by Gavelli *et al.* (2008), Qi *et al.* (2010) found that the capacity for CFD to model complex geometry and physical properties made the CFD based simulations preferable to the integral based alternatives. They also show that, somewhat counter intuitively, it is possible for a denser CFD mesh to result in a shorter overall simulation time as the simulations were able to converge faster.

Along with Ansys®, another major vendor in CFD Simulations is GEXCON AS® with their FLACS® Software. Hansen *et al.* (2010) discuss the use of FLACS® in LNG simulation, citing the U.S. Federal Regulations requirement that a LNG facility be shown to have no vapour cloud with a LNG concentration higher than 1/2 of LFL capable of reaching the perimeter of the facility. In their paper, they compare the results of several FLACS® based CFD simulations to results gathered from wind tunnel based experiments, finding that the FLACS® simulations were generally in good agreement with the experimental data.

They reference the work of Hanna *et al.* (2004), who determined that a good model was one with at least 50% of its data points falling within a "factor-of-2 range" of the baseline data. The results achieved by FLACS® against the wind tunnel data, coming in at 88% and 98% respectively, demonstrate that their model can be considered very good. Hansen *et al.* (2010) do go on to note that FLACS® did not perform well when attempting to model the data from the Falcon test series (See Figure 2.2), though at the time of the paper GEXCON AS® were already working to add functionality to the model that they believed would assist in the simulation of Falcon and similar situations.


Y (m)
X (m)
Z (m)
Job=000114. Var=FUEL (-). Time= 69.001 (s).
X=-107 : 53, Y=-76 : 53, Z=0.3 : 19.1 m

(a) Image of the Falcon test, courtesy of R.P. Koopman (b) FLACs simulation of same at approximately 40s

Figure 2.2: Figure 32 and Figure 34 from Hansen *et al.* (2010), showing the Falcon experimental test and the FLACs simulation of the same

More recent simulations involving FLACS® , such as the work in Li *et al.* (2017), show better agreement between experimental data and simulation results, suggesting that the FLACS® software has improved reliably over the intervening years. The paper by Li *et al.* (2017) introduces a computationally cheaper alternative to CFD, Vented Methane-air Explosion Overpressure Calculation (VMEOC). The paper examines the safety benefits of several variants of hatch that are designed to fail gracefully in an overpressure situation, and explores the ability of FLACS® and VMEOC to accurately model the same.

The authors present VMEOC as a way of addressing the costs involved in CFD, not just in terms of software licenses and processing clusters, but also in terms of the labour involved to set up and run a good quality simulation. They provide details on VMEOC, and also demonstrate a good understanding of GEXCON AS® FLACS®. They also describe the dataset they used to examine their simulations accuracy, a small set of experiments that were carried out with purpose built vessels involving the ignition of a methane mix to test the vessels failure in an overpressure situation.

Li *et al.* (2014) also put forward an alternative to CFD in their work, with similar reasoning that CFD "... remains computationally and labour intensive.". The method they present, Confinement Specific Correlation (CSC), is superior to the

Multi-Energy Method sometimes used to model vapour clouds, but falls short of the capabilities of actual CFD. They also advocated for the use of complex geometry in modelling LNG facilities, due to concerns that overly crude geometry can result in inaccurate simulations that fail to capture geometry driven fluid interactions.

In order to examine the accuracy of a CFD code or model, several different methods are used in the literature. Several papers compare their methods to proven systems used in industry, such as FLACS® and Ansys® Fluent®, or use well established experimental data such as that from the Coyote or Falcon test series. Only a small number engage in performing their own experiments which is understandable given the costs and risk involved.

Amongst the papers where researchers generate their own experimental data, several use alternatives to LNG that allow for safer simulation. In Kim *et al.* (2015), experimental data was generated using Helium because it behaves in a reasonably similar manner to Hydrogen without the inherent risks. Differences in behaviour between the two are highlighted, such as Helium not generating condensate like Hydrogen does, which slightly alters the density and behaviour of the cloud.

Kim *et al.* (2015) used the Large Eddy Simulation (LES) code, which provides a coarser approximation of fluid interactions than those provided by Direct Numerical Simulations (DNS), but is more suitable for larger simulations in terms of computational costs. The authors expressed the need to include greater geometric complexity in order for the simulation to more accurately model reality. They concluded that when it comes to safety, compromises may not only be necessary in the simulation of a scenario, but also in experimental data collection, reinforcing the need to err on the side of caution when making safety assessments given the difficulty in gathering accurate data to compare models to.

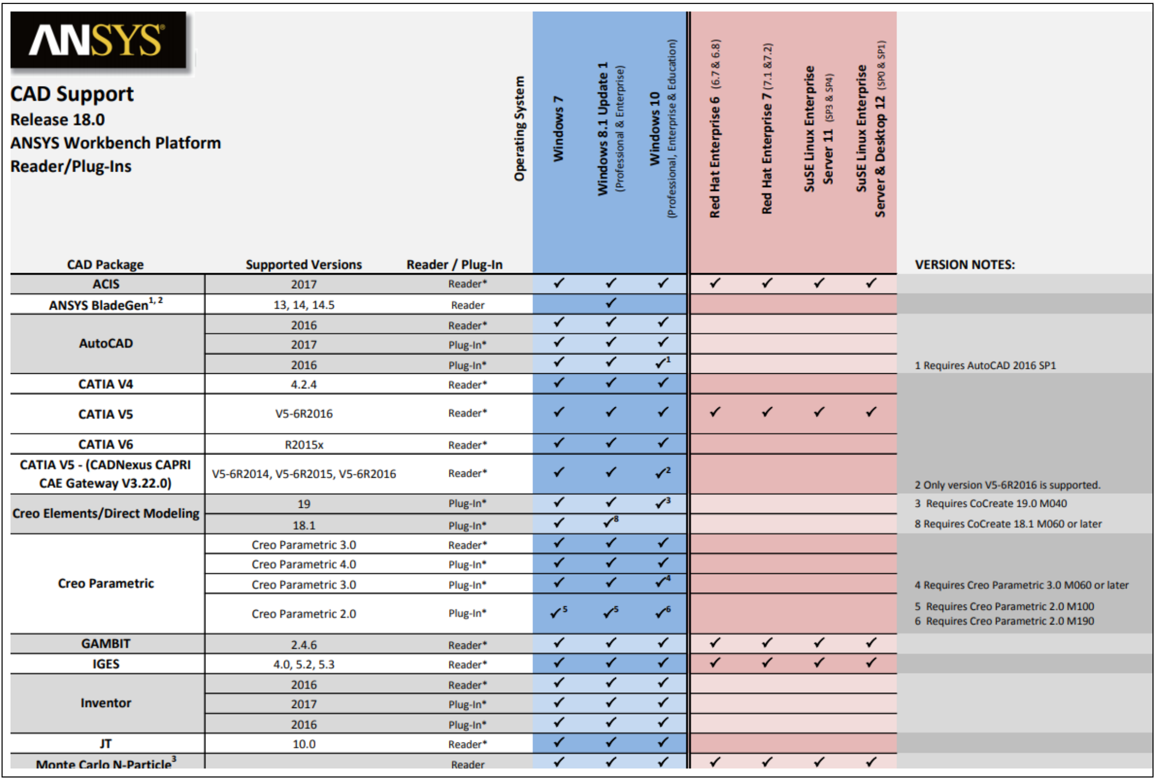
Franke *et al.* (2004) examine and compare several different turbulence models, including LES, κ-ϵ, and κ-ϵ RNG with regards to the problem of wind engineering. While not specific to vapour cloud modelling, their work is still informative given vapour cloud modelling also requires accurate modelling of air flows and propagation, but wind engineering is typically interested in only single species simulations on a broader scale. The authors present some interesting guidelines for the simulation of wind based problems, which may have implications for vapour cloud modelling. They suggest that the top boundary should be placed at $5H$, where H is the building height, and the outlet boundary should be position at least $15H$ beyond the building to allow for flows to develop. They provide several other suggestions about minimal grid density, boundary types and preferred mesh type which were informative to the work carried out for this thesis, but the problem of vapour cloud simulation is sufficiently different that their results did not provide a significant foundation for the work presented in this thesis.

Fothergill *et al.* (2002) concluded that prismatic cells in a mesh were preferable to tetrahedral and, if tetrahedral cells must be used, then they should be used in conjunction with prismatic cells for best results. They recommend the use of second order methods for achieving final simulations results, particularly when looking at time-dependent problems, as they provide a more realistic approximation of physics compared to single order methods.

They also discussed the importance of grid-independent solutions in CFD simulations. When performing a CFD simulation, the problem area is typically broken down into a grid or mesh made up of asymmetrical cells, with each cell covering an area of the problem domain defined by minimum and maximum mesh size settings. The fluid dynamics calculations are then performed on a per cell basis, usually over several iterations so the code can solve each cell while taking into account any impact from neighbouring cells. This means a poorly formed mesh will produce a poor simulation, but even a seemingly well formed mesh can impact the end result.

To avoid this, Fothergill *et al.* (2002) suggest that a grid convergence study should be carried out, involving performing the simulation against three or more variants with increasing cell density, in order to establish the mesh settings required to fully resolve each cell. The results of the simulations should be compared to establish the point at which the increase in cell density no longer impacts the results. If all of the simulations show differing results, then a new grid should be created with cell density greater than the previous highest, comparison continuing until the two densest grids give the same result. At that point, the results from the second most dense mesh can be considered grid independent.

2.3 The interaction of Models and CFD



ANSYS

CAD Support
Release 18.0
ANSYS Workbench Platform
Reader/Plug-Ins

		Operating System	Windows 7	Windows 8.1 Update 1 (Professional & Enterprise)	Windows 10 (Professional, Enterprise & Education)	Red Hat Enterprise 6 (6.7 & 6.8)	Red Hat Enterprise 7 (7.1 &7.2)	SuSE Linux Enterprise Server 11 (SP3 & SP4)	SuSE Linux Enterprise Server & Desktop 12 (SP0 & SP1)	
CAD Package	**Supported Versions**	**Reader / Plug-In**								**VERSION NOTES:**
ACIS	2017	Reader*	✓	✓	✓	✓	✓	✓	✓	
ANSYS BladeGen[1, 2]	13, 14, 14.5	Reader		✓						
AutoCAD	2016	Reader*	✓	✓	✓					
	2017	Plug-In*	✓	✓	✓					
	2016	Plug-In*	✓	✓	✓[1]					1 Requires AutoCAD 2016 SP1
CATIA V4	4.2.4	Reader*	✓	✓	✓					
CATIA V5	V5-6R2016	Reader*	✓	✓	✓	✓	✓	✓	✓	
CATIA V6	R2015x	Reader*	✓	✓	✓					
CATIA V5 - (CADNexus CAPRI CAE Gateway V3.22.0)	V5-6R2014, V5-6R2015, V5-6R2016	Reader*	✓	✓	✓[2]					2 Only version V5-6R2016 is supported.
Creo Elements/Direct Modeling	19	Plug-In*	✓	✓	✓[3]					3 Requires CoCreate 19.0 M040
	18.1	Plug-In*	✓	✓[8]						8 Requires CoCreate 18.1 M060 or later
Creo Parametric	Creo Parametric 3.0	Reader*	✓	✓	✓					
	Creo Parametric 4.0	Plug-In*	✓	✓	✓					
	Creo Parametric 3.0	Plug-In*	✓	✓	✓[4]					4 Requires Creo Parametric 3.0 M060 or later
	Creo Parametric 2.0	Plug-In*	✓[5]	✓[5]	✓[6]					5 Requires Creo Parametric 2.0 M100 6 Requires Creo Parametric 2.0 M190
GAMBIT	2.4.6	Reader*	✓	✓	✓	✓	✓	✓	✓	
IGES	4.0, 5.2, 5.3	Reader*	✓	✓	✓	✓	✓	✓	✓	
Inventor	2016	Reader*	✓	✓	✓					
	2017	Plug-In*	✓	✓	✓					
	2016	Plug-In*	✓	✓	✓					
JT	10.0	Reader*	✓	✓	✓					
Monte Carlo N-Particle[3]		Reader	✓	✓	✓	✓	✓	✓	✓	

Figure 2.3: An excerpt from the Ansys® 18.0 support document on CAD compatibility. [1]

This research is not the first to look at using existent models in conjunction with CFD. The practice is well enough established that many industry standard tools for CFD have built-in support for the import of CAD/BIM models in one form or another. Due to the competitive and often proprietary nature of the Architecture, Engineering and Construction (AEC) industry, the support for different file formats and versions of software can vary greatly. The complexity of matching CFD and CAD software can be seen in the version compatibility table in Figure 2.3. The

[1]https://www.ansys.com/-/media/ansys/corporate/files/pdf/solutions/it-professionals/platform-support/cad-support-18.pdf, accessed 2019.07.05

changing nature of CAD/BIM software and file formats means that typically only recent releases of widely used CAD/BIM software are likely to be supported, work-arounds such as saving to an older file format sometimes being necessary. While there are some open formats which would ideally provide an easy way for information to be shared between all parties, support for these open formats can also be quite variable as developers prefer to focus on their proprietary formats where they can include additional features to encourage the use of their ecosystem of products.

At present, there is no direct support in Ansys® Workbench® for importing BIM models. One of the currently viable ways to import the models is to utilise CAD as an intermediary, going from BIM to CAD to CFD. There has been significant research on the process of moving from CAD to CFD, which in itself is a challenging transition, but the transition from BIM to CFD is even more so.

Converting a BIM file to a CAD file is a manual process, requiring anyone in the AEC industry who is working with a BIM file to output the CAD file and then keep track of the file and ensure it is up to date before any simulation is run to avoid wasting considerable resources simulating an out of date design. Even if the end user has an effective system for file management to ensure an up to date CAD file is available for any simulations, CAD files do not contain all the information a BIM file does. Information such as material properties are lost, preventing any automatic assignment of materials within the CFD simulation.

The process of migrating data from BIM to CFD has a significant impact on the cost and time of CFD simulations, because inclusion of small-scale features in large-scale simulations may not only make CFD simulations prohibitively computationally-expensive, it may also prevent the simulation from converging. Cormier *et al.* (2009) discusses the complexity of such environments with regard to LNG trains, warning that overly simplified simulations can miss risks, such as when an ignited gas cloud interacts with complex geometry to produce a shock wave. Shock waves can result from ignited vapour being constrained by pipes and surfaces, causing the pressure to build into an additional risk to safety and facility integrity on top of the existing fire. CFD Simulations can predict such risks, but not in situations where too much detail has been removed for the fluid simulation to allow the outcome to be modelled.

Li *et al.* (2014) discuss how some methods of calculating risk that are used in the LNG industry, such as the Multi-Energy Method (MEM), are inadequate. Similar to Cormier *et al.* (2009), they stress that complex environments can heavily alter the effects of an accidental ignition. MEM in particular can be more than an order of magnitude out in its calculations because it does not take into consideration the complex geometry of an industrial facility, which can cause pressure waves to be focussed or channelled in complex ways.

Dawes *et al.* (2001) emphasise the need to reduce the time required to progress from CAD to mesh to CFD simulation. They discuss in their work that, at present, the CAD to CFD process presents a significant bottleneck, preventing CFD from efficiently contributing to the design process. Their presented solution is to create new software, utilising improved meshing algorithms, to try and improve the process.

Li *et al.* (2016) identifies the current bottleneck in going from CAD to CFD as a serious problem for Simulation Based Design (SBD). They note that performing CFD simulations currently is complicated by the degree of knowledge required by the user, meaning the potential benefits of cyclical simulation are difficult to explore. They suggest the creation of a meta file to allow the transfer of information from design to simulation, with *fluid physics feature*, *dynamic physics feature* and *fluid functional feature* used to convey intent and support automation.

The work by Li *et al.* (2016) has the potential to improve CAD/CFD integration, though is still at an early stage and will require further development. There is no single 'silver bullet' solution to the current challenges facing integrating CFD with existing models. Given the wide range of applications of CFD, it may not even be possible to create a single solution, instead requiring a variety of novel solutions for different use cases.

Once a model is in a CFD compatible format, numerous papers in the literature demonstrate CFDs versatility and potential. Zhang *et al.* (2012) present work on modelling coal mine evacuations using CFD, Yang *et al.* (2011) present work on simulating a storehouse fire to establish cause, and Al-Kodmany (2013) discuss how CFD has been used to help plan crowd management for the Islamic pilgrimage of Hajj. Sun *et al.* (2014) have demonstrated the capability of CFD to help confirm safe design within the scope of an LNG facility, thus showing the potential of CFD to inform design.

Aliabadi *et al.* (2006) discuss the cost in generating a city level mesh of Atlanta; requiring 64 person months of time. While meshing an entire city seems like a more difficult task than meshing an LNG facility, city level simulations are typically carried out with heavily simplified geometry and relatively large cell sizes, meaning they are computationally much cheaper for comparable size areas. Later in the same work they discuss the creation of a mesh for performing dispersal simulations against Oklahoma city. To generate the mesh, they used a surface grid cell resolution of 10 metres, giving them a coarse mesh of 4,338,393 tetrahedral elements. For comparison, a single LNG Module ($4m \times 4m \times 6m$) with a simulation domain of only $6m \times 6m \times 30m$, and a relatively coarse grid size of 10cm would produce a mesh of at least a million cells.

The large person hour costs described by Aliabadi *et al.* (2006) has driven work on automation, their paper referring to the use of a custom in-house automation system to assist in the work they were presenting. Their work should not be used to conclude that all conversion from model to CFD requires months of work, but to illustrate the scales of effort that can be required. In terms of mesh complexity, the interior of a single building or the external geometry of an LNG train is likely to far exceed that of a city scale simulation.

In a city-scale simulation, buildings are typically represented by very simple versions of the actual building which are suitable for the intended task, as can be seen in Figure 2.4. Meanwhile, the interior of a building or an LNG train is likely to require a much finer mesh and thus the capture of much finer details using more complex

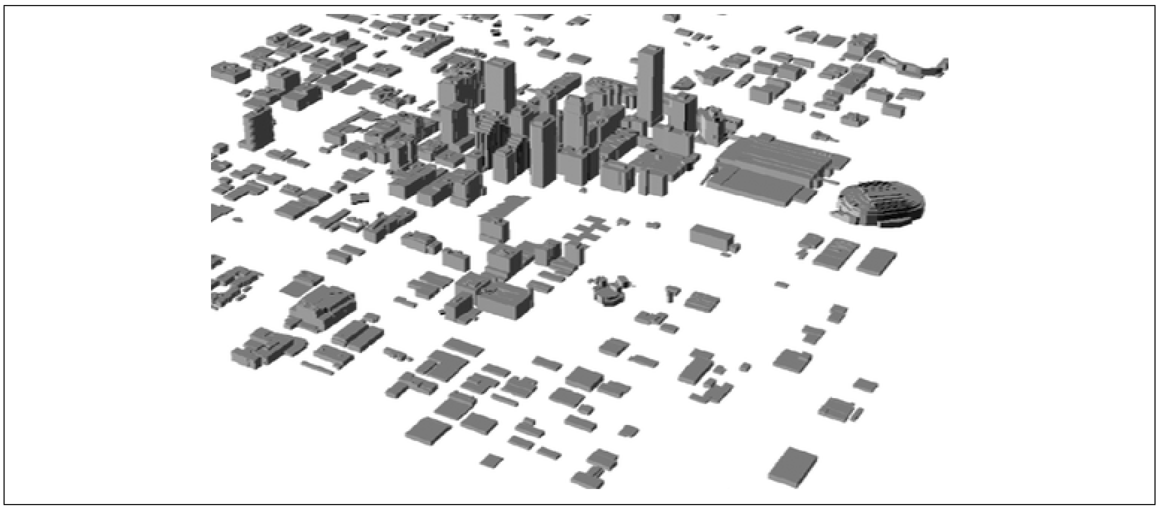


Figure 2.4: Figure 20 from Aliabadi *et al.* (2006), showing a city scale geometry model. Note the relatively simple shapes used.

geometry. Should fully automatic conversion not be available or suitable, the computationally much smaller problems of a building interior or LNG train would likely require much greater outlays in terms of person hours to make them simulation ready.

The work by Kawaguchi and Oguni (2016) makes it obvious that even when working with relatively simple representations at a city scale, small issues within the geometry can introduce problems. They looked at the challenges and methods for automatically converting digital maps to CFD capable meshes for city scale simulations. One of the challenges they addressed was small gaps introduced to the topology or mesh when converting from the original representation.

An example of the kind of issues they were looking to address can be seen in Figure 2.5. In Figure 2.5b, it can be seen that Object 2 does not sit flush with Object 1 down the common long side. If not addressed, this would leave a small gap that will likely cause a sliver at the time of meshing, resulting in a lower quality mesh.

To resolve this issue, Kawaguchi and Oguni (2016) implemented existing algorithms to identify small gaps and apply an appropriate correction. In the case of Figure 2.5, dealing with a gap of only 1 mm on a building over 5 metres, they resolved the issue by re-cutting the objects to allow merging of the overlapped portions, then

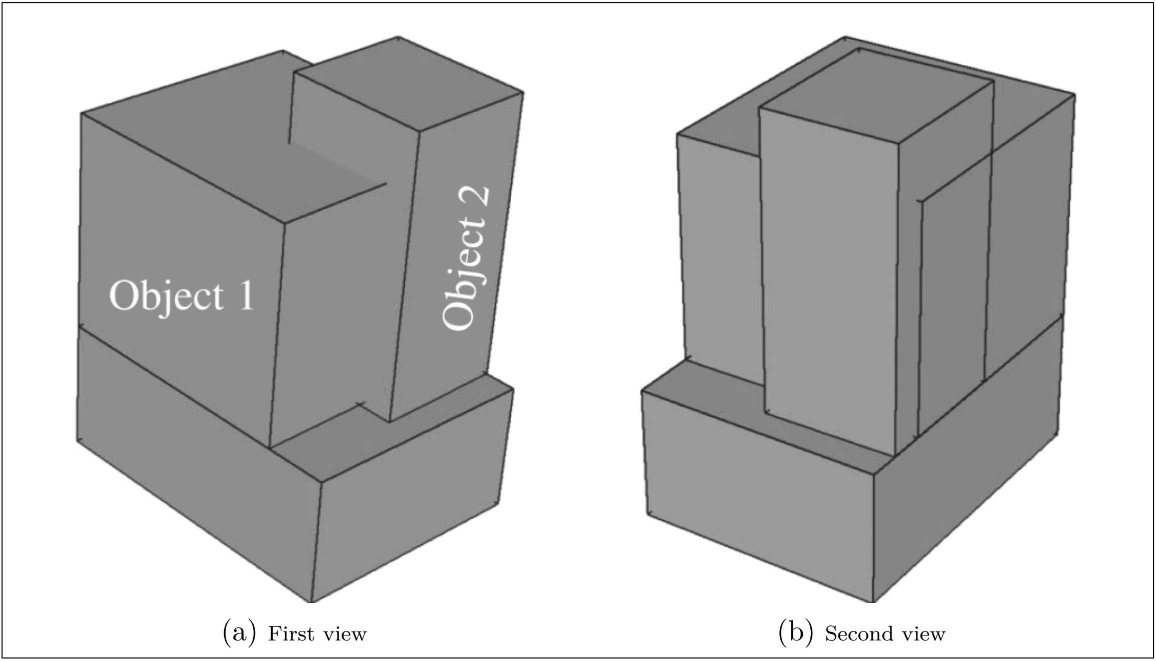


Figure 2.5: Figure 2 from Kawaguchi and Oguni (2016), showing the multiple simple geometric shapes used to represent a building. Note the need for multiple, overlapping shapes, which do not necessarily align well.

adjusting the size of Object 1 to align its edge with the base. Removing gaps like these at a city scale will have a minimal impact on observable flows, while significantly reducing the difficulty of meshing the geometry, improving the quality of the resulting mesh, and reducing the computational cost of simulations.

The following chapters will present details of the current study, which aims to make CFD more accessible by reducing the knowledge and time costs of moving from an existent model to a completed CFD simulation. Chapter 3 will detail the solution built to address the current challenges in going from geometry to simulation. In Chapter 4 and 5, two test cases utilising the developed solution are described, with analysis of its impact.

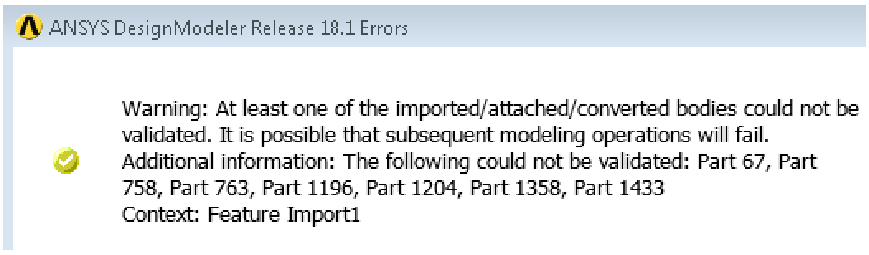
Chapter 3

Level Of Detail and Object Simplification

The main aim of this research is to reduce the cost of performing useful CFD simulations, both in terms of person hours and the required computation resources. The starting point for simulation is with any existing model assets, such as a Building Information Model (BIM) or CAD representations. Ansys® Fluent® was chosen as the CFD solver to be used in this research due to its wide usage in both academia and industry, which introduces a small challenge as it is part of the Ansys® Workbench® software suite which does not support BIM import at this time.

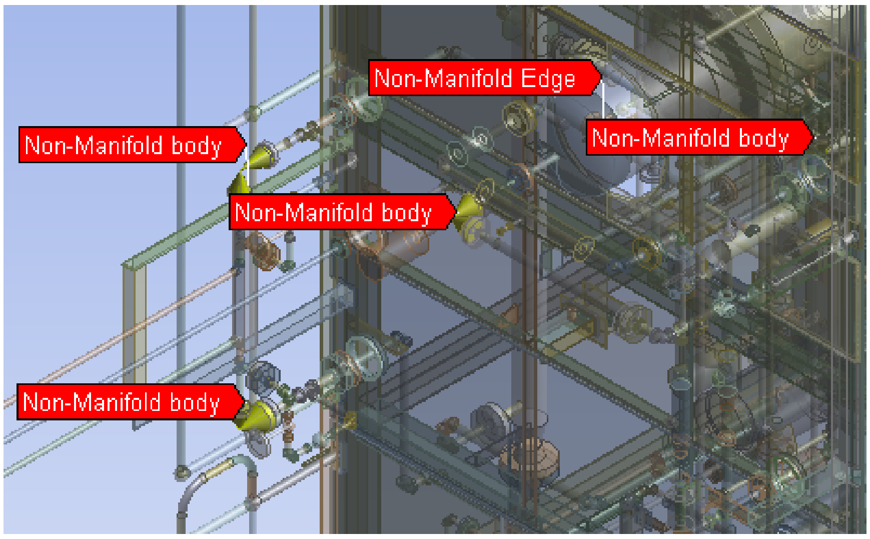
Ansys® Workbench® does support the import of CAD models from several programs, including Autodesk® AutoCAD®. Autodesk® offers strong interoperability between their software packages, meaning it is possible to take a BIM model in Autodesk® Revit® and transfer it into Autodesk® AutoCAD® (Autodesk.help, 2020), though with a loss of information as the BIM is translated into CAD. From Autodesk® AutoCAD® the data can be imported into Ansys® Workbench® , and thus made available to Ansys® Fluent® using the process provided by Ansys®.

Autodesk® AutoCAD® provides an API for the development of plugins (Porter, 2014), through which plugins can access and alter information from a loaded model. These toolsets in tandem offer a path for beginning with a BIM file in Autodesk® Revit® and taking it to the point of CFD simulation in Ansys® Fluent® . To achieve this outcome, there are are several challenges in how the tools interact that must be addressed such as different base units, and default geometry orientations. Tools in the Ansys® Workbench® suite can also have difficulty when interacting with complex models, with operations on larger models proving to be slow and even failing in some instances. Non-standard geometry imported into Ansys® DesignModeler® and the Ansys® Workbench® default meshing system also presents issues, as can be seen in Figure 3.1.


ANSYS DesignModeler Release 18.1 Errors

Warning: At least one of the imported/attached/converted bodies could not be validated. It is possible that subsequent modeling operations will fail.
Additional information: The following could not be validated: Part 67, Part 758, Part 763, Part 1196, Part 1204, Part 1358, Part 1433
Context: Feature Import1

(a) The error message generated when importing unusual geometry



(b) Some of the problematic elements, tagged by Ansys® DesignModeler®

Figure 3.1: An example of problematic geometry being imported, Ansys® DesignModeler® 18.1 is unable to parse it correctly or automatically mend it, so it ends up as a non-manifold object.

Manually correcting problematic geometry, either by removing it or replacing it with a suitable alternative, is resource intensive. To make BIM to CFD a more economical and accessible process, existing software and methods presented in literature were examined for possible solutions, as covered in Chapter 2. Finding no sufficient solution, the idea of using a Level of Detail (LoD) process similar to what is used in computer graphics was formed.

Within computer graphics, the term Level of Detail can apply to several different methods for manipulating geometry during visualisation, all with the purpose of reducing the required processing. A simple example is ‘Z-buffer filtering’, where polygons that cannot be seen from the viewport are filtered out, so time is not spent drawing them before drawing over them with obstructing polygons. Another common method is to reduce the number of polygons used by distant objects, on the basis that with the object being far away, it will not greatly impact the visualisation to use an “abstracted”/less detailed form of the geometry, while removing the processing overhead of drawing each of the polygons present in the actual geometry.

Automatically replacing problematic geometry with simple alternatives within a CFD context using Level of Detail (LoD) based methods were examined. Within CFD there is the concept of Dimensionality, the idea that when looking at a problem, any elements with dimensions below a certain threshold can be removed or simplified, as they lack the necessary dimensional presence to impact the simulation in a significant way. Similar to a LoD algorithm removing or reducing geometry at a distance for a visualisation, some small features and geometric detail can be removed from a CFD simulation without noticeably impacting the result.

It is not uncommon for CFD practitioners to limit the complexity of their geometric models, reducing the resources required to simulate the scenario. Several papers in the literature describe the process across a range of research areas, with some using formulae such as boundary roughness to help compensate, while others discuss the reduction and provide reasons why it does not negatively impact their simulations (Ji *et al.*, 2007; Rong *et al.*, 2016; Tominaga *et al.*, 2008; van Hooff and Blocken, 2010). No single book or paper was found within Chemical Engineering or CFD that explicitly advocates for geometry simplification, but the widespread use in literature shows that simplifying geometry is accepted practice.

Given the difficulty of dealing with large scale complex models, and that much of the complexity is undesirable to simulate, algorithms for converting complex models to an abstracted (simplified) form were devised. Two main systems were developed, Simplification and Cubification, with the first seeking to maintain as much of the original geometry’s flow profile as possible, while the later produces a much simpler approximation of a model's airflow. There are strengths to both systems and they were developed to be usable iteratively, and in combination with each other or themselves, allowing a user to control the level of detail they wish to keep. The algorithms are collectively called the Level Of Detail and Object Simplification system, or the LODOS system.

The following sections first present a method for taking BIM geometry through to being inside Ansys® Workbench® (3.1), ready for meshing and simulation. Details of developing a plugin within Autodesk® AutoCAD® (3.2) are then discussed, explaining some of the benefits and challenges of using the API, while providing context for later sections' descriptions of how the LODOS algorithms work. A brief explanation of the process for using LODOS within Autodesk® is provided (3.3), before the concepts and methods underpinning the main algorithms are presented; Hiding (3.3.1), Simplification (3.3.2) and Cubificiation (3.3.3). This chapter concludes with a discussion about the limitations of the current LODOS implementation (3.4).

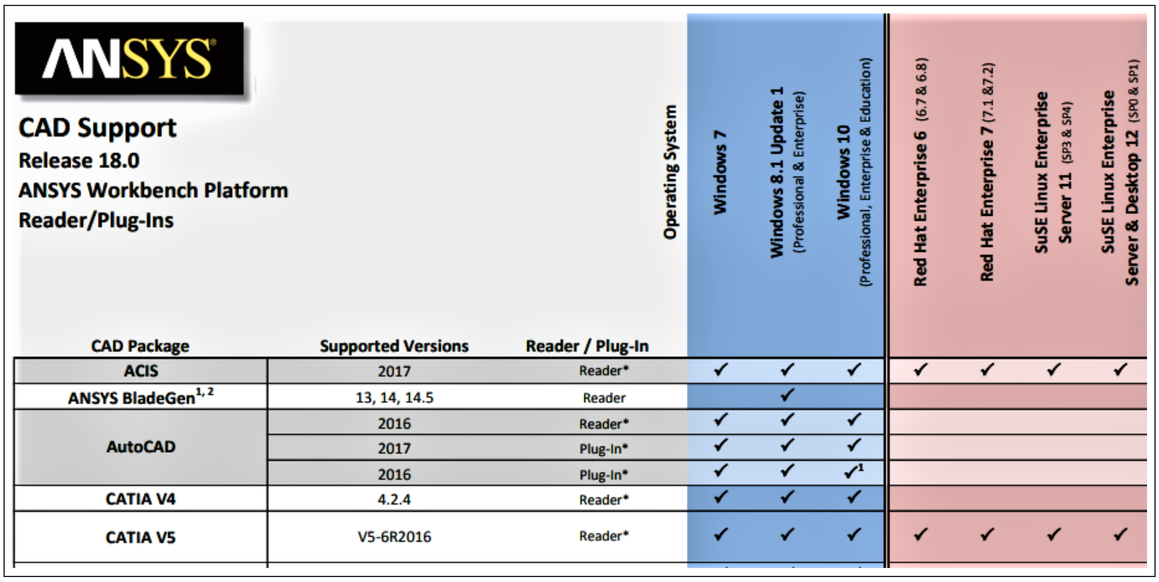
3.1 BIM to CFD

While guides exist for portions of the BIM to CAD/CAD to CFD process, no complete BIM to CFD process could be found described in the literature. This section provides an explanation of the steps used to go from a BIM file to meshable geometry in Ansys® Workbench®. This process is not presented as a novel addition to the field of study but simply to help others navigate issues encountered during this research, and as such mostly covers the Autodesk® Revit®, Autodesk® AutoCAD®, and Ansys® Workbench® versions used in this work.

Ansys® Workbench®, at least up to version 18.1, does not support direct import of any BIM file formats, so to use a BIM file it must first be converted to a supported format. BIM software packages typically support exporting to one or more CAD file formats that are well supported within Ansys® Workbench®. In this research, the BIM files begin in Autodesk® Revit®, in the Autodesk® proprietary ".rvt" file format, and the **File** ⇒ **Export** option is used to output the file as an Autodesk®

AutoCAD® drawing file.

The Autodesk® AutoCAD® drawing file, which uses the ".dwg" extension, can be read by Ansys® Workbench®, but only certain versions of Autodesk® AutoCAD® are compatible with certain versions of Ansys® Workbench®, as can be seen in the excerpt in Figure 3.2. Users should check their chosen BIM/CAD/CFD software for compatibility information, but typically the CFD software must be newer than the BIM/CAD software. For example, Ansys® Workbench® 2018 and Autodesk® AutoCAD® 2017.



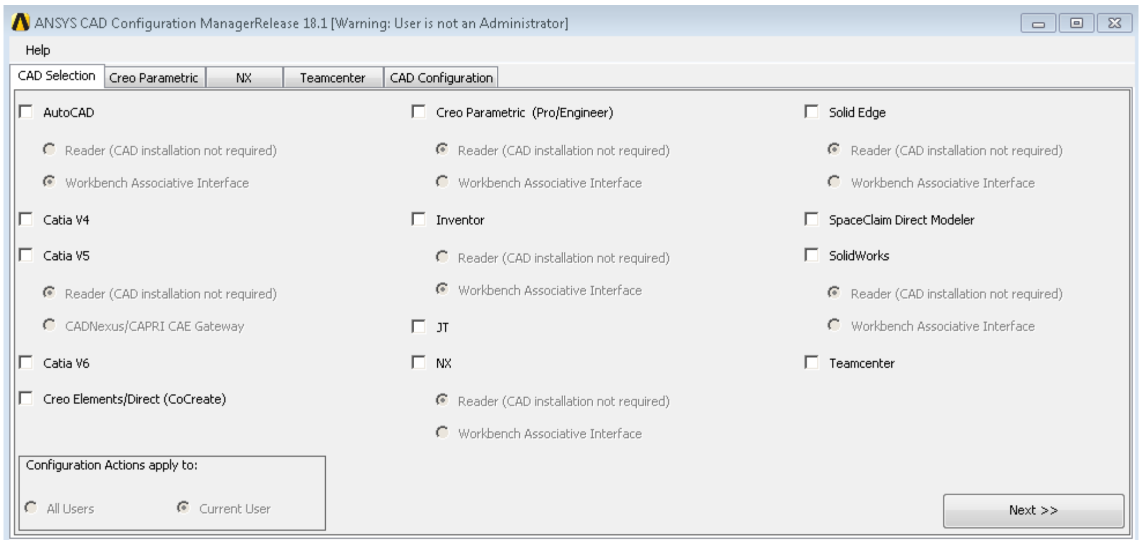
CAD Support
Release 18.0
ANSYS Workbench Platform
Reader/Plug-Ins

			Operating System						
CAD Package	Supported Versions	Reader / Plug-In	Windows 7	Windows 8.1 Update 1 (Professional & Enterprise)	Windows 10 (Professional, Enterprise & Education)	Red Hat Enterprise 6 (6.7 & 6.8)	Red Hat Enterprise 7 (7.1 &7.2)	SuSE Linux Enterprise Server 11 (SP3 & SP4)	SuSE Linux Enterprise Server & Desktop 12 (SP0 & SP1)
ACIS	2017	Reader*	✓	✓	✓	✓	✓	✓	✓
ANSYS BladeGen[1, 2]	13, 14, 14.5	Reader		✓					
AutoCAD	2016	Reader*	✓	✓	✓				
	2017	Plug-In*	✓	✓	✓				
	2016	Plug-In*	✓	✓	✓[1]				
CATIA V4	4.2.4	Reader*	✓	✓	✓				
CATIA V5	V5-6R2016	Reader*	✓	✓	✓	✓	✓	✓	✓

Figure 3.2: An excerpt from the Ansys® Workbench® 18.0 compatibility information[1].

In the case of Ansys® Workbench® it offers support for direct communication with some CAD packages, which can save time exporting a CAD file with one program, then importing it with a CFD package. The time saved for each geometry import sums to a significant time saving over the course of iterative development work, and reduces the risk of errors due to the wrong file version being loaded. Ansys® provide instructions on how to link Ansys® Workbench® with CAD software on their help pages, which should be the first point of reference for any user, with the following steps based on their instructions provided with commentary to help avoid issues encountered during this research.

In order to integrate the two packages, the Ansys® CAD Configuration Manager should be run with administrator privileges, an example of its user interface can be seen in Figure 3.3. From this interface the relevant CAD package(s) should be selected, in the case of this work Autodesk® AutoCAD®, and proceed to the CAD Configuration tab. Select **Configure Selected CAD Interfaces** and the Configuration Manager will attempt to create the interface between the two packages. There are often issues linking the systems, but Ansys® CAD Configuration Manager provides error messages and a log file to assist users in resolving them.


ANSYS CAD Configuration ManagerRelease 18.1 [Warning: User is not an Administrator]
Help
CAD Selection
Creo Parametric
NX
Teamcenter
CAD Configuration
Next >>

AutoCAD
- Reader (CAD installation not required)
- [✓ Workbench Associative Interface]

Catia V4

Catia V5
- [✓ Reader (CAD installation not required)]
- CADNexus/CAPRI CAE Gateway

Catia V6

Creo Elements/Direct (CoCreate)

Creo Parametric (Pro/Engineer)
- [✓ Reader (CAD installation not required)]
- Workbench Associative Interface

Inventor
- Reader (CAD installation not required)
- [✓ Workbench Associative Interface]

JT

NX
- [✓ Reader (CAD installation not required)]
- Workbench Associative Interface

Solid Edge
- [✓ Reader (CAD installation not required)]
- Workbench Associative Interface

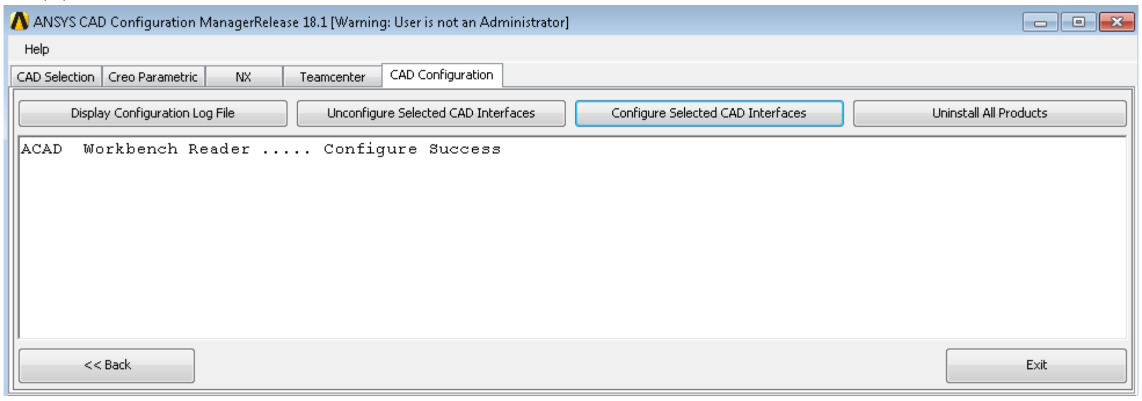
SpaceClaim Direct Modeler

SolidWorks
- [✓ Reader (CAD installation not required)]
- Workbench Associative Interface

Teamcenter

Configuration Actions apply to: All Users [✓ Current User]

(a) The CAD Selection panel, allowing the user to choose which packages are integrated and at what level.


ANSYS CAD Configuration ManagerRelease 18.1 [Warning: User is not an Administrator]
Help
CAD Selection
Creo Parametric
NX
Teamcenter
CAD Configuration
Display Configuration Log File
Unconfigure Selected CAD Interfaces
Configure Selected CAD Interfaces
Uninstall All Products
<< Back
Exit

```
ACAD  Workbench Reader ..... Configure Success
```

(b) The CAD Configuration panel, were a user may configure their selected packages to be integrated, or 'Unconfigure' them to remove the links.

Figure 3.3: A screen capture of the Ansys® Workbench® 18.1 CAD Configuration Manager control panel

In this research, three different versions of Autodesk® AutoCAD® and two different versions of Ansys® Workbench® are used, requiring the CAD linkage to be setup several times. During this process, errors were encountered that required careful tracing of the log to resolve. Through trial and error, the following steps were established as best practice to improve the chances of the software packages linking without issue:

- Ensure only one version of the CAD and CFD software are installed.
- Install software in default locations, e.g., **C:\Program Files\<software>**.
- After installation and before linking, run each package once.
- Restart the computer after installation and before attempting linking.

Once the software is installed and any link established, the method for progressing from BIM/CAD to CFD ready geometry varies depending on what software is being used. Users should consult the documentation for their particular software, or search the internet for applicable tutorials, to determine the proper steps to follow. In this research, the following method is used: Inside Ansys® Workbench®, add a new **Fluid Flow (Fluent)** to the Project Schematic, then right click the Geometry field on the Fluid Flow element and select new Ansys® DesignModeler® geometry. During geometry import the model can generate at the wrong scale, if the imported CAD file uses millimetres as its base unit, as Ansys® Workbench® uses metres as its default. To resolve this, it is necessary to establish what scale the CAD package uses. Once the CAD package scale is known, set Ansys® DesignModeler® to use the same scale by opening the **Units** menu item and setting it to the appropriate value. Some files and packages handle this conversion automatically while others require manual correction.

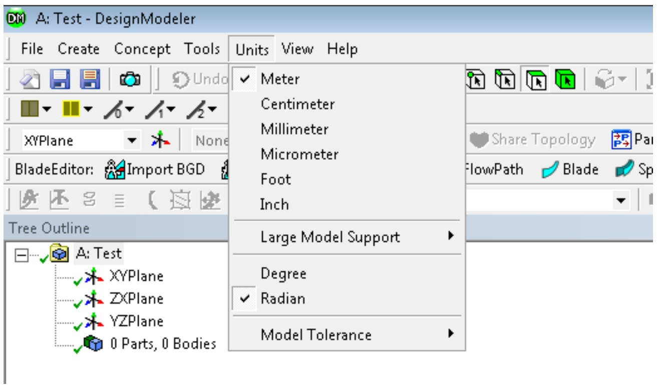
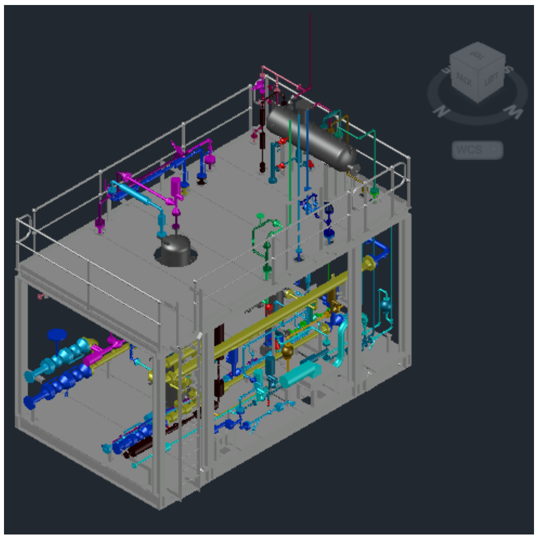
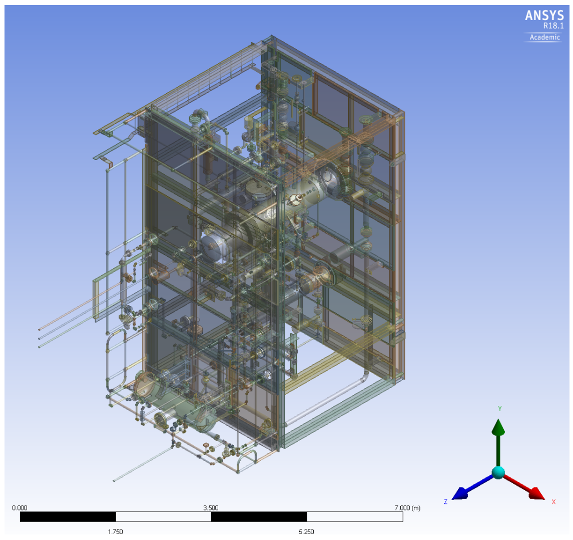


Figure 3.4: Ansys® DesignModeler® 18.1 units menu, for setting unit scale and Degree or Radian

To import the geometry within Ansys® DesignModeler® , the menu item **File** ⇒ **Import External Geometry File...** is used and the source geometry file selected. When using linked packages, the menu item **File** ⇒ **Attach to Active CAD Geometry** is instead selected while the appropriate CAD package is open with the file loaded. Once the geometry source is set, the **Generate** operation is run in Ansys® DesignModeler® to begin reading in the geometry. Once done, the geometry appears in the view plane.

Another issue is that the CAD and CFD packages used different co-ordinate systems. Autodesk® AutoCAD® uses positive X for right of origin, positive Y for away from origin and positive Z as up from origin. Ansys® DesignModeler® uses positive X for away from origin, positive Y as up from origin and positive Z as right of origin, which results in imported geometry being oriented differently in each package, as illustrated in Figure 3.5.


(a) The LNG Module in Autodesk® AutoCAD® 2018.

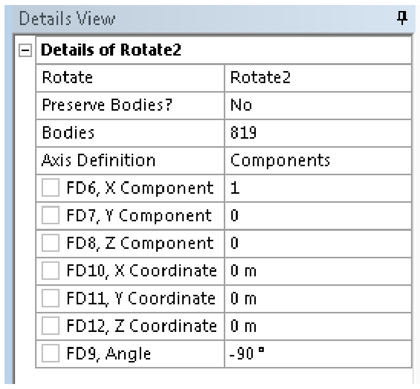


(b) The LNG Module after import in Ansys® DesignModeler®.

Figure 3.5: A comparison of the LNG Module saved in Autodesk® AutoCAD® and loaded in Ansys® DesignModeler®, without any changes to orientation or co-ordinates applied.

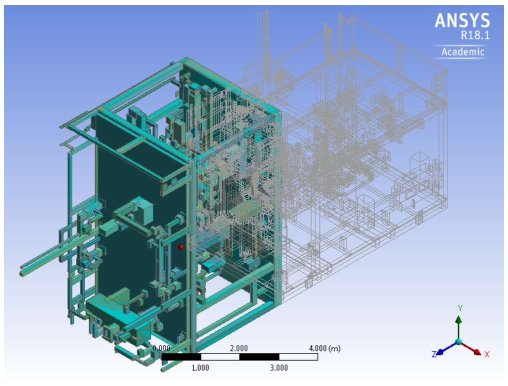
There are a few options for keeping the models aligned so the vertical and horizontal remain the same from a user view perspective, such as changing the co-ordinate system used in the CAD software, or re-orienting the geometry in the CFD package after import. To re-orient the model within Ansys® DesignModeler® to match the orientation from Autodesk® AutoCAD®, two sequential rotations can be used, starting with selecting **Create** ⇒ **Body Transformation** ⇒ **Rotation**, then selecting all the geometry parts and defining the axis of rotation using **Components**.

For the first transformation, the **X Component** should be set to 1, everything else to 0, and then -90 degree entered for the rotation, as per Figure 3.6. Ansys® DesignModeler® provides a preview outline of the, but the transformation is not actually applied until **Generate** is selected. This operation reorients the selected components around the origin so that the ZX plane is "down" for the imported geometry. To complete the reorientation, apply a second -90 degree transformation, this time with only the **Y component** set to 1.

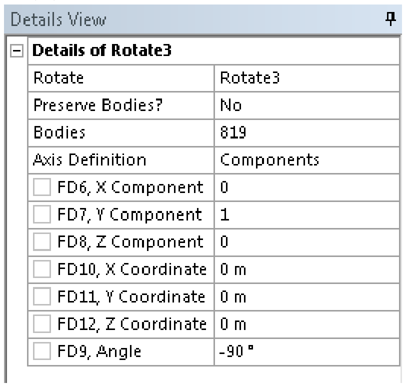


Details of Rotate2	
Rotate	Rotate2
Preserve Bodies?	No
Bodies	819
Axis Definition	Components
FD6, X Component	1
FD7, Y Component	0
FD8, Z Component	0
FD10, X Coordinate	0 m
FD11, Y Coordinate	0 m
FD12, Z Coordinate	0 m
FD9, Angle	-90 °

(a) Settings for the first operation.

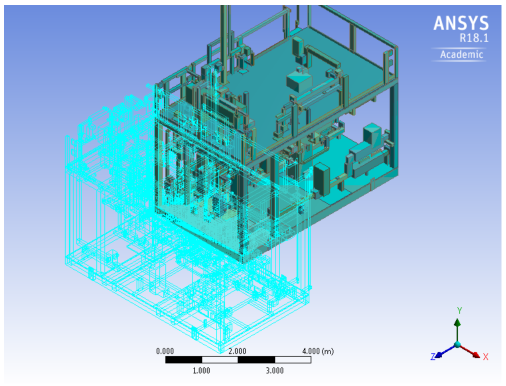


(b) The geometry, with a preview of the operation in grey.



Details of Rotate3	
Rotate	Rotate3
Preserve Bodies?	No
Bodies	819
Axis Definition	Components
FD6, X Component	0
FD7, Y Component	1
FD8, Z Component	0
FD10, X Coordinate	0 m
FD11, Y Coordinate	0 m
FD12, Z Coordinate	0 m
FD9, Angle	-90 °

(c) Settings for the second operation.



(d) The geometry, with a preview of the operation in light blue.

Figure 3.6: Ansys® DesignModeler® 18.1 reorient example

Once the geometry is imported and oriented as desired, a simulation domain is defined to allow for meshing. Within Ansys® DesignModeler®, select **Create** ⇒ **Primitives** ⇒ **Box**, then position the object to encase the imported geometry, with dimensions for the box chosen to define a suitably large problem area. Use **Create** ⇒ **Boolean** to apply a boolean-subtract operation, removing the imported geometry from the box to create an air domain for meshing.

In Ansys® DesignModeler® it is best to apply "named selections" to the faces next, so that during later meshing and simulation stages it is easier to track and set-up inlets, outlets and boundaries. To reduce the setup time for simulations with the same domain but different geometry, duplicate a working simulation in Ansys® Workbench®, then open the duplicates Geometry field in Ansys® DesignModeler® and change the file referenced at the import stage to the alternate geometry. This allows most of the Named Faces to be kept and ensures each geometry is positioned and treated the same way through each subsequent step.

While the above process differs depending on which specific software packages are used, it is presented to aid in identifying possible issues and how to navigate them. Having geometry ready to mesh is still quite far from having a mesh ready to simulate, but simulation preparation depends on the meshing solution used, and the simulation goals. The next section looks at the issues in developing the devised solutions as a plugin within Autodesk® AutoCAD® .

3.2 Working with Autodesk® AutoCAD® via the API

To manipulate the geometry of a CAD file through Autodesk® AutoCAD® in the way needed, it is nescessary to build a plugin using the Autodesk® AutoCAD® API. Development of such a plugin is costly, as each test of the plugin entails running Autodesk® AutoCAD® . Each test can take 10 to 15 minutes as the software is loaded followed by the model and the plugin, before executing and checking the output of the plugin.

The layer of removal represented by running as a plugin also makes correcting errors difficult, as if the plugin causes Autodesk® AutoCAD® to crash, only limited crash information is provided to the user/developer. The API is also restrictive, only exposing a limited set of the Autodesk® AutoCAD® functionality to a plugin developer. For example, the Autodesk® software must logically have complete details about how a 3D object is composed to be able to represent it in the view window. Even a primitive object such as a cylinder needs to be stored so that Autodesk® knows where to draw each end and how long to make the object. Working with the API, however, there is no apparent method or user readable fields that allow an object to be identified as a cube, cylinder, sphere, or something more exotic.

The only information exposed by the API about the shape of a 3D object is the bounding box; a simple record of the upper and lower xyz values necessary to draw a box containing the entire object. Along with the centre point, this allows the location and upper volume limit of an object to be identified, but no detailed information about its shape. Fields can be manually populated with extra data, so it is possible to manually label each object by its type and then use that to filter how objects are treated, but this is a very time consuming way to make the information available to a plugin.

When working with 3D CAD models, every object within Autodesk® AutoCAD® has a unique identifier known as the ObjectID, from complex 3D object to simple 2D lines. Through the API it is possible to request a list of ObjectIDs based on provided criteria. A developer can then loop through these ObjectIDs and request the Entity from Autodesk® AutoCAD® that relates to that ObjectID. During the development of LODOS this system presented some challenges; The entity object is a high level object, and because every item in the drawing has an ObjectId, the properties available on a given entity can vary significantly. This can lead to errors when developing loops to process retrieved entities if the plugin attempts to access a property of interest, such as the bounding box which is found on many but not all objects that inherit from the entity class. To avoid such errors, before attempting to read the properties of a retrieved entity the **ObjectClass.DxfName** property can be checked to ensure it is either **3DSOLID** or **BODY**, the two most common 3D object types and the items of interest for Geometry Abstraction.

While this additional checking adds stability to the plugin, it also results in unexpected behaviour with some models. Some models define an object such as a support beam as a collection of surfaces and lines, not as a solid 3D object. At present, the plugin simply ignores such objects, which can lead to LODOS producing results where the model does not match expectations, see Figure 3.7.

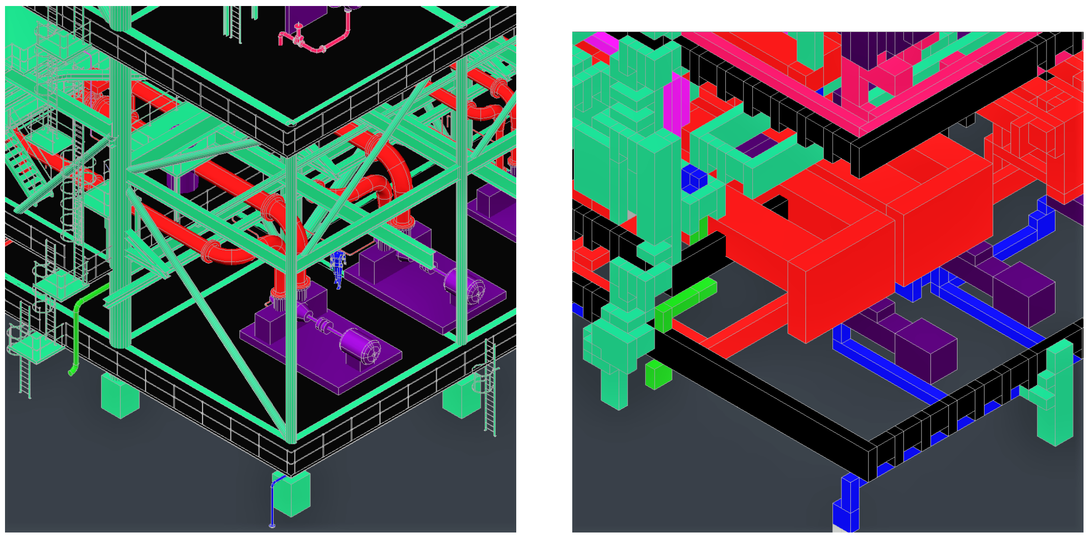

(a) The original model (b) The LODOS abstraction

Figure 3.7: An example of unexpected detail loss in a LODOS abstraction due to ObjectClass incompatabilities.

As the API limits what information is available to the plugin, there is no simple resolution for this issue. A designer developing a model with LODOS in mind could ensure that all geometry they wish to alter is of the 3DSOLIDs type, but this increases the burden on the designer. It also limits LODOS to only working fully on models built for it, instead of being widely usable on any existing model asset, a design goal that the Autodesk® AutoCAD® environment was specifically chosen to make possible.

The Autodesk® AutoCAD® environment does provide a useful feature in the form of its **layer** system, based on the traditional design practice of using transparent sheets to allow important information to be layered on top of an existing drawing, or removed to make important details more clear. This feature is familiar to users of Autodesk® AutoCAD®, so by leveraging it to control what geometry is included in LODOS operations, the LODOS system can be more intuitive to use. By only including visible objects on visible layers in any LODOS operation, users familair with Autodesk® or similar design programs can easily control the scope of any given operation.

This feature is further exploited in LODOS by adding two layers when it is first run on a model, a working layer and a results layer. The working layer, **LODOSworking**, is used as a scratch pad to temporarily store objects as the solutions run each of their algorithms in turn. Any objects that are specified for exclusion from alteration, but are to be included in the final result, are copied straight to the **LODOS** layer which is used for the final output.

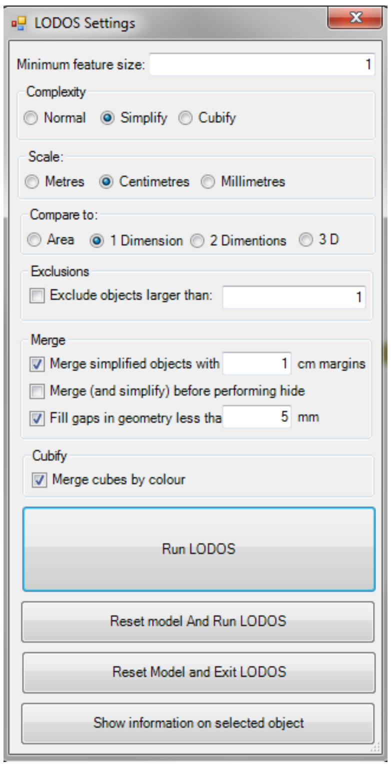
Once all algorithms have completed, the remaining objects from the LODOSWorking layer are copied to the LODOS layer, before the LODOSWorking layer is cleared. Along with the layer system allowing LODOS to be more accessible to BIM/CAD users, it means LODOS is non-destructive, allowing a user to revert their model simply by turning layers on and off. The current prototype includes a function for doing this, allowing the user to revert the model and run LODOS again with new settings to see the results, or to simply revert to their original geometry and exit LODOS.

Working with the Autodesk® API to implement the LODOS system has provided both benefits and challenges. Checking an API will provide necessary functionality at the outset of development is important to do but difficult to achieve, as it is often not possible to know what functionality is needed to create a working solution. The Autodesk® API is well supported and allows for at least a partial implementation of LODOS without needing a BIM/CAD system to be built from scratch, but other researchers should examine the options available to them with reference to their goals before deciding to work within a given API.

In the next sections the design of the proposed solutions, and how they are implemented via the Autodesk® AutoCAD® API, is discussed. The user interface created to provide access to the functionality in LODOS is presented together with examples of how different settings can affect an abstraction.

3.3 Using LODOS

When running LODOS within Autodesk® AutoCAD® , the control panel shown in Figure 3.8 is presented. The control panel presents options for how to manipulate the geometry, along with accepting the input necessary to control the algorithms.


LODOS Settings

Minimum feature size: 1

Complexity

Normal [✓ Simplify] Cubify

Scale:

Metres [✓ Centimetres] Millimetres

Compare to:

Area [✓ 1 Dimension] 2 Dimentions 3 D

Exclusions

☐ Exclude objects larger than: 1

Merge

✓ Merge simplified objects with 1 cm margins

☐ Merge (and simplify) before performing hide

✓ Fill gaps in geometry less tha 5 mm

Cubify

✓ Merge cubes by colour

Run LODOS
Reset model And Run LODOS
Reset Model and Exit LODOS
Show information on selected object

Figure 3.8: The LODOS control panel

Because it is designed for iterative applications, LODOS writes out the settings from the interface to a model specific ‘state’ file each time it is run. Upon launching, LODOS first checks if a state file exists, and reads the settings back into the interface if it does, or loads default values if it does not. Along with the values from the user interface, the state file includes a list of the layers that were visible when LODOS was run, so that the system can restore a model to its initial layer state when requested. This functionality is included to allow a user to apply multiple iterations of LODOS algorithms quickly and easily, while also being able to revert to a pre-LODOS state as required.

The first and most important option in the control panel is the **Minimum feature size** box. This controls what size of objects is either manipulated by, or generated by, the LODOS functions. The selection of which abstraction functions should be applied is controlled by the buttons grouped under **Complexity**.

To perform Hide operations against the original geometry, **Normal** should be selected. To abstract the geometry, either **Simplify** or **Cubify**, based on the desired abstraction algorithm. The **Scale** group that follows is applied to the number in the minimum feature size field, allowing for the minimum feature size to be specified in **Metres**, **Centimetres** or **Millimetres**.

The **Compare to** controls apply to **Normal** and **Simplify** operations, being disabled during **Cubify**. This option affects how the minimum size option is applied to the geometry during Hiding and Simplification. It allows control of how an object is determined to be too small to include, and if **Exclusions** is enabled, when an object is too large to abstract.

The **Merge** controls affect merging operations performed during Simplification. These options are detailed under **Simplification and Merging** (3.3.2). The merging algorithm allows two adjacent objects to be turned into one, with the controls defining a threshold at which that should occur. The final control, **Merge cubes by colour**, is applied during **Cubification** (3.3.3).

Cubification colour codes cubes based on the objects that generated them to make it easier to determine what parts contribute to what cube. If Merge cubes by colour is used, then Cubes are grouped together based on colour, meaning cubes generated by a single object will all be combined into the minimal number of prisms needed to fill the area. If this option is not used, Cubes are grouped together using a simple greedy algorithm to reduce object count, applying the colour of the seed cube to the eventual combined prism.

The remaining buttons determine how LODOS is run, or in the case of **Reset Model and Exit**, not run. The **Show information on selected object** provides information on the XYZ co-ordinates of a selected object, along with information such as ObjectID, which assisted during development in identifying and tracing elements. This should not be considered a permanent feature of LODOS.

The following sections details the abstraction algorithms, the reasoning for the development of each, and how they function. They are presented as Hiding (3.3.1), Simplification and Merging (3.3.2), and Cubification (3.3.3).

3.3.1 Hiding

The Hiding method takes in the minimum size and Compare information from the LODOS UI panel. The Compare buttons allow selection of whether items need to have one, two or three dimensions which meet or exceed the minimum size. Alternatively, items can be compared to their area.

The concept for this algorithm emerged from discussions about how CFD engineers actually model an object for simulation. If a engineer is looking to model the airflow over a building, and that building is surrounded by a wire fence, then the engineer will most likely ignore the wires in the fence. While the wires will obviously impact the actual flow of air, their impact on the scale of modelling a building is so minimal as to make them not worth the person hours to recreate in the model or the computational time that would be required to capture a feature of their size.

To assist in removing such objects from a BIM/CAD geometry, the Hiding algorithm gives a versatile and powerful way to isolate small features. In the case of a wire fence, the wires will likely have one large dimension and two small dimensions. So a minimum size of 5mm in any two dimensions can be specified to remove any wires or similarly small features from the model by the use of LODOS.

For the Hide algorithm, LODOS requests a list of item IDs for all visible objects from Autodesk® AutoCAD®. This allows items which have been hidden previously to be excluded. The item IDs are turned into a list construct in C#.

A list performs much the same as an array does in most programming languages, providing a number addressable collection of objects. The advantage of the list over a standard array is that items can be added or removed from the list without needing to perform a full array copy, giving it some of the dynamic usage of a linked list. This was particularly useful within the Hiding and Merging algorithms, allowing for a dynamic list of objects to be considered as elements were removed or added.

The pseudocode for the Hiding algorithm can be seen in Figure 3.9.

```
hiding(autocadDocument aCAD, float minSize, int numDimensions)
  list<itemID> itemIDs
  object current
  3dPoint xyzMin
  3dPoint xyzMax
  bool hide
  object copied

  itemIDs ← aCAD.getVisibleObjects() #Retrieve visible objects from autocad

  foreach itemID in itemIDs
    current ← getObject(itemID)
    if (current.type == (3DSOLID || BODY))
      xyzMin ← current.xyzMin
      xyzMax ← current.xyzMax

      #Check if the objects dimensions are below the hide thresholds.
      #  If so, functions return true. Else, return false.
      if(numDimensions ≠ 4)
        hide ← checkDimensions(minSize, xyzMin, xyzMax, numDimensions)
      else
        hide ← checkArea(minSize, xyzMin, xyzMax)

      #if the object is not to be hidden, make a copy of it on the LODOSworking
          layer
      if(hide is false)
        copied ← current.copy()
        copied.layer ← working
  endforeach

  aCAD.layers(LODOS).objects.removeAll()
  aCAD.layers(LODOS).objects.copy(aCAD.layers(working).objects)
  aCAD.layers(working).objects.removeAll()

  foreach layer in aCAD.layers
    if(layer.name ≠ LODOS)
      if(layer.visible is true)
        save layer.name
        layer.visible ← false
  endforeach
end hiding()
```

Figure 3.9: Pseudocode representation of the Hiding algorithm

To control the number of dimensions that are examined, numDimensions can be set as 1 through 4, with 1 through 3 representing the number of dimensions that should exceed the minimum. To reduce the number of variables that need to be passed to the function call, numDimensions can be set to 4 if the area is the ’dimension’ of interest. In the checkDimensions function, the number of dimensions that exceed the minimum are counted, and at the end of the function true is returned if that number is smaller than the passed in numDimensions.

During development, the Hiding algorithm had to be adapted to work around a few restrictions with the Autodesk® AutoCAD® API. There is no apparent way to access the actual geometry of the object being dealt with, so all examinations of dimensions must be done against the bounding box for that object. This presents some challenges. In the case of a cube, the bounding box has the same dimensions and minimum and maximum points as the cube itself. In the case of a sphere, the bounding box sides are tangent to the sphere, but by necessity also describe a larger volume. This introduces uncertainty when dealing with a sphere or cylinder, but is more problematic if the object is a complex geometric shape, made up of several smaller objects, or is a thin but long object aligned diagonally.

With no way to access the geometry directly, the algorithms must rely on the values provided by the bounding box and assume that they provide a relatively good representation of the underlying object. This provides a small benefit during the Simplification process, discussed in Section 3.3.2, but generally introduces uncertainty about the accuracy of the algorithms when implemented via Autodesk® AutoCAD®. In a future implementation, access to the actual geometry would enable more options for abstraction.

A second issue is that Autodesk® AutoCAD® has many different types of objects, such as text labels, 3D objects and point to point lines, but only 3D objects have a bounding box, so if one of the itemIDs returned by getVisibleObjects() is for a line or 2D spline, attempts to access the xyzMin or xyzMax from the non-existend bounding box of that object will crash Autodesk® AutoCAD®. To prevent this, defensive code was added to the function that, after the itemID has been used to instantiate an object, checks if the object is either a Body or 3DSolid.

3.3.2 Simplification and Merging

When performing CFD simulations, particularly on facilities such as LNG trains, there can be numerous small features such as pipes which are computationally expensive to simulate but may only have a minimal impact on flow, depending on size. There are several methods available to a CFD practitioner when confronted with this problem. Oberkampf and Trucano (2002), looking at the problem from an aerospace standpoint, recommend that large complex systems be broken down into 'building blocks'. So rather than attempt a high quality CFD simulation of an entire aeroplane, just the engine can be modelled, then that simulation can be used to inform a simulation of the wing, and so on.

While such a method can provide higher confidence in simulation results and allow the models to be validated at each level, it requires a great deal of resources. An alternative used in the FLACS® simulation system, popular in the mining and energy (M&E) industry for modelling of crisis events, is to use a larger size per mesh cell and then approximate occlusions within the cell with sub-cell formulae (Li *et al.*, 2014, 2017). While the approach taken by GEXCON AS® enables useful simulations, it also imposes some limitations on users in terms of mesh and simulation method.

As covered earlier in this chapter, Geometry Abstraction is a commonly accepted practice within CFD, often determined by the dimensionality of the problem. If a simulation is focused on a set of pipes, or their impact on simulation flow is likely to be significant, then the CFD practitioner setting up the simulation will need to model them appropriately. However, in a situation where the pipes need to only be approximately represented, Simplification can be of aid. Rather than having to manually replace the pipes with alternative geometry, Simplification automates the process by replacing more complex geometry with simple rectangular prisms.

One use case envisioned for this is when needing to study an object inside a complex array of objects, such as the flare tower in an LNG facility. By using Simplification all the pipes, taps, and railings can be abstracted from the facility unless exclusion is turned on and they exceed the specified threshold. This would allow the original geometry of the flare tower to be kept, while Simplifying geometry of lesser concern.

Because the only information readily available on the geometry of an object is its bounding box, it is not possible to determine if an object is a complex geometric shape or even which simple geometric shape it might be. When replacing a simple cylinder, abstracting it to an octagonal column would provide a good alternative, being easier to mesh than a cylinder while retaining some of the flow characteristics. But given only bounding box dimensions, it is only possible to replace elements with a prism generated from the bounding box dimensions.

The Merging algorithm was developed as an aide to Simplification, as it was noted while working on a test model that visually continuous objects, such as pipes, are often modelled as multiple individual parts in engineering designs. This allows the BIM to store information on each segment, such as last service date and supplier information, but can vastly increase the number of objects in the model. This greater number of objects tends to make the model more difficult to load and work with in a CFD suite.

Merging goes through the objects in the model and intelligently combines adjacent items which share common dimensions. To achieve this, LODOS goes through each item and compares it to every other item. The current iteration of the algorithm is therefore not optimal, being $O(n^2)$, but in testing on the models available the comparison process takes a few seconds while the drawing of the objects by Autodesk® AutoCAD® can take up to a couple of minutes, depending on the model complexity and algorithms used. For proof of concept purposes, the current algorithm is adequate, though it would be worth implementing a sorting algorithm such as Octree if LODOS sees wider use.

A pseudocode representation of the Merge algorithm can be seen in Figure 3.10. For each possible merger between two objects, a mergeCheck() function is called that examines them to see if they should be combined based on meeting at least one of three possible conditions. The first is if one object encloses the other, the second

```
merging(autocadDocument aCAD, float margin)
  list<itemID> itemIDs, compareIDs
  object curr, comp
  bool merge

  itemIDs ← aCAD.getVisibleObjects()

  foreach itemID in itemIDs
    merge ← false
    curr ← getObject(itemID)
    compareIDs ← itemIDs.copy()
    compareIDs.remove(itemID)
    foreach compareID in compareIDs
      comp ← getObject(compareID)
      merge ← mergeCheck(margin, curr.xyz.min, curr.xyz.max, comp.xyz.min, comp.xyz.max)
      if (merge is true)
        object.new ← combine(curr.xyz, comp.xyz)
        new.layer ← working
        itemIDs.remove(compareID) //Only possible when using C# lists
        escape foreach
endmerging()
```

Figure 3.10: Pseudocode representation of Merging algorithm, see Figure 3.11 for the pseudocode of the mergeCheck() function it calls

if the two objects are adjacent and within margin of the same size on the adjacent side, and the third is if one object is wrapped or 'sleeved' around the other.

If at least one of the merge checks passes, then the merge algorithm generates a new object using dimensions from the parents. Both parent objects are then removed from the comparison list, so the remaining objects do not waste time comparing to them.

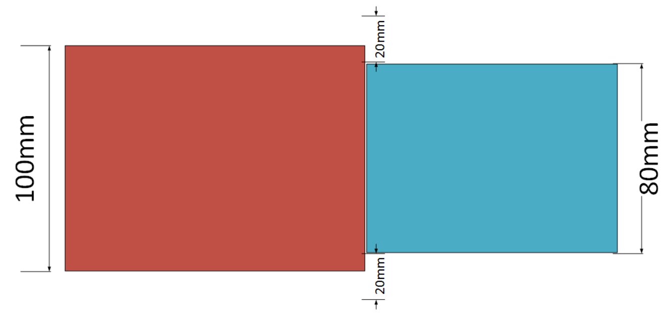
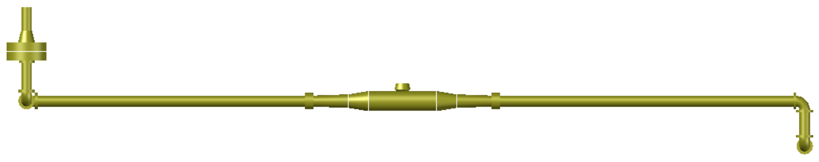


Figure 3.12: An example of two objects that would be merged, based on a specified 20mm user margin

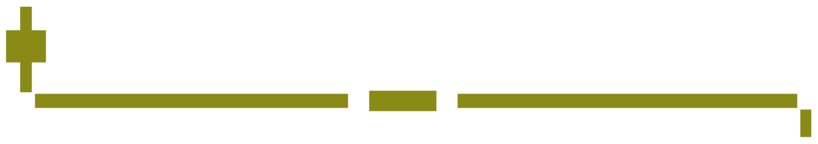
```
mergeCheck(float margin, 3dPoint minA, 3dPoint maxA, 3dPoint minB, 3dPoint maxB)
  bool merge ← false

  //If A completely covers B, then merge them
  merge ← doesACoverB()

  //If A is next to B, and their size is within margin of each other
  if (!merge)
    merge ← adjacentMerge()

  //Check if B sleeves A along the X axis
  //A needs to extend through B on X axis
  if (!merge && ((minA.x < minB.x) && (maxB.x < maxA.x))
    //As Y dimensions are bigger than Bs
    && ((minA.y - margin) ⩽ minB.y) && ((maxA.y + margin) ⩾ maxB.y)
    //As Z dimensions are bigger than Bs
    && ((minA.z - margin) ⩽ minB.z) && ((maxA.z + margin) ⩾ maxB.z))
  {
    merge ← true
  }

  //Repeat sleeve check for Y axis
  ...
  //Repeat sleeve check for Z axis
  ...
  //Return the value of merge.
  return merge
endmergeCheck()
```

Figure 3.11: Pseudocode representation of mergeCheck algorithm

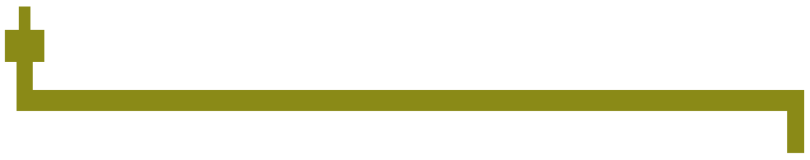
As an example, look at Figure 3.12, which shows a red and a blue object. When the Merge operation compares them, it does so with a user specified margin of 20mm. Given any margin of 10mm or more, the Merge algorithm will replace the red and blue objects with a new child object with a 100mm Y axis, from the red parent, and a X axis that is a combination of both parents.



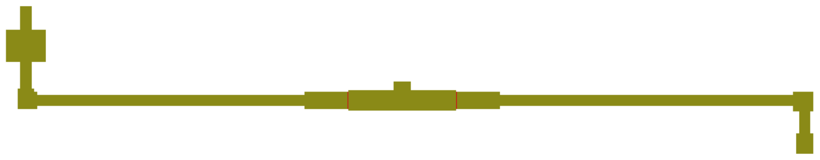
(a) The original pipe geometry



(b) Hide all segments that do not have at least 1 Dimension that is 10cm, then merge all remaining segments within a 5cm margin of their neighbours.



(c) Same as above, but with the Merge operation performed before the hide operation.



(d) Hide all segments that do not have at least 1 Dimension that is 5cm, then merge all remaining segments within a 2cm margin of their neighbours.

Figure 3.13: The images contained in this figure depict a section of pipe with variations of the Merge and Hide algorithms applied.

For an example where the effects of using several variations of Merge and Hide on the same piece of geometry can be seen, look to Figure 3.13. At present all settings used by LODOS must be determined by the user considering their geometry and their simulation goals. The settings used in these examples illustrate how even quite similar settings can have quite varied results. As shown in Figure 3.13b and 3.13c, the exact same Merge and Hide parameters can have drastically different results if the operations are applied in a different order, as objects that would have been removed on their own can become part of a larger object which passes the Hide check.

Figure 3.13d shows a Simplification that closely matches the original geometry while reducing geometric complexity, with the Hide and Merge values based on an examination of the base geometry. While the Hide operation is performed before the Merge operation, the minimum size has been set at 5cm, meaning it does not remove any of the lengths of pipe considered of interest. When using Merging and Hiding, a user should examine the original geometry and consider which elements are too small to have a significant impact, as well as what important elements may be removed by a given setting. This can difficult, particularly on large and complex models, but LODOS also allows the abstraction process to be performed iteratively, so a user can apply one set of values and examine their impact before trying again with different values.

3.3.3 Cubification

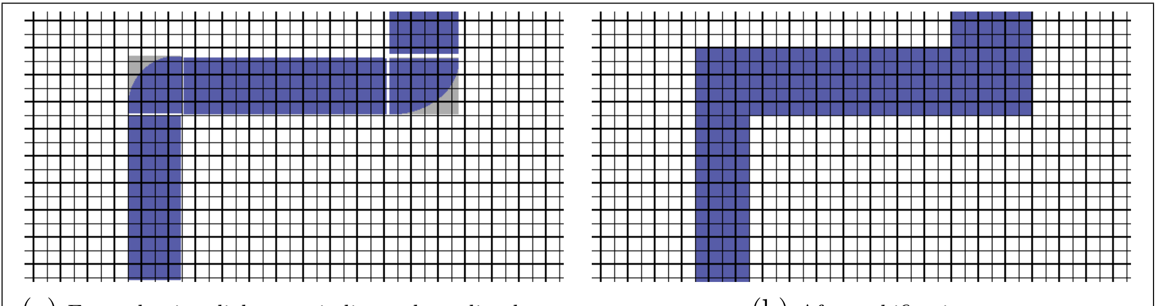

(a) Example pipe, light grey indicates bounding boxes (b) After cubification

Figure 3.14: A simple 2D representation of a pipe before and after cubification

While Simplification provides a lot of utility, in complex models with small and irregular spaces between objects it can encounter problems. If performing a Simplification operation on a collection of small pipes with taps and collars, the resulting geometry can retain small gaps from the original or introduce new ones as a side effect of several bounding boxes in close proximity. To resolve this issue and provide a simple way to mesh models with dimensional requirements that are less fine, Cubification was developed.

Cubification divides the problem domain into a 3D grid made up of cells, or cubes, of a specified size. It then iterates through all of the objects, comparing their bounding box to the grid. Wherever an object would impinge upon a cell, that cell is marked for filling. This leaves the user with an abstraction of the model were no feature face can be smaller than the specified size of their cubes, an example of which can be seen in Figure 3.14.

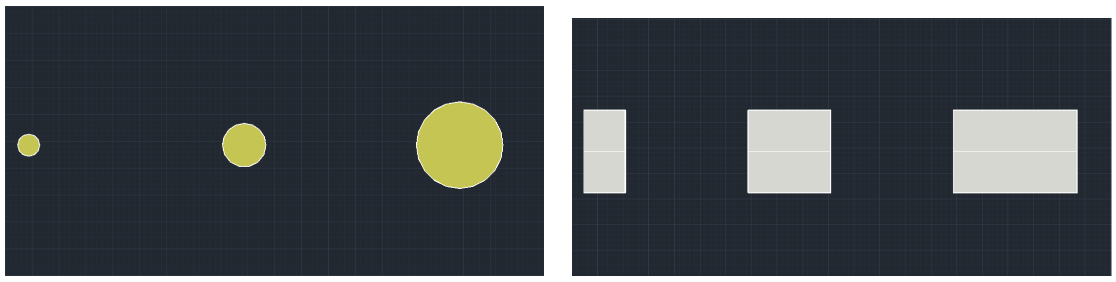

(a) Three pipes of 10, 20 and 40 cm diameter respectively (b) After applying a 20cm Cubification Abstraction

Figure 3.15: A comparison of a model before and after a 20cm Cubification

The present implementation of Cubification does not take into account how much of an object impinges upon a grid cell, if it is impinged at all then that cell is marked for output. This can result in “grid creep”, where the Cubified representation will take up more space than the original geometry, as can be seen in Figure 3.15. The grid creep is made worse in the example image by the grid starting from Origin, meaning that the left most pipe lies across the boundary of two adjacent grid cells.

Grid creep is an inescapable side effect of the Cubification process, which presently relies upon user intervention to mitigate. Both Hiding smaller objects, and using a smaller cube size can help limit the amount of space consumed by Grid Creep, but add additional costs in terms of person hours or processing time. As with other LODOS functions, Cubification can be used iteratively or with exclusion criteria, meaning a user could use Cubification on the outskirts of their simulation domain to reduce geometric and meshing requirements, then use Simplification or the original geometry nearer the area of interest.

Early implementations of the Cubification algorithm took an unexpectedly long time to run, several minutes in the case of a moderately sized model and over an hour for a large and complex model it was tested against. The problem was investigated and it was found that the LODOS algorithm itself actually completed quite quickly, but the usability bottleneck was being caused when drawing the cubes back out. Autodesk® treated each cube as a separate draw call, meaning it had to render each cube to the viewport before the next could be added to the working layer.

The large number of cubes present in some abstractions also presented challenges when importing into Ansys® Workbench®. A workaround for this, to use Autodesk® AutoCAD® to perform a **Union** operation on the cubes converting them into a single object, did help with the Ansys® Workbench® import issues but did not resolve the Autodesk® AutoCAD® draw issues. The usability issue was sufficient to justify revising the Cubification algorithm to group the cubes based on adjacency, allowing cubes to be output as merged clusters.

A greedy algorithm was implemented that, after the Cubification grid has been fully populated, moves through the grid looking for a cell marked for output. Upon finding one, it will then search adjacent cells, attempting to form the single largest cube it can. Once it can add no more cubes, it outputs the large Cube, then marks all the contributing cells from the Cubification grid as empty. To allow the cubes to be clustered based on colour, a second grid storing colour information from the first object that caused that grid cell to be marked is used.

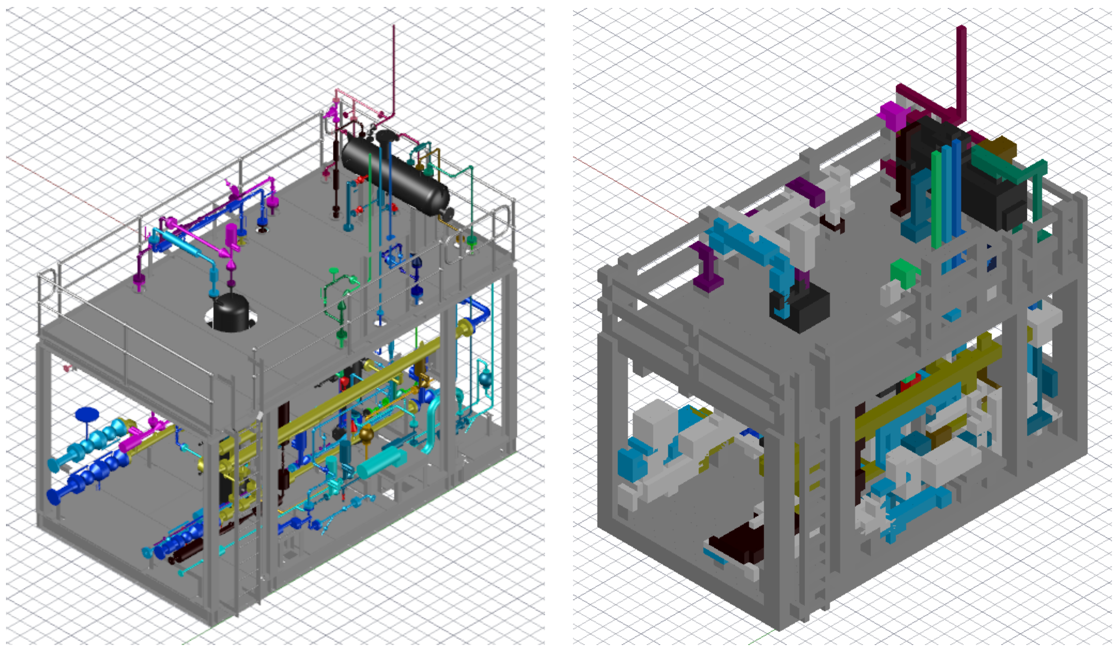

(a) The original BIM of an LNG module (b) The LNG model after applying a 20cm Cubification

Figure 3.16: A comparison of an LNG model before and after a 20cm Cubification

Because objects in a BIM tend to have colour codes based on their type or material, Cubification is likely to form collections of cubes caused by the same object when using the colour based grouping. While the grouping technique was originally implemented for performance reasons, the group by colour option has the added benefit of making it much easier to compare the Cubified abstraction to the original model, as can be seen in Figure 3.16.

3.4 Discussion

CFD analysis can be prohibitively costly, with Geometric Abstraction offering a method to reduce the cost while keeping the dimensionally significant details. The present implementations of the abstraction algorithms via the LODOS system are limited, with a few notable weaknesses. These limitations come about due to the limited resources common to research, where only so much time can be spent on any given problem.

Geometry Abstraction being restricted to simple prisms is one notable weakness. The inability to assess the nature of the geometry in a given 3DSOLID means intermediary Geometric Abstractions, such as replacing a cylinder with an octagonal column, are currently impossible. Rectangular prism abstractions often obstruct fluid flow in ways the original geometry or a close approximate would not, reducing the available simulation fidelity when using Geometry Abstraction.

Another weakness is unexpected behaviour during certain abstractions. Most of the development of the LODOS plugin was carried out against two models, with a further much more complex model acquired late in the research. Due to the time and effort involved in performing CFD simulations and analysis against the first two models, the third model was only briefly examined as it was clear there would not be sufficent resources to perform CFD simulation and analysis against it.

During this examination a few abstractions were performed against the large model as a test of the system and anomalous behaviour was observed, such as elements that should be abstracted vanishing, and some abstracted elements appearing different than expected. These problems were only observed on this model in a few locations, and given insufficient resources to simulate and analyse the model, minimal resources were applied to identifying and resolving them. Unlike the other models used, this model began in a Bentley® MicroStation file format, leading to uncertainty over whether the abnormal behaviour was a result of converting the file from Microstation to Autodesk® AutoCAD® to Ansys® Fluent®, or problems with how the current implementation handles known objects or its inability to handle an unknown type of object.

Despite these weaknesses, the current LODOS system provides enough functionality to study the impacts of Geometry Abstraction. It is a functional package with a feature set well suited to making its intended use, iterative Geometry Abstraction, easy to carry out. While further development resources would allow the system to be improved and functionality expanded, the present implementation is well suited to examining Geometry Abstraction as a path from existing model assets to completed CFD Simulation.

The following chapters present examples of using LODOS to abstract models and the results of CFD simulations against those abstractions. Chapter 4 presents a small and relatively simple model, nicknamed the “Pan Pipes” model, as a test case for examining the impact of abstraction on CFD simulations. Chapter 5 presents a much more complicated model based on an LNG Module, a test case that explores how Geometry Abstraction can make a realistically complex models accessible to CFD simulation analysis.

Chapter 4

Case Study: Simple Geometry

Figure 4.1: An example of a Pan Flute/Pan Pipes, image courtesy of Wikimedia

While development of the LODOS system began using the LNG Module geometry, the complexity of that model and the computational cost of simulating it means it is not ideal for the iterative improvement cycles common to research. The difficulty of meshing the LNG Module geometry was an inciting incident that motivated this research, as the base geometry cannot be simulated without significant manual rework. It also means it can not be simulated against to provide a baseline for comparison to any abstracted geometries, making it necessary to use an alternative geometry in order to quantify the impact of the proposed solutions against.

To this end, a conceptually simple model was developed for testing; the "Pan Pipes" model, which can be seen alongside the LNG module model in Figure 4.2. The Pan Pipes model consists of a small series of increasing size columns, and is named after the Pan Flute or Pan Pipes musical instrument, an example of which can be seen in Figure 4.1[1]. While the model features only three columns, their round shape makes

[1]https://tinyurl.com/y3b4exhn, accessed 2020/08/20

them one of the more costly geometries to simulate in CFD, as they require a dense mesh for the small straight edged cells to approximate fluid flow over the round surface. This model provides an interesting test case, balancing a design intended to highlight the effects of abstraction on simulation against a model that is easy to work with due to its size and limited complexity.

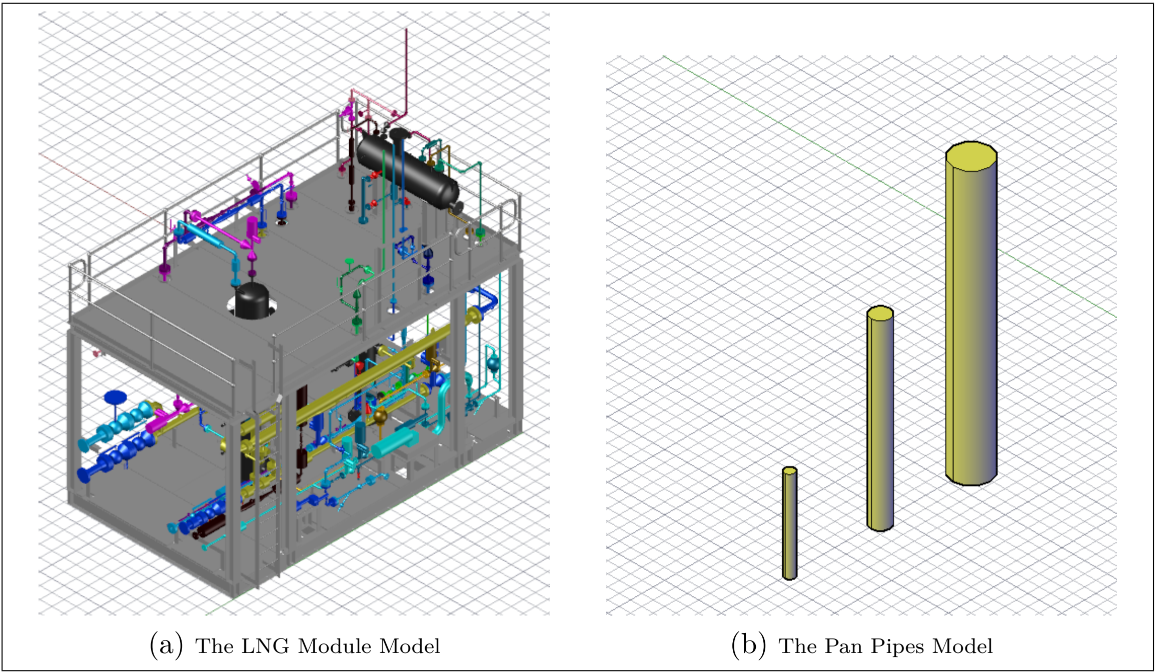

(a) The LNG Module Model (b) The Pan Pipes Model

Figure 4.2: LNG Module beside Pan Pipes

Abstracting from round objects to simple prisms represents one of the most severe forms of Geometry Abstraction, meaning the Pan Pipes provide an opportunity to understand the maximum impact of abstractions in a context that is simple enough to understand and experiemnt with. In order to assess that impact, simulation data is captured from specific locations within the simulation domains that are common across the geometry variants. By comparing this data between simulations, it is possible to measure the variance introduced by different abstractions, which will allow for the assessment of their affect on CFD simulations.

This chapter describes the variations of the Pan Pipes model that were created with LODOS (4.1), details the CFD simulations used to compare the impacts of abstraction (4.2), presents the results from the CFD simulations along with analysis (4.3), and finally discusses the limitations of this test case (4.4).

4.1 Pan Pipe Abstractions

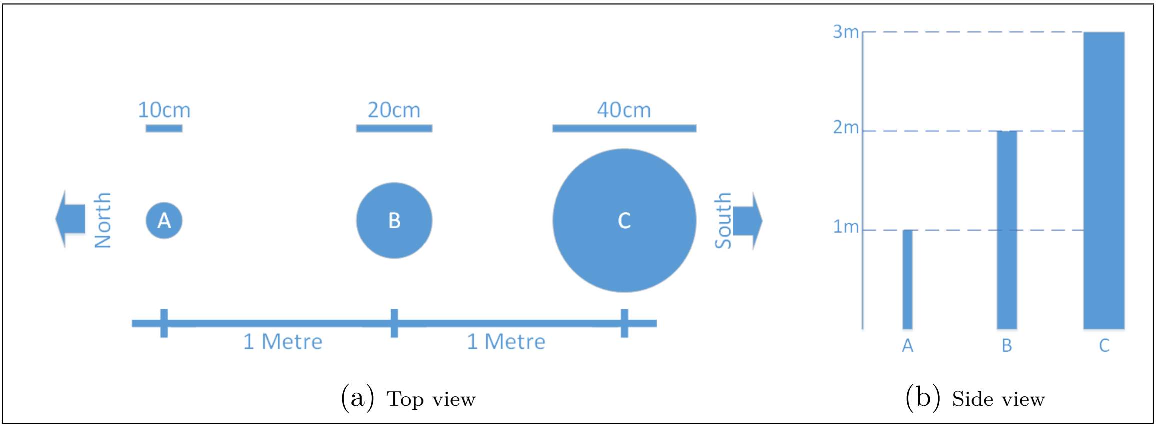


Figure 4.3: The design diagram for the Pan Pipes model

The Pan Pipes model is composed of three cylinders arranged along a central line, as can be seen in Figure 4.3. The first cylinder is 10cm in diameter by 1m high, the second cylinder is 20cm in diameter by 2m high, and the final cylinder is 40cm in diameter by 3m high. The centre point for each of the cylinders fall at 1m intervals from each other, meaning there is a 85cm gap between the bodies of the first two, and a 70cm gap between the latter two.

The difficulty of modelling airflows around cylinders was underestimated when the model was conceived, requiring more resources to mesh and simulate than had been expected. The intention in designing the columns with differing heights and widths is to provide multiple stages of interference for any gas flows, with the change from cylinders to prisms initially expected to have only a minor impact on flows. The multiple stages of interference allows for more interesting mixing and turbulence patterns to form with only a limited number of objects, giving a greater scope for changes in the geometry to impact the overall fluid simulation.

The design was implemented as a Building Information Model within Autodesk® Revit®, to best simulate the workflow required to go from an existent model to a com-

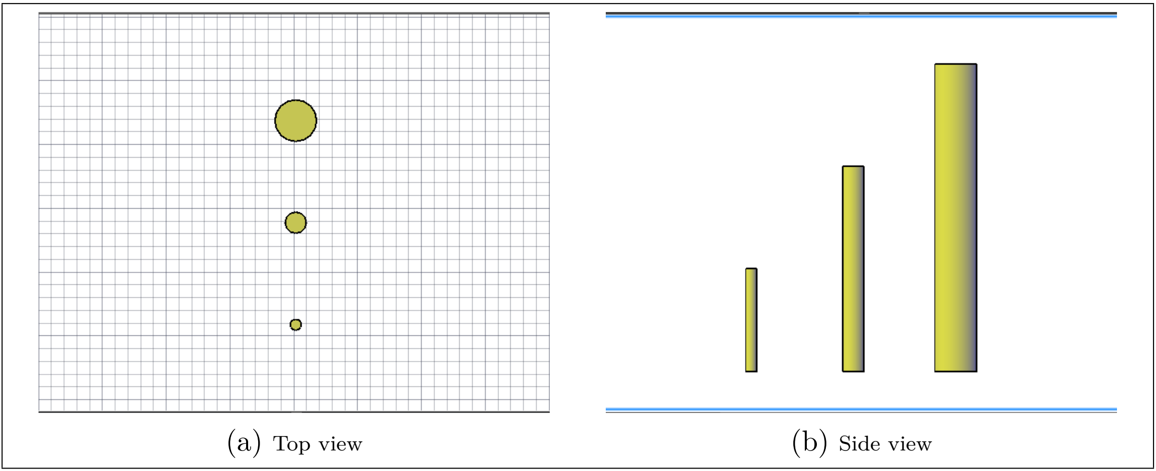


Figure 4.4: The original Pan Pipes geometry (PPO), as depicted in Autodesk® AutoCAD®

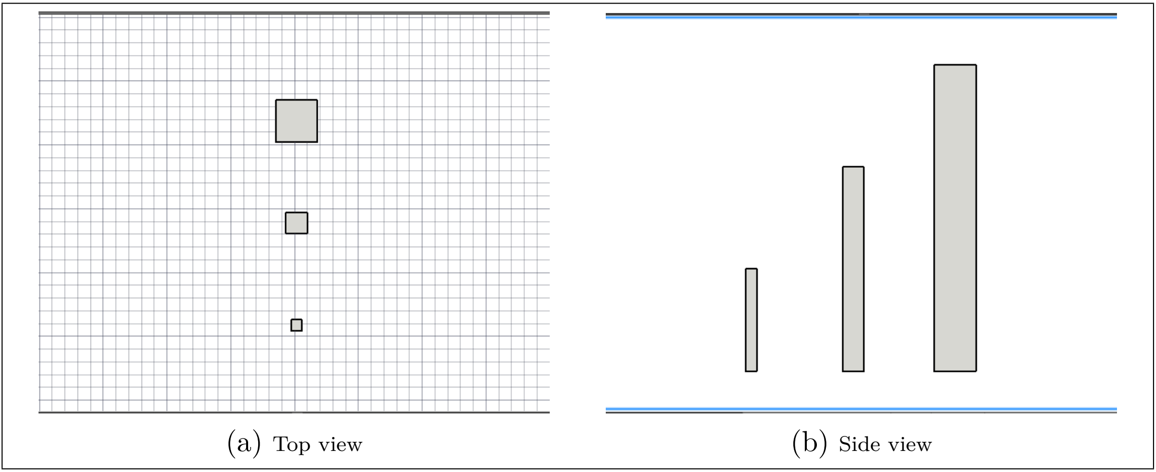


Figure 4.5: The Simplified Pan Pipes geometry (PPS), as depicted in Autodesk® AutoCAD®

pleted CFD simulation. The BIM model is then loaded into Autodesk® AutoCAD® ready for the LODOS plugin to be used upon it, with the original geometry shown in Figure 4.4. Each abstraction of the model generated by LODOS, as well as the original, are exported as legacy Autodesk® AutoCAD® 2010 files to maximise compatibility for importing them into Ansys® Workbench® .

The first abstraction generated was the Simplified variant, as shown in Figure 4.5. Due to the simple geometry of the base model (ie. only three pillars) there is no need to use the hiding or merging algorithms for this abstraction. To generate the Simplified geometry, a 5cm minimum size was selected so all objects would be above the Hide threshold, with all other options left off or at default settings.

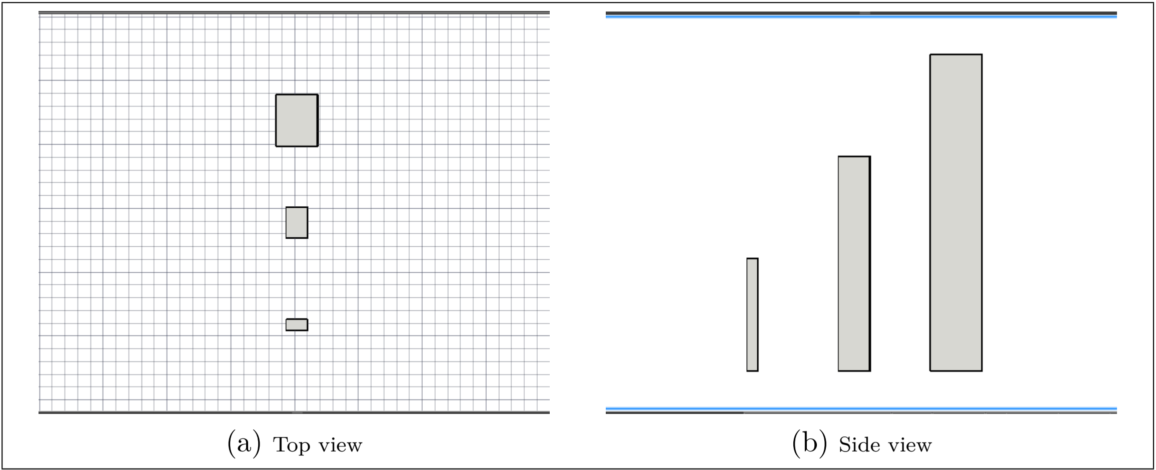


Figure 4.6: The 10cm Cubified Pan Pipes geometry (PPC10), as depicted in Autodesk® AutoCAD®.

The second abstract model generated was a Cubified variant, shown in Figure 4.6. Several different cube sizes were examinded to identify a good size for testing, with a 5cm cube size found to align with the columns, creating an abstraction effectively identical to the simplified variant. To help differentiate the Cubification results from those seen in the Simplification abstraction, as well as to test the impact of imperfect abstraction, it was decided a 10cm Cubification should be used for the comparative simulations. This cube size results in some grid creep, allowing its impact to be observed.

Other abstractions were created, and several even simulated, across the course of the research, but the PPS and PPC10 variants above are best suited for comparing the impact of abstraction on a simulation. Variants where only some objects were abstracted, inline with foreseeable use cases, showed results in-between the two abstractions above and the original, making them less useful for evaluating the margin of introduced error. Variants with larger cube sizes introduce greater levels of error, but that behaviour is easy to predict as the cubes become visually less and less like the original geometry, making their inclusion in results unnecessary.

The primary research aim for this case study is to evaluate the error introduced by abstraction, with the PPS and PPC10 variants offering good comparison points. Comparing and contrasting the data between three sets also strikes a good balance between showing the effect and keeping the information parsable. The following section (4.2) provides details of the setup used to perform the CFD simulations and capture the comparative data.

4.2 CFD Simulation Setup

The CFD Simulations were performed in two rounds; First qualitative, then quantitative. The initial simulations uses an air domain to simulate turbulence from wind flowing around the Pan Pipe variants, with the later simulations adding a methane inlet to observe and measure the impact of abstraction on mixing and dissipation. The same air domain is used for both simulations and settings were kept the same between them where possible, with changes made to the latter simulation set up to support multi-species mixing and thermal energy.

To define the air domain, the dimensions of the pipes were examined and buffers selected. Given a maximum height of 3m, a 4m high air domain was decided upon to give sufficient room for turbulence and mixing without adding too much processing workload. A metre and a half each side of the central line was assumed sufficient to allow expanse and airflow, with a 2m upwind and 6m downwind area to allow the establishment and resolution of flows, see Figure 4.7. While the 6m downwind is too small to allow complete dissipation of all gas, it seemed a reasonable distance based on the dimensions of the Pan Pipes, without performing full vapour dispersion distance simulations to establish the required distance.

To create this domain, each of the geometries were imported into Ansys® DesignModeler® using the process outlined in Section 3.1. A primitive "Cube" measuring 10m x 5m x 8m was added with its property set to fluid, with the Pan Pipe geometry subtracted from it to leave the air domain. The geometry is then saved before being opened in the Ansys® Workbench® default meshing program.

To achieve the final meshing settings, an iterative approach was used with the Pan

Pipes Original geometry as the basis. A set of assumed values for the meshing were input as a starting point, then iterations of the mesh were generated with the mesh density doubled to allow multiple meshes to be compared. This allowed for a mesh independent solution to be found for the original geometry, with those mesh settings then used to mesh the abstractions. The reasoning being that if the mesh settings worked on the original geometry, which is by its nature the most complex, they would work on the abstractions and provide the best grounds for a comparison of performance.

Using this method, the mesh settings shown in Table 4.1 were arrived at.

Sizing			
Size Function	Prox. and Curv.	Rel. Center	Fine
Prox. Size Function	Faces and Edges	Max Tet Size	200.70 mm
Quality			
Smoothing	High		
Assembly Meshing			
Method	CutCell		

Table 4.1: Mesh Settings used for Pan Pipes simulations. Other settings were left as defaults.

Higher order physics were used for the CFD Simulations, to give a more accurate baseline and better grounds for observing variance in the abstracted runs. A Transient simulation was selected as the timesteps would allow measurements and images to be captured at particular time slices for comparison. The CFD Simulation settings of note are listed in Table 4.2.

Species Transport, which calculates the conservation equation for each chemical species in the system, is only turned on for the quantitative simulations. It is unnecessary in the initial round of qualitative simulations, as they only modelled a single fluid/species, air.

Models			
Energy	On	Viscous	SST k-omega
Species*	Species transport		
Solution: Methods			
Scheme	PISO	Skew Correction	1
Neighbor Correction	1		
Gradient	Least Squares	Pressure	SO
Momentum	SO Upwind	Turbulent Kinetic	SO Upwind
Specific Dissipation	FO Upwind	CH4*	SO Upwind
o2*	SO Upwind	Energy	SO Upwind
Transient Formulation	FO Implicit	Warped-Face Grad. Corr.	On
Solution: Run Calculation			
Time Stepping Method	Fixed	Time Step Size	0.03
Number of Time Steps	340	Max Iter./Time Step	100
Reporting Interval	1	Profile Update Interval	1

Table 4.2: CFD Settings used for the Pan Pipes simulations, FO meaning First Order and SO meaning Second Order. Other settings were left as defaults. Settings marked with '*' would have been absent or different in the initial single species simulations.

As mentioned, the three initial simulations are assessed using qualitative methods, namely visual comparison of the captured images. The later round of simulations use quantitative assessment, by comparing measured gas values. To achieve the quantitative readings, a series of "probes" are introduced to the flow domain, with the average methane level measured across their surface. The probes are actually 50cm x 50cm planes, positioned as shown in Figure 4.7, or at:
Probe 1 - 2.25m, 1m, 0m
Probe 2 - 3.25m, 1m, 0m
Probe 3 - 4.25m, 1m, 0m
Probe 4 - 6.25m, 1m, 0m

These probe co-ordinates are given in the form (X, Y, Z), treating the middle of the air domains North edge at ground level as origin (0, 0, 0). North-South is treated as X, Ground-Sky as Y, and West-East as Z with 0 representing the mid line. Looking

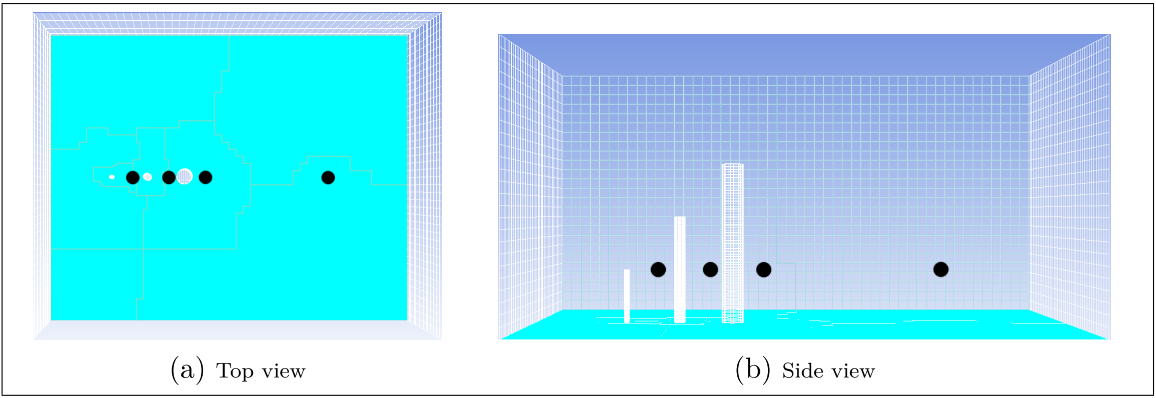


Figure 4.7: The probe locations within the Pan Pipes Domain, denoted by black dots.

at Figure 4.7, the probes can be matched to the above list by numbering from left to right.

The reporting mechanism within Ansys® Fluent® outputs the probe measurements to a delimited text file, which is then imported into Microsoft® Excel® to generate comparative graphs. The results of these simulations are shown in the following section.

4.3 CFD Results and Analysis

The first stage of comparison for the LODOS abstraction system was qualitative. Simple CFD simulations were conducted with air flow moving over the different geometries, with lines added to show the movement of air and false colour to highlight pressure and velocity. These results can be seen in Figure 4.8.

As can be seen from the images, these early results were promising. While changes in air flow and pressure can be observed, particularly along the upwind face of the simplified geometry, the overall shape of the airflow is largely the same. Table 4.3 presents the measurements of the mesh complexity and quality, which shows that the abstracted geometry greatly reduced the number of nodes required for the CFD simulations without adversely effecting the mesh quality.

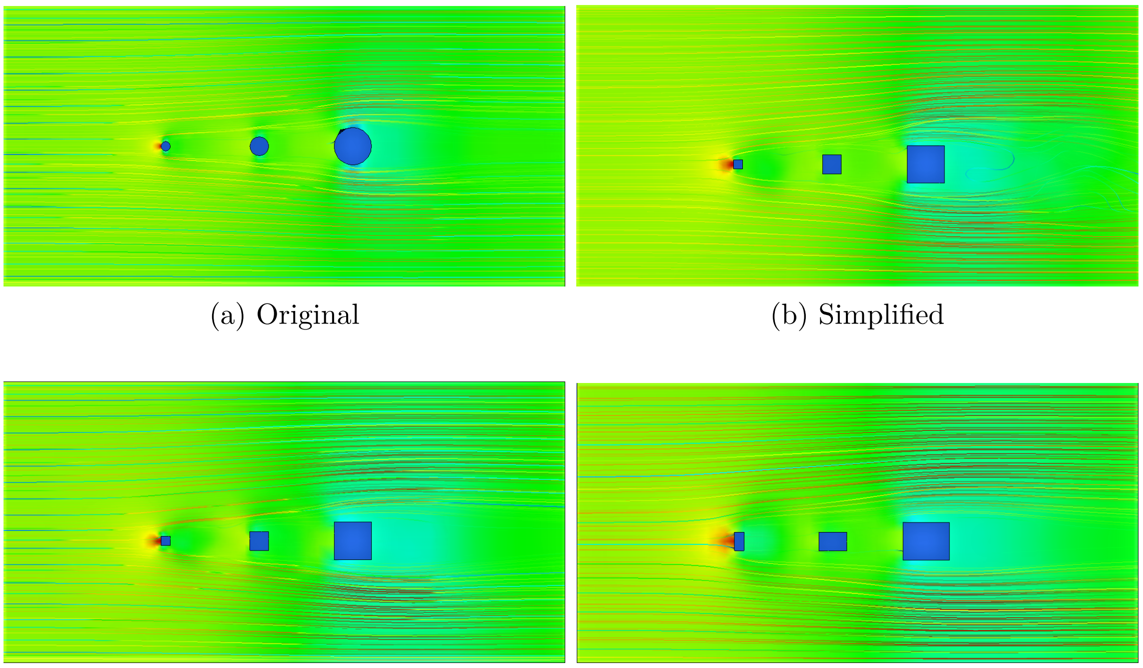

(a) Original (b) Simplified
(c) Cubified - 5cm Grid (d) Cubified - 10cm Grid

Figure 4.8: Side by side visual comparison of flow over the Pan Pipes abstractions

Model	**Nodes**	**Elements**	**Skewness**	**Orthogonal**	**Runtime**
Original	295,167	278,452	0.016	0.96	61 min
Simplified	65,422	58,702	0.004	0.99	21 min
5cm Cubes	38,788	32.931	0.008	0.964	-
10cm Cubes	97,946	91,088	0.006	0.966	22 min

Table 4.3: Statistics for Pan Pipes model at different levels of abstraction. The Skewness and Orthogonal values given are the average.

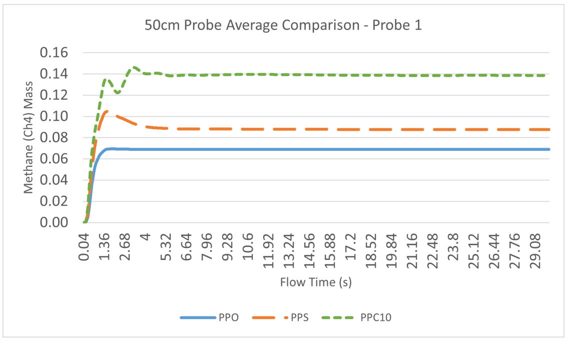
Based on qualitative analysis of these results it seems likely that abstraction can be used to reduce computational costs while maintaining a fairly similar flow profile. To confirm this result and measure the impact of abstraction more precisely, a variation to the qualitative simulations was devised. A single phase simulation utilising a methane vapour cloud was run with measurements of the methane concentration at specified points used for comparison.

Going into these simulations, it is important to be mindful of the susceptibility of CFD to Chaos Theory; Even small variations can result in large changes after propagation. To mitigate this, the methane levels were measured using averaged surface planes instead of single points. 50cm a side planes were created, giving a 250cm^2 surface for values to be averaged across; large enough to mitigate propagation variance while being small enough to capture any significant change in flow.

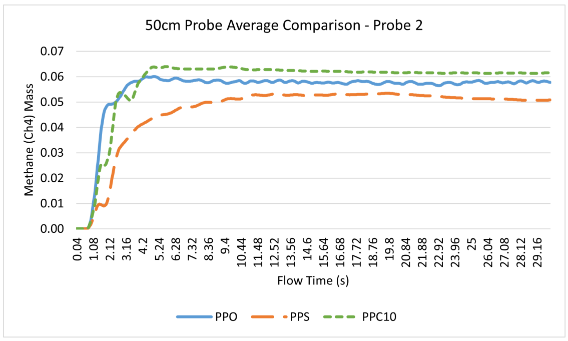
The methane is introduced from the top of the first pillar, to allow mixing and movement around the second and third. The surface averaged methane values were captured from the probe planes and output to a text file for each time step. The resulting values were then graphed, as can be seen in Figure 4.9.

The Probe 1 results, those from the plane closest to the methane inlet, show that both abstracted geometries have higher measurements than the original geometry. This suggests reduced dissipation, most likely due to the area of reduced velocity caused by the abstractions of pillar a. While the closest of the two, the PPS results are about 25% higher than the original. In vapour cloud simulations an over estimation is generally preferred because it is better to be overly cautious when dealing with safety, so this over estimation by the abstractions is notable but does not immediately rule out the use of abstracted geometry in simulation analysis.

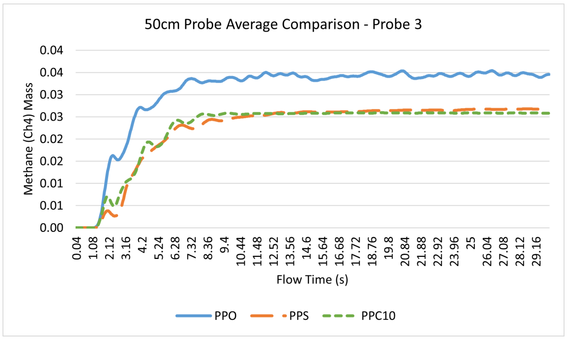
While the abstracted readings do show some early peaks, both the PPS and PPC measurements from Probe 1 taper off towards a stable value, in-line with the results from the original geometry. It is not surprising that the PPC results show the most variance, given that the PPC abstraction features grid creep, which would also have



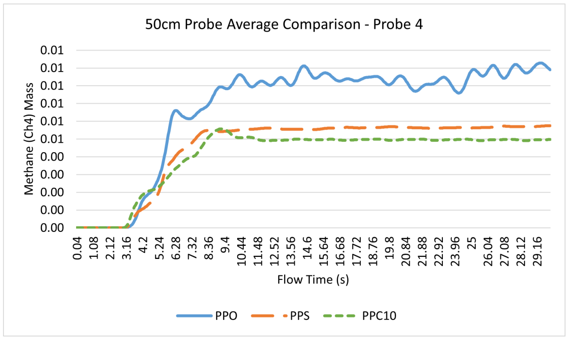
(a) Probe 1 results



(b) Probe 2 results

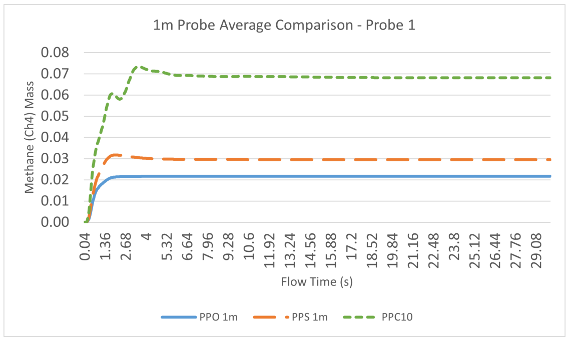


(c) Probe 3 results

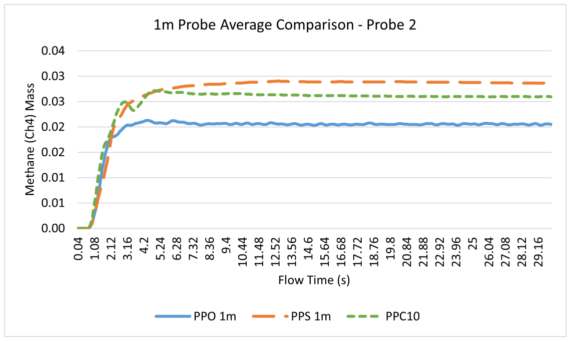


(d) Probe 4 results

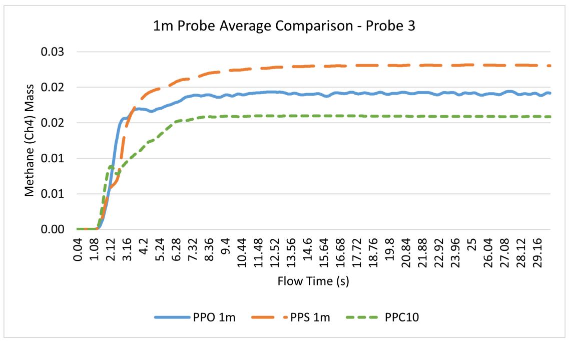
Figure 4.9: Graphed values from the methane probes. PPO represents the Pan Pipes Original geometry, PPS Simplified and PPC the 10cm Cubified variant.



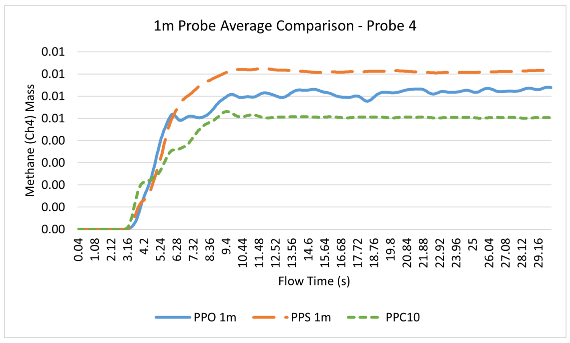
(a) Probe 1 results



(b) Probe 2 results



(c) Probe 3 results



(d) Probe 4 results

Figure 4.10: Graphed values from the 1m methane probes. PPO represents the Pan Pipes Original geometry, PPS Simplified and PPC the 10cm Cubified variant.

altered the shape of the methane inlet based upon the top plane of the first tower. While the Probe 1 readings show more variance than originally expected, the measured methane levels are already quite small, which would compound the ability for small changes to cause disruption.

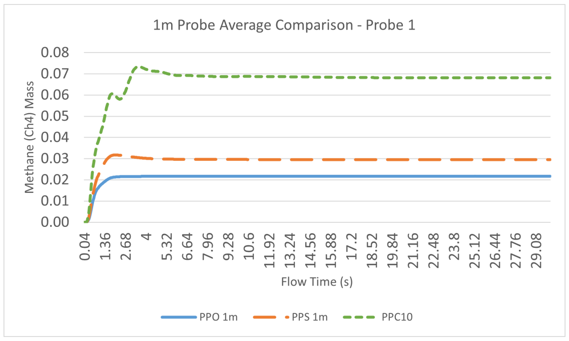
Looking at the overall probe readings, the PPS results are typically closest to the PPO readings, which might be expect given it has the most similar geometry. While there is certainly disparity between the values seen at each probe, the overall shape of the PPS results show marked similarity to the baseline simulation. The PPC results show a greater variance compared to the PPO, but still have some familiarity in terms of behaviour and value. For instance, looking at the Probe 2 results around the 1.95s mark, all three results have a peak followed by a dip and subsequent peak.

It is interesting to note that while the methane levels measured at Probe 1 were higher for both abstractions, the measurements at subsequent probe locations tended to be lower. This result was counter intuitive enough that it prompted the simulation to be re-examined, as while some variance between the original simulation and abstraction simulations is expected, it was expected the results within the abstractions would be consistently higher or lower. While examining the simulation setup it was noted that the probe locations placed them relatively close to the pillars, which the qualitative simulations have shown create a partial dead zone down wind from their north-most face.

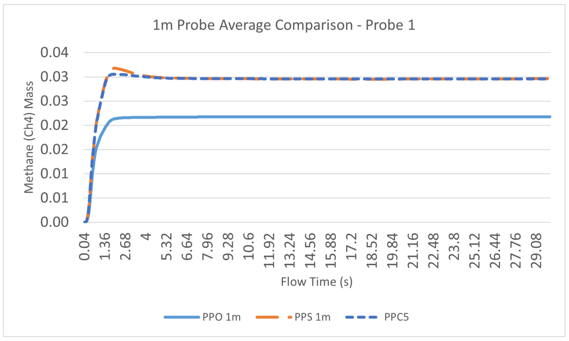
Logically, similar dead zones would be present in the methane simulations, causing the vapour cloud to spread out further than it did when moving around the cylindrical pillars. Given pillar C is 40cm, with a plane of 50cm it is likely only 10cm of the probe is directly exposed to the methane flow at Probe 3. This realisation lead to re-running the simulations using 1m x 1m planes as the probe surfaces, giving the results in Figure 4.10.

It is interesting to note the disparity at probe 1 in these new results with regards to the PPC data. This can most likely be explained by the wider top surface of the Cubified pillar A, causing the methane to be released over a larger area and thus more of it being captured at the first plane with its larger area. At the later probes, once the vapour flows have had more of a chance to stabilise, the quantities of methane being captured average to values much closer to those seen from the Original and Simplified abstractions.

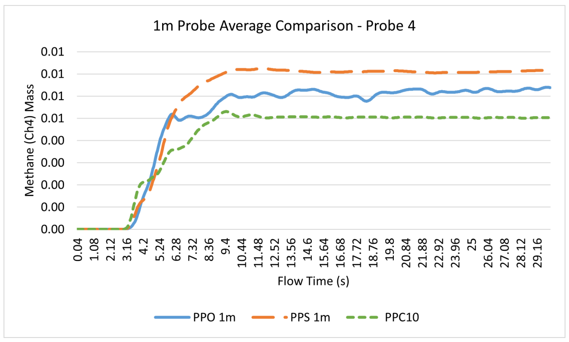
Continuing to look at the Cubified abstraction, similar to the initial results from the 50cm Probes, the values detected at Probes 3 and 4 are actually lower than the values registered in the Original simulation at the same points. This can be explained by the dead zones created by the Cubified geometry, given its grid creep, and demonstrate the risk when using a poorly fit Cubified abstraction with grid creep. While the Cubified abstraction provides the most straight forward way to generate CFD ready geometry, it can also introduce a level of unpredictability to the results, requiring a user to consider their Probe size and location, or use a smaller cube size to reduce or eliminate grid creep.



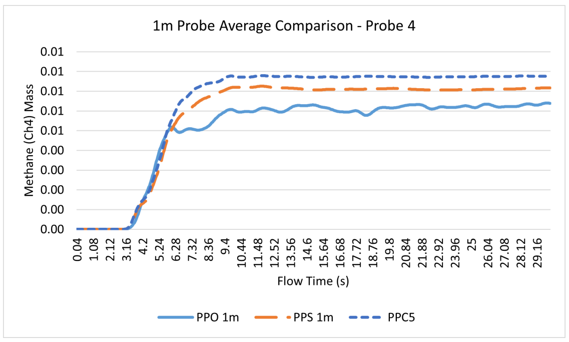
(a) Probe 1 results - 10cm Cube



(b) Probe 1 results - 5cm Cube



(c) Probe 4 results - 10cm Cube



(d) Probe 4 results - 5cm Cube

Figure 4.11: A comparison of simulation results at 1st and 4th Probes between 5cm and 10cm Cubification methods, Original and Simplified data included for comparison

When a 5cm cube size is used during Cubification, the results are nearly identical to those from the Simplified abstraction for this geometry, as shown in Figure 4.11. While the use of Cubification with extremely small cubes (E.g. 1cm or 1mm) seems like it could be the the best overall solution, capturing the geometry to a high degree of fidelity while also having a known minimum size, Cubification is a much more costly process in terms of computation and memory when compared to Simplification. On a larger model, measuring roughly 100 meters a side, performing Cubification with a cube size less than 10cm took well over an hour, with a cube size of 1 or 2cm causing a system with 64GB of Ram to run into memory issues. As LODOS is only being used in a research context there is likely room to improve the performance of Cubification, but the need for it to map data to a grid means it will always require more memory and processing than Simplification, making it unlikely it will ever allow kilometre long LNG trains to be abstracted into millimetre sized cubes.

Looking at the Simplified abstraction results, the 1m Probes show methane values that measure universally higher than in the Original data set. The values also tend to be quite stable, with the Simplified Probes at 1 through 3 stabilising around 0.03 CH4 Mass, while the Original geometry stabilises around 0.02. The values measured at Probe 4 are all noticeably lower, which is expected given the extra distance from the Methane outlet and thus time the vapour has had to dissipate.

The fact the Simplified abstraction generates reliably higher readings when using an appropriately sized plane supports the idea that abstracted geometry can be used for CFD based analysis. The overall behaviour of the readings is very similar between the Simplified and Original simulations, with the probes beginning to detect methane and reaching their relative peaks at very similar time intervals. The Simplified data does tend to be smoother, showing less of the subtle oscillations that the Original simulation shows, which can be attributed to the combination of the flow barrier caused by the flat North faces of the prisms, a prism being less likely to introduce an unstable eddy, and the smaller number of Nodes present.

4.4 Discussion

Computational Fluid Dynamics Simulation is a complex field, where simulated recreations of experimental scenarios with a margin of error between 10% (Sun *et al.*, 2014) and 25% (Gopalaswami *et al.*, 2017) are considered acceptable. These margins of error are reported in the literature by experienced CFD researchers, working from well documented test scenarios that were conducted for the purpose of providing ground truth data for evaluating CFD codes and methodology against. So even in situations where experienced CFD practitioners have a well documented scenario to recreate, the limitations of the available hardware, software, physics models and resources can lead to non-trivial margins of error.

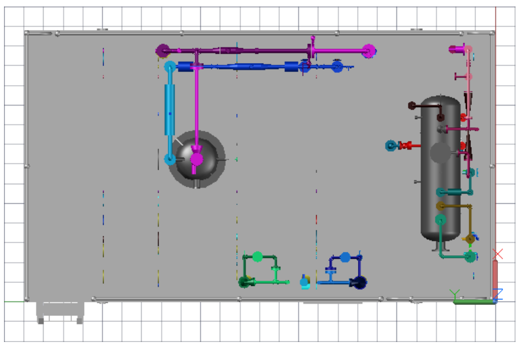
The work with the Pan Pipes test case described in this chapter is valuable both in developing a CFD Simulation work pipeline, and to explore the impact of the proposed solutions on a simulation. It allows reasoned assumptions to be made about the margin of error a well formed abstraction is likely to introduce, shows that measured results from abstractions are likely to be reliably higher, and highlighted the greater level of error introduced by a poorly formed abstraction, but it remains a single model with a small number of objects. Performing further simulations with a selection of models of increasing complexity, each simulated at various levels of abstraction, would add confidence to the findings from this test case.

Looking at the results from the simulations and comparing the values between the Original data set and the abstractions, there seems to be sufficient data to support the use of abstracted geometry in CFD Simulations for some use cases. There is admittedly more variance between the Original data and the abstractions than was originally expected, but the Simplified and 5cm Cubifications both show sufficient conformity in their results compared to the Original to make them indicative, provided a margin of error is assumed. This supports the idea that in situations where the original geometry cannot be simulated for whatever reason, an abstracted variation can produce useful data.

The next chapter examines a much more complex model, the LNG Module. This model provides a test case that examines Geometric Abstraction in terms of a foreseeable real world application, while also allowing the ability of LODOS to scale to larger problems to be examined.

Chapter 5

Case Study: Complex Geometry


(a) Top view

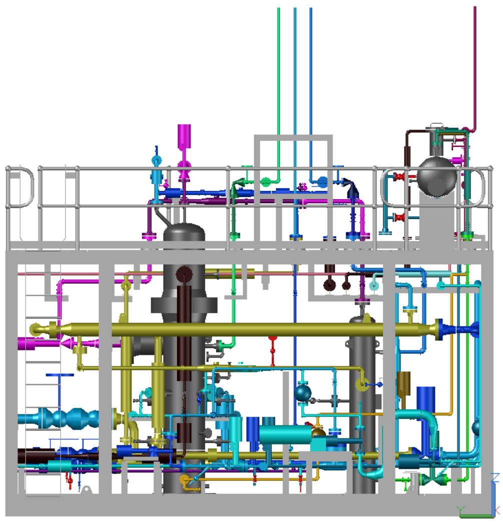

(b) Side view

Figure 5.1: The original LNG Module geometry (LMO), as depicted in Autodesk® AutoCAD®.

The LNG Module is an interesting challenge. The original geometry, shown in Figure 5.1, can not be easily modelled within CFD software. While the geometry can be imported into the Ansys® Workbench® meshing software, it is not possible to successfully mesh the unaltered geometry.

There are several reasons for this incompatibility. The first is the sheer number of objects and the geometric complexity present in the model is difficult for the Ansys® Workbench® software to deal with, causing issues such as poor responsiveness and missing elements when importing and meshing the geometry. Secondly, the LNG Module contains non-standard geometric shapes, such as an element where two cones connect at the tip, shown in Figure 5.2. These non-standard shapes rely on custom 3D objects which are not natively supported by the Ansys® Workbench® meshing software, resulting in warnings about non-manifold bodies.

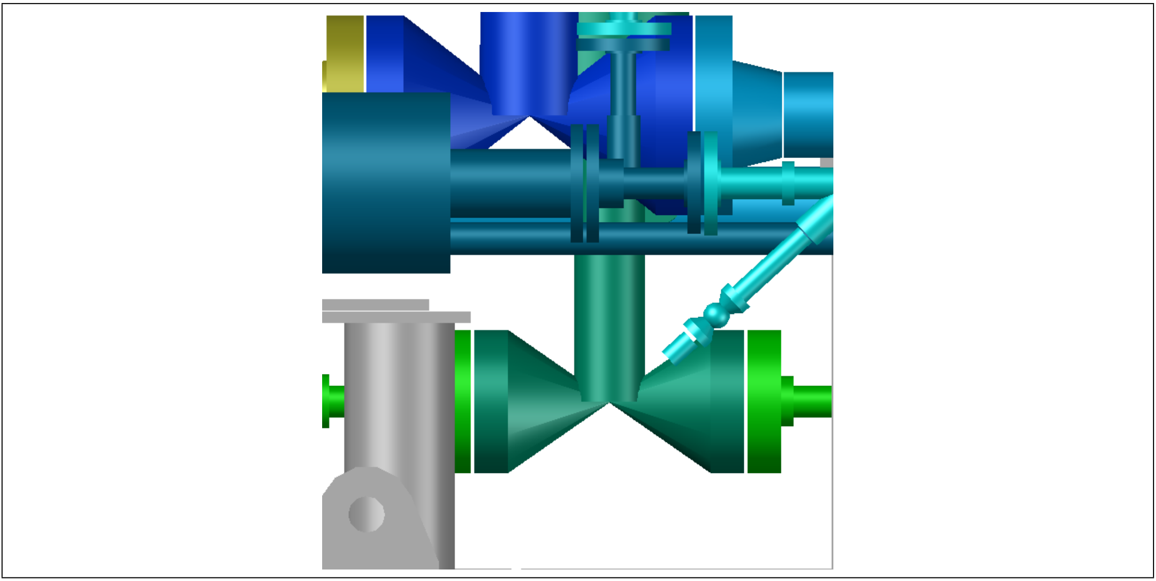


Figure 5.2: Some of the non-standard geometry that presents issues when attempting to mesh the LNG Module, note the cones meeting at their tips.

When these items are replaced or removed, the model still suffers from issues due to minor gaps between some objects. As an example, the central column is comprised of several cylinders, with two used to represent a flare in the pipe where it is bolted together to form a pressure seal. When the designer modelled this they left a 1mm gap between the two surfaces, either by accident or to allow for machining and fit tolerances, which is virtually undetectable to visual inspection of the model due to the size of the pieces in question.

Attempting to mesh this gap, and several similar small gaps that appear elsewhere, causes numerous problems. The gaps cause the meshing process to take longer, as the software attempts to resolve the small gaps, with the meshing process eventually either failing or producing a mesh with very low quality metrics due to bad cells such as slivers. Between these problems and the issues with non-standard geometry, despite numerous attempts to correct the mesh by hand or repair it automatically using various software packages, no functional mesh of the original LNG Module geometry could be generated.

While an inability to compare CFD Simulations of abstracted LNG Module geometry to the original makes it harder to assess the impact of the system on complex geometry, it does provide an opportunity to test the ability of LODOS to enable the simulation of a model that would otherwise be impractical. Without an original simulation for comparison, the results from the Pan Pipes simulations will provide the margin of error assumed for the results presented later in this chapter. While the complexity of the LNG geomtry is considerably greater than the Pan Pipes, the level of introduced error they established should still hold true.

With regard to the LNG Module model itself, it is an accurate BIM representation of a compressor module from a modular LNG train. It was made available for use in this work by the Australasian Joint Research Centre for Building Information Modelling at Curtin University, with thanks and credit to Woodside Energy for the original model. This makes it an excellent "real world" test case for LODOS, with the model having a high degree of geometric complexity due to the numerous pipes, supports and mechanical items present.

This chapter presents the two LODOS abstractions that were generated from the LNG Module (5.1), provides details on the CFD simulations used to examine the abstractions (5.2), presents the results of the simulations along with analysis of what they represent (5.3), and then discusses the successes and limitations of the test case (5.4).

5.1 LNG Module Abstractions

To keep simulation results from the LNG Module abstractions comparable to the Pan Pipes results, similar abstractions were needed. This required creating a Simplified variation of the LNG Module as well as a Cubified variant. Due to the much more complex nature of the geometry present in the LNG Module it was necessary while generating the Simplified variant to use additional functionality that the Pan Pipes abstractions did not require.

While working with the LNG Module there was a large number of meshing failures,

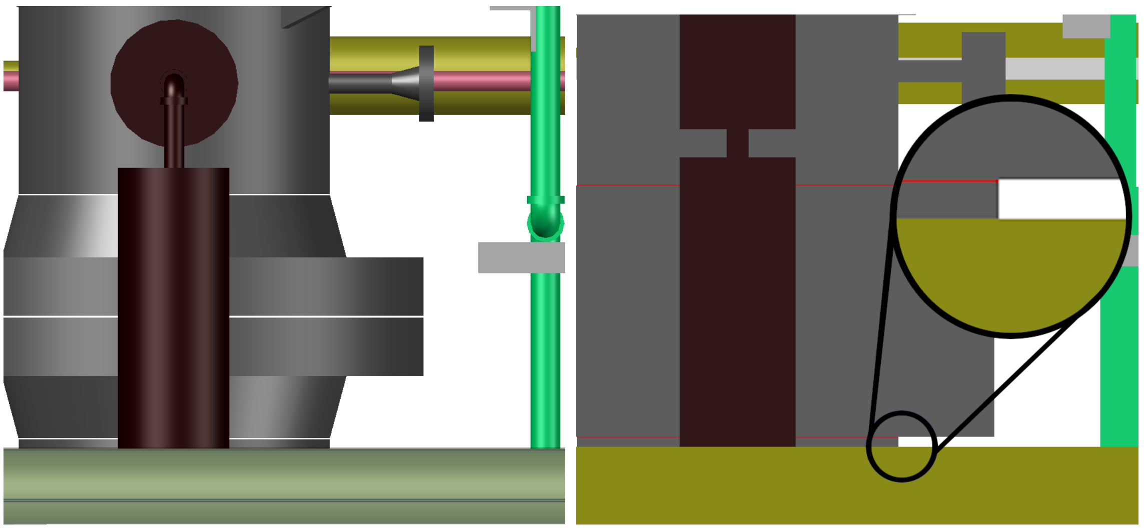

(a) Original Geometry (b) After Simplification and Gap Fill

Figure 5.3: A section of LNG Module Geometry before and after the use of Simplification and Gap Fill. The thin red lines are the added gap fill segments, see the enhanced portion of (b) for a better look.

and the successful meshes had poor quality metrics due to a large number of slivers. From examining the model, it was identified that the slivers were being caused by a series of small gaps present between a few items of geometry within the model, such as two halves of a pressure valve. From informal discussions, such gaps can be included to allow space for the addition of items such as rubber seals, or because they allow for the items to be secured manually by tightening of bolts, both of which would not be possible with perfectly machined zero gap parts.

Due to these very small gaps causing meshing issues, a new feature was added to LODOS. This feature, **Gap Fill**, identifies objects within the model that feature adjacent planes within a user specified distance of each other. It then generates a new object to fill that space as can be seen in Figure 5.3, preventing small misalignments or gaps in the geometry from disrupting the meshing process.

Due to the the LNG Module featuring overlapping geometric shapes such as pipes with insulation sleeves, it is also necessary to use the merge function described in 3.3.2. To briefly reiterate, this function identifies overlapping objects with similar dimensions and replaces them with a single object featuring the maximum size characteristics of both. This process is useful when dealing with models with high complexity such as the LNG Module, as the Merge process can greatly reduce the number of individual objects, makeing importing and working with the model in Ansys® Workbench® much easier.

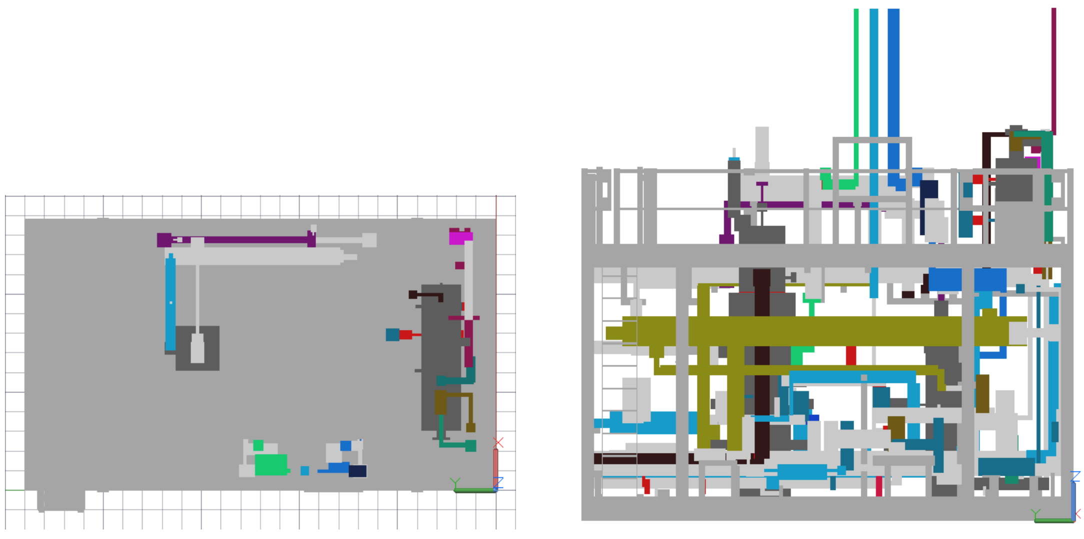
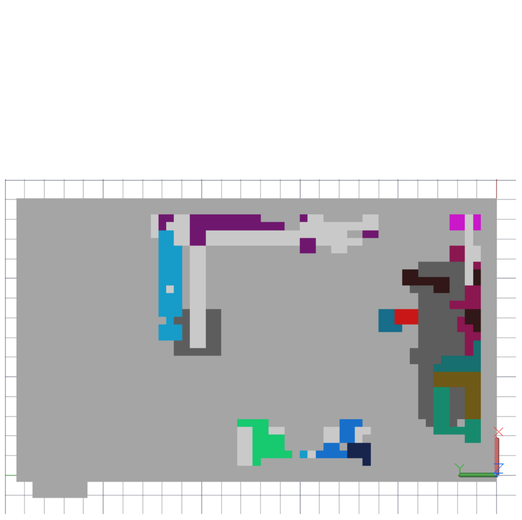

(a) Top view (b) Side view

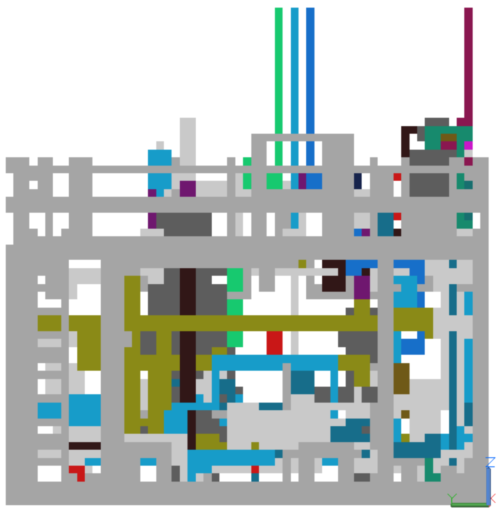
Figure 5.4: The Simplified LNG Module geometry(LMS), as depicted in Autodesk® AutoCAD®.

With these additional functions applied, the Simplified abstraction of the LNG Module shown in Figure 5.4 was generated. LODOS was instructed to use at least 5cm on any one dimension for the Hide algorithm, followed by a 5cm margin for the Merge algorithm, with a 5mm distance specified for the Gap Fill algorithm. These settings were arrived at through iterative applications of LODOS, providing a good balance between removal of small elements while maintaining overall shape.

Figure 5.5 shows the Cubified abstraction of the LNG Module, generated using a 10cm cube size. Cubification applies the colour of the underlying geometry to generated cubes, making comparing the abstractions to the original model much easier, even if some colours are lost as objects are merged. The flat surfaces present in both abstractions can be visually difficult to understand when represented in 2D, the lack of shading making it difficult to see the geometric complexity present even after the abstractions.

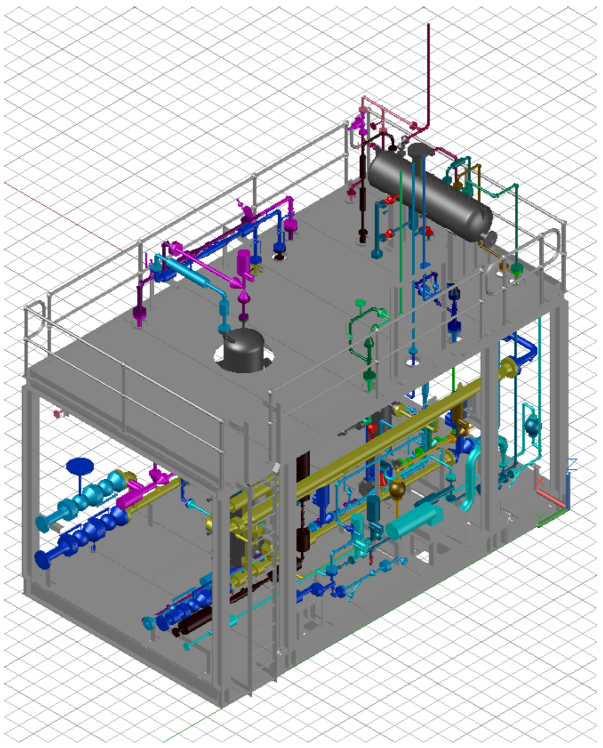


(a) Top view

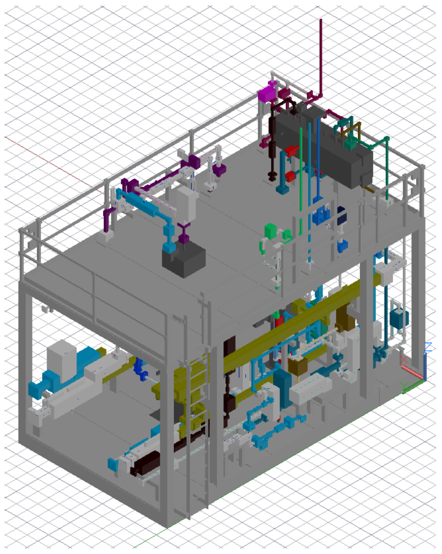


(b) Side view

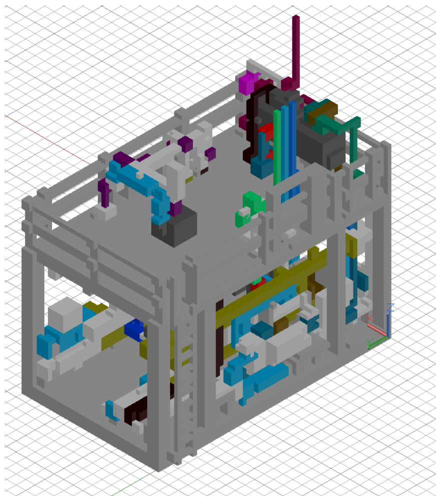
Figure 5.5: The Cubified LNG Module geometry (LMC), as depicted in Autodesk® AutoCAD®.



(a) The original LNG Module geometry



(b) Simplified LNG Module



(c) 10cm Cubification LNG Module

Figure 5.6: An orthogonal view of the LNG Module geometry from within AutoCAD, along with the same view of the two abstractions.

The orthogonal view, visible in Figure 5.6 for the original geometry and both abstractions, can provide a better sense of the spatial complexity of the geometry. It can be seen that both the abstractions remove the gaps in the top plates that allow the condensing tower and pipes through. Comparing the two abstractions, the 10cm Cubification retains less of the complexity and nuance of the original geometry, with a reasonable amount of volume that is empty in the original and Simplified geometries filled when using 10cm Cubes.

5.2 CFD Simulation Setup

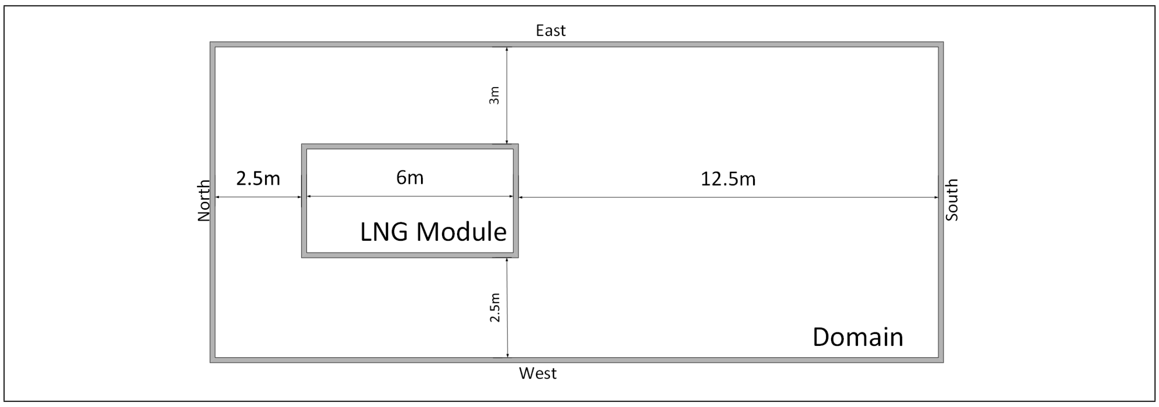


Figure 5.7: A diagram showing the layout of the domain for the LNG Module

The process described in Section 3.1 was followed when setting up LNG Abstractions for meshing, that is; Import the geometry, reorient it, and create a fluid domain around it using a primitive. The dimensions of this domain are 21m long by 8m high by 9m wide, with the LNG Module abstractions placed near the air inlet at the 'North' end, as can be seen in Figure 5.7. To simulate an LNG leak caused by material failure within the LNG module, a section of geometry that mapped to the North most face of the central column was selected as the methane inlet, a face common to both abstractions though its dimensions varied slightly between the two.

As discussed, it has not been possible to perform a CFD Simulation using the original geometry. For this reason, unlike when working with the Pan Pipes in 4.2, it was not possible to establish comparative mesh settings for use across all abstractions.

Instead the settings identified when working with the Pan Pipes geometry were used as a basis to make results between those simulations and these more comparable, with the settings modified as necessary for the LNG Simplified geometry.

Sizing			
Size Function	Prox. and Curv.	Rel. Center	Medium
Prox. Size Function	Faces and Edges	Min. Size	100mm
Prox. Size	100mm	Max Tet Size	400 mm
Growth Rate	1.5		
Quality		Smoothing	High
Assembly Meshing		Method	CutCell
Face Sizing			
Scoping	Geometry Selection	Geometry	2729 Faces
Definition Type	Element Size	Def. - Element Size	400mm
Face Sizing - Advanced			
Defeature Size	1mm	Size Function	Prox. & Curv.
Growth Rate	1.5	Local Min. Size	10mm
Prox. Min. Size	10mm	Cells Across Gap	1
Prox. Size Func. Sources	Faces & Edges		

Table 5.1: Mesh Settings used for LNG Module simulations. Other settings were left as defaults.

Using this method, the mesh settings in Table 5.1 were arrived at. Upon initial abstraction the Simplified LNG geometry contained several sub-millimetre edges, likely due to either features from the original geometry or offsets introduced by the abstraction process. These tiny edges resulted in poor mesh metrics, so Ansys® SpaceClaim® was used to adjust the domain geometry to have no edges less than 1mm to resolve the issue. Once a mesh of reasonable quality could be generated, a CFD simulation was set up using the settings listed in Table 5.2.

Models			
Energy	On	Viscous	SST k-omega
Species	Species transport		
Solution: Methods			
Scheme	PISO	Skew Correction	1
Neighbor Correction	1		
Gradient	Least Squares	Pressure	SO
Momentum	SO Upwind	Turbulent Kinetic	SO Upwind
Specific Dissipation	FO Upwind	CH4	SO Upwind
o2	SO Upwind	Energy	SO Upwind
Transient Formulation	FO Implicit	Warped-Face Grad. Corr.	On

Table 5.2: CFD Settings used for LNG Module simulations, FO meaning First Order and SO meaning Second Order. Other settings were left as defaults.

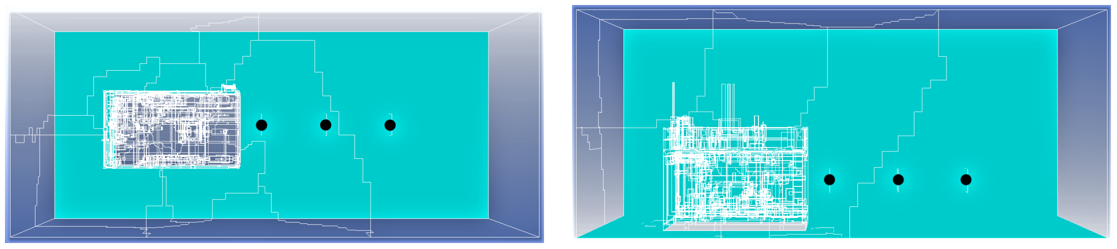

(a) Top view (b) Side view

Figure 5.8: The probe locations within the LNG Module Domain, black dots added for emphasis

Figure 5.8 shows the location of the methane probe planes, which are 100cm a side like the ones used in the Pan Pipes simulations. Data was also captured using 50cm x 50cm planes, but unlike in the Pan Pipes simulations the results show nearly identical behaviours across both the 50cm and 100cm plane averages. The geometry and abstractions in the LNG Modules did present their own challenges, which will be discussed in the results and analysis section (5.3).

As with the Pan Pipes probe co-ordinates, locations are given in the form (X, Y, Z), treating the middle of the North edge at ground level as origin. North-South is treated as X, Ground-Sky as Y, and West-East as Z with 0 representing the mid

line. Using these references, the LNG Module probes were located at:
Probe 1 - 10m, 1.5m, 0m
Probe 2 - 13m, 1.5m, 0m
Probe 3 - 16m, 1.5m, 0m

The same methods for capturing the probe readings and graphing them are used for the LNG Module as was used with the Pan Pipes. Namely, the built-in reporting function in Ansys® Fluent® was used to capture the surface averaged values and output them to a text file, with Microsoft® Excel® used to plot and visualise the data. Due to the complexity of the simulations, they were run against the local computing cluster available within Chemical Engineering in the West Australian School of Mines at Curtin.

The local cluster is composed of 8 Nodes, each accessible via a PBS based queueing system. Each Nodes has two Intel Xeon E5-2680 2.7Ghz processors with 8 cores each and 64GB of RAM, giving 16 logical cores per system with Hyperthreading. For initial simulation of the LMS and LMC10 abstractions, 16 cores were used on each of three Nodes, for a total of 48 logical cores.

Using this system, simulating the LMS abstraction with 3 compute nodes for 30 seconds of flow time, using 100 steps per second for a total of 3000 time steps, took 18 hours and 16 minutes. Simulating the LMC10 abstraction for the same length of flow time using the same time steps took 11 hours and 52 minutes. The 10 millisecond time step intervals were chosen to assist the LMS Simulation in resolving its initial flow state, as larger time steps resulted in turbulence runaways leading to simulation failure.

After 30 seconds of flow time the values observed at the probes had not stabilised, prompting running the simulations again over longer flow intervals to ensure the flows are able to properly resolve. While examining this issue, mistakes in the simulation setup were identified, meaning the results would not be comparable to the Pan Pipes results as intended, or even between the LMC10 and LMS simulations.

When running the Pan Pipes simulation, a User Defined Function (UDF) was used to vary the inlet velocity based on height. At some point during the simulation iterations, this function was left out during the setup of a LMS simulation, meaning the results from the LMS simulations are not directly comparable to the LMC10 simulations which did include the UDF.

Consideration was given to re-running the LMC10 simulation without the UDF, due to the greater computational resources required to re-run the LMS simulations. This would have introduced a disparity in the results when comparing Pan Pipes and LNG Modules, unless the Pan Pipes simulations were also re-run without the UDF function. It was decided that it would be best to re-run the LMS simulation with the UDF function so that all the simulations would be as complete as possible while also being comparable.

Running LMS simulations with longer flow times while keeping the 0.01 time step would require substantial computational investment, but the complex flows in the LMS simulation tended to become unstable if not given small time steps at the start of simulation. To resolve this conflict, a journal file was produced that runs the initial second of the simulation using the same 0.01 of a second time step as the previous simulation, then switches to progressively longer time steps for the remainder, using a modified output interval within the reporting function to keep the output data consistent. Using this method, the reported values are tending to stable values after 40 seconds of flow time.

The information in Table 5.3 provides details on the time intervals used for the final simulations, and the journal file used for LMSimp can be found in Appendix A.

Time Step	Iterations	Equivalent Flow Time	Reporting interval
0.01	1000	10s	10
0.02	1000	20s	5
0.05	600	30s	2
0.1	600	60s	1
Total	3200	120s	-

Table 5.3: The size of the time steps, number of steps at that length, total flow time at those intervals and the reporting output.

5.2.1 The Challenge of Finding a Mesh Independent Solution

As covered by several papers in the literature, it is best when working with a CFD Simulation to identify a mesh independent solution. As discussed in Chapter 2.2, a coarse mesh can impact the results from a CFD Simulation, so establishing a mesh independent solution ensures the best simulation outcome. A simple method for doing this is to double the mesh density of a "parent" mesh and compare the simulation results against the denser "child" mesh, which Luketa-Hanlin *et al.* (2007) described as a "Grid Study".

Once the simulation results from the child mesh match the parent mesh, the solution is considered mesh independent, and future simulations can use the parent mesh with confidence it is not impacting the simulation. When dealing with meshes that are already computationally expensive this process becomes more challenging, as going from a 10 x 10 mesh with 100 cells to a 20 x 20 mesh with 400 cells is still computationally cheap to simulate. If the initial mesh is 1000 x 1000 however, then the resulting 1,000,000 cell mesh is already computationally costly to simulate, making a mesh independent solution prohibitevly expensive to find as each subsequent child could take days of computing time to assess.

In order to identify a mesh independent solution for the LNG abstractions, alternative methods which are less resource prohibitive were explored. First, the simple method used with the Pan Pipes was tried, simply reducing the maximum sizes of the faces used to generate the mesh while also allowing a smaller minimum value. This method did not work, with the adjusted values either failing to produce a mesh or producing one with very poor quality metrics. These difficulties can be traced to the sheer complexity of the LNG Module, even once Simplified, particularly before GapFill was implemented. Attempts to smooth and repair the initial geometry through a variety of methods were examined, but the scale and complexity of the geometry even in an abstracted form made it costly to attempt to mesh.

Some attempts to generate a suitable mesh took several hours of processing only to fail, or create a mesh with tens of millions of cells which often still had poor overall quality measurements. Examination of these meshes showed that 99% of the cells in the mesh were of good quality, but a small number of cells in areas of particular complexity were throwing out the overall mesh quality and rendering the meshes unable to be simulated. A great deal of time was spent iterating on mesh settings known to produce functional meshes in attempts to create higher density meshes that could be used to test the level of mesh dependence, with no success.

Next, the use of Ansys® Fluent® to improve the mesh directly was explored, using functions within Ansys® Fluent® to identify cells of low quality or high activity and increase the cell density dynamically in those areas. It was hoped these methods would not only allow for identifying a mesh independent solution, but produce an asymmetrical mesh which would concentrate computational resources on the areas of interest, giving the best possible solution for the least possible computing time. Assisted by a computing background and understanding of scripting, journal files were produced that made this process semi-autonomous, proceeding through several iterations of the mesh and applying optimisations at intervals.

These meshes showed odd characteristics, the cell density increasing in some areas, but remaining unchanged in areas adjacent or even between areas which had increased. To establish if the dynamic optimisation process was for some reason simply ignoring these areas, a new journal file was created that simply used the functionality within Ansys® Fluent® to increase the cell density equally across the entire mesh. When these meshes were examined, they showed the same odd characteristics of the cell density increasing in most of the mesh, but specific areas being left unaltered.

The unaltered areas were examined and it was discovered that the mesh in those locations was composed of Polyhedral cells. Polyhedral cells provide several technical advantages over tetrahedral cells and are generated by merging several tetrahedral and/or hexahedral cells. However, because they result from complex computational merging of other cells, they often have unusual shapes and cannot be easily split back into their original form.

This means that within Ansys® Fluent® , once a Polyhedral cell has been created, it cannot be altered. Further examination of this issue and its cause led to documentation by Ansys® explaining that the Cut Cell Assembly meshing method, which has been used throughout the Pan Pipes and LNG Module meshing, introduces Polyhedral cells automatically as a way to help it achieve a higher quality hex dominant mesh by converting difficult mesh areas into Polyhedrals. This led to attempts to generate a hex or hex dominant mesh without using the Cut Cell method so that a mesh free of Polyhderal cells could be created and then used for the Grid Study.

Unfortunately, the discovery of these problems occurred late in the research life cycle. Due to the time constraints of a PhD and the resource cost of generating meshes of the LNG Modules abstracted geometry, there was simply not enough time to find appropriate mesh settings for the LNG Abstractions and iterate through simulations to a mesh independent solution. As such, the results presented in the next section should be viewed with the understanding that the meshes are the best that could be produced with the available resources, but are not mesh independent.

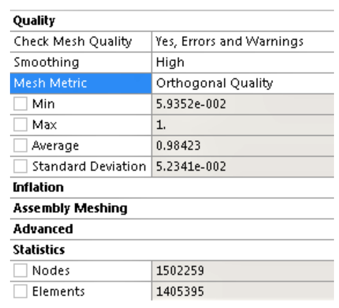
While a proven mesh independent solution would be preferable, the use of consistent settings across the simulations will allow for reasonable comparisons to be made.

Model	Nodes	Skewness	Orthogonal
Simplified	4,630,438	0.036	0.939
10cm Cubes	1,319,147	0.052	0.941

Table 5.4: Statistics for LNG Module model at different levels of abstraction. The Skewness and Orthogonal values given are the average, all values rounded to 3 decimal places.

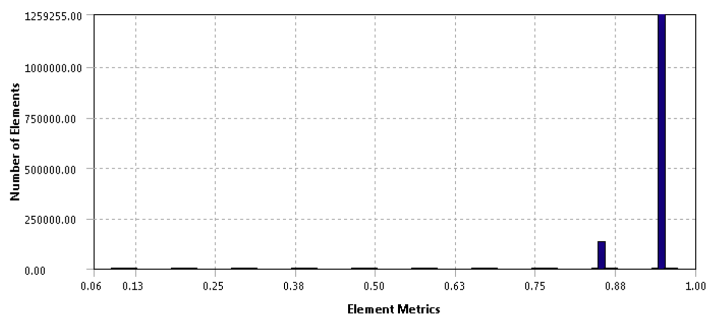
5.3 CFD Results and Analysis

Table 5.4 describes the meshes that are used in the final LNG Module Simulations. As discussed in section 5.2.1, there were many challenges in working with the geometry of the LNG Module Abstractions that made achieving good quality meshes difficult. The final meshes used for the simulations generally showed good quality averaged metrics, but due to the scale of them still have problematic areas where a few low quality cells introduce problems.



Quality	
Check Mesh Quality	Yes, Errors and Warnings
Smoothing	High
Mesh Metric	Orthogonal Quality
Min	5.9352e-002
Max	1.
Average	0.98423
Standard Deviation	5.2341e-002
Inflation	
Assembly Meshing	
Advanced	
Statistics	
Nodes	1502259
Elements	1405395

(a) Quality metrics from the Ansys® meshing software.



(b) Orthogonal Quality of the LMS Mesh by cell count. The two bars on the right represent the good quality cells, making up most of the total cell count.

Figure 5.9: An example of the quality problems when meshing LMS; The vast majority of cells exhibit good metrics, with only a tiny fraction causing issues.

The LMC10 mesh generally shows good quality metrics, the Cubified abstraction allowing for a much cleaner mesh with few areas where cells might be generated poorly. The LMS geometry is more problematic, with small gaps between elements and the alignment of the source objects introducing difficult to mesh areas. This resulted in the LMS geometry producing a mesh where the majority of cells are good quality, but small areas of poorly formed cells exist, as shown in Figure 5.9. These low quality cells are problematic to simulate, often resulting in errors and failure unless the simulation was run at very small time steps, such as 0.01s.

Past simulations had established it might take a few minutes of flow time for the vapour flows to stabilise, meaning using such small time steps throughout would consume vast quantities of computing time. This presents an issue, as only so much compute time is available, and before the final mesh was found other meshes with reasonable characteristis were also found and simulated against, with several of them appearing to simulate perfectly for several real world hours before a problematic cluster of cells caused a feedback loop resulting in the simulation ultimately failing.

Not being able to spend an unknown number of person days attempting to generate a perfect mesh with no problematic areas, it was decided to use a slightly modified approach when simulating the LMS mesh. It was observed in the simulation logs the failed simulations typically ran into an issue with resolving the viscosity turbulence formula, which eventually causes divergence and simulation failure. Ansys® Fluent®, in an attempt to prevent the simulation failing, would apply corrections when a cells turbulence exceeded 10000.

To allow the LMS simulation to resolve successfully without needing to trial an unknown number of meshes, its journal file was modified to include a dynamic mesh adaptation process, the key commands for doing so can be seen in Table 5.5. This process assesses the mesh at specified intervals, identifies any cells that meet the given criteria, and then modifies the cell. This modification takes a single cell which is having trouble resolving due to the input from its neighbouring cells, and subdividing it into several cells to reduce the complexity of the interactions occurring in each cell.

The built in improve/repair function from Ansys® Fluent® was also used on the base LMS mesh before packaging it for simulation on the cluster, using 1000 iterations of the Improve function to target the worst 0.1% of cells in the mesh. Between the

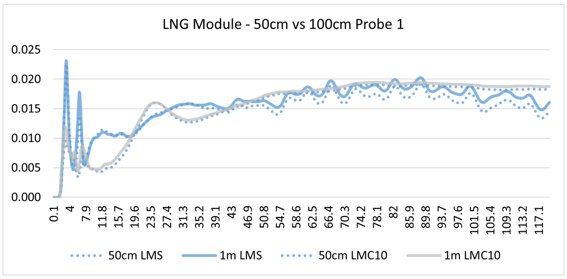
ANSYS Journal Command	**Purpose of command**
/adapt set min-number-cells 0	The minimum number of cells to change per adaptation.
/adapt set max-number-cells 2000000	The maximum number of cells to change per adaptation.
/adapt set max-level-refine 3	Each time a cell is adapted, it is marked. This sets the maximum number of times a cell may be changed.
/adapt mwg viscosity-ratio iso-value 0 5000 0 yes 20	Specifies the adaptation process; use mark-with-gradient (**mwg**) to assess the **viscosity-ratio** of cells based on their **iso-value**. Any cell with a value over **5000** will be dynamically refined (**yes**) every **20** iterations.

Table 5.5: The Ansys® journal commands that were added to the LMS journal, along with a brief description of what they do.

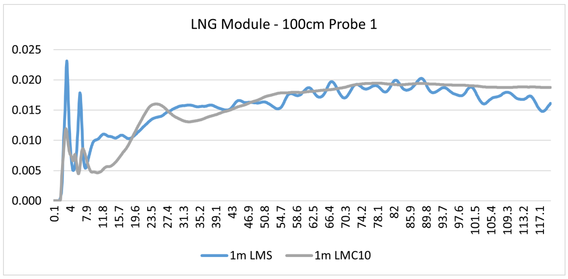
effort put into generating a mesh of reasonable quality, Ansys® Fluent® improving some of the cells, and the dynamic adaptation process, it was finally possible to successfully complete the LMS simulation while using increasing time steps. This approach allowed the computational resources to be balanced against simulation fidelity goals and limited real world research time, while keeping the simulation results comparable.

Using this method, the results shown in Figure 5.10 were achieved. Figure 5.10a shows that the 50cm and 100cm planes recorded not only very similar behaviour but also values at the first probe location. The values show slightly greater variance on the later probes, but continue to behave in nearly identical fashions and the values remain very close between the two different probe planes. To make comparisons to the Pan Pipes results easier the 100cm plane results will be used in this analysis.

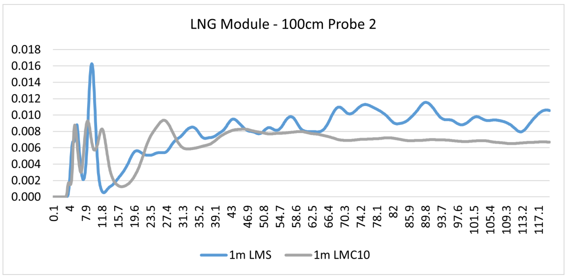
Looking at the results, a few differences can be observed when comparing to the Pan Pipes results. The first is that the values tend to show more fluctuation than the results from the Pan Pipes simulations which became stable very quickly, par-



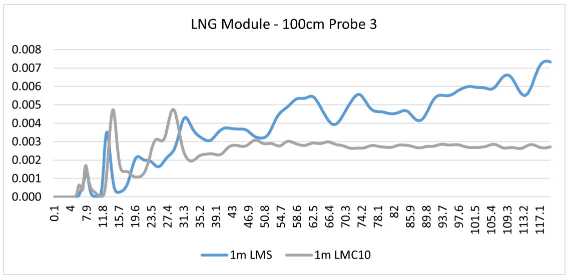
(a) 50cm vs 1m Probe 1 comparison



(b) Probe 1 results



(c) Probe 2 results



(d) Probe 3 results

Figure 5.10: Graphed values from the methane probes.

ticularly the readings from the LMS. This can be attributed to the more complex geometry of the LNG Module, causing ongoing interactions as flows move around various structures.

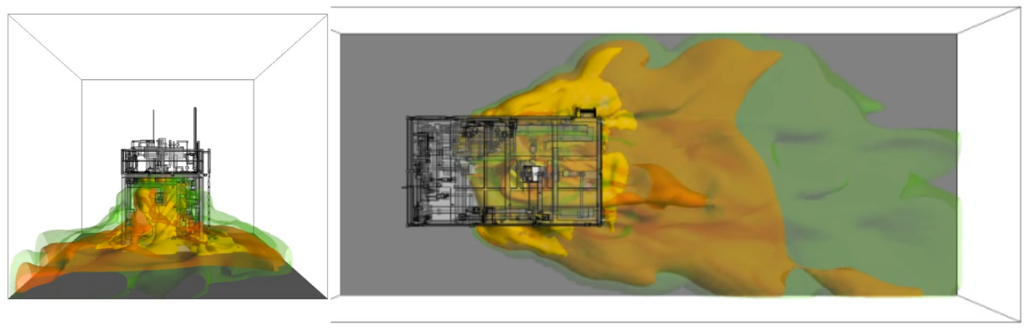
The recorded values in the LNG Module graphs also tend to be lower than those seen in the Pan Pipes simulations. This is due to the location of the probes, which in the Pan Pipes simulations were placed mostly within a short distance of the methane inlet and amongst the geometry itself. This was appropriate on the Pan Pipes to help examine the impact from one abstracted column to the next, but the complexity of the geometry in the LNG Module makes it more beneficial to compare levels further down stream, once methane flows have had a chance to stabilise more.

This is partly due to the disparity of the base geometry, where in the Pan Pipes model the geometry was carefully designed to have predictable results from abstraction, while the LNG module features much more complex geometry. While in the Pan Pipes model it was easy to predict how applying a 10cm abstraction would alter the shape of the existing 10cm diameter first pillar, within the LNG module there are numerous small pipes and features that can be vastly altered by going from a 1cm diameter pipe to a 10cm column. These differences mean that fluid flow within the abstractions are likely to vary much more greatly than they did in the Pan Pipes simulations.

This is why, at present, a user of LODOS must be aware of what details they can afford to abstract, and what details they should use a higher fidelity abstraction on or leave in their original state. If a user wished to use LODOS on the LNG Module while measuring methane levels within, they could use coarser abstractions like 10cm cubification on the outer edges of the geometry, and use original or simplified geometry within the structure. The abstraction settings used in this chapter were intended for a hypothetical simulation across multiple modules, where a researcher would want to retain a reasonable over and through flow approximation, but would not be interested in accurate measurements within the geometry itself.

In the Pan Pipes simulations, looking at the probe 3 and probe 4 results which are measured once the flows begin to resolve after the geometry, it was observed that the PPC10 values were lower than those in the PPO and PPS simulations, while the PPS values retained a similar behaviour and comparable values to those obtained from the PPO simulation. In comparison, the values in the LNG Module results do not only continue to fluctuate, but the LMC values compared to the LMS values do not behave in a manner that is as predictable. This raised questions of why the LNG Module abstractions behaved differently to what had been observed in the Pan Pipe simulations.

In order to analyse this behaviour, data from the methane clouds were visualised at various time points to examine the vapour flow. From these images it was determined that the vapour cloud in the LMC simulation flows differently to the one in the LMS simulation, the 10cm Cubification restricting the flow straight through and causing the methane cloud to vent along the East edge of the module's geometry, as can be seen in Figure 5.11. This restricted flow will be a large factor in the different behaviour between the Pan Pipes simulations and the LNG Module simulations.


(a) LMS Simulation at 10s flow

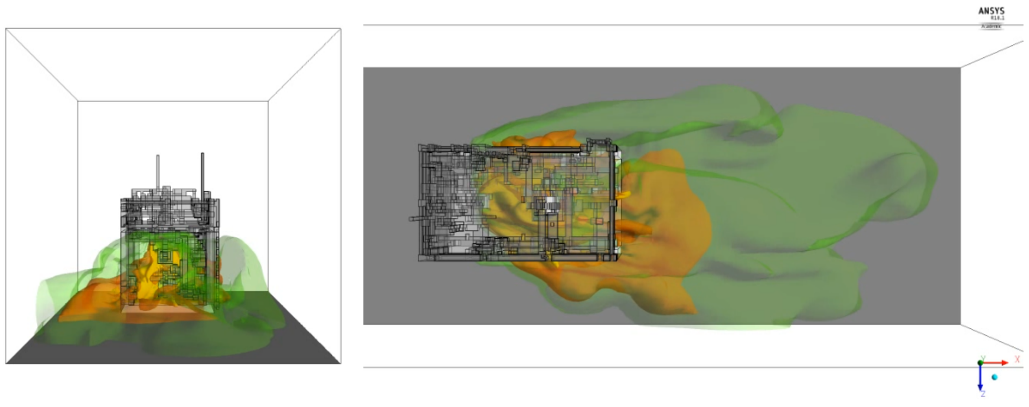

(b) LMC10 Simulation at 10s flow

Figure 5.11: A comparison of the LMS and LMC10 flow simulations, showing the difference in flow due to obstruction/restriction. The green area represents methane concentrations below the LFL, the orange methane concentrations between the LFL and UFL, and the red represents methane concentrations above the UFL.

Flow restrictions being introduced by Geometry Abstraction was also observed, to a lesser extent, in the PPC10 simulations, and has the potential to significantly affect the results of a flow simulation. At present, a user must take this into consideration when performing an abstraction and visually inspect models after each application to ensure important elements have not been lost, or fluid flow pathways obstructed. Further analysis and development of LODOS would allow for the creation of guidelines or an automated tool to assist the user with this issue, but the wide variety of applications CFD is used for makes creating guidelines or a tool to eliminate the issue difficult.

5.4 Discussion

From the literature it is clear that going from an existent model to a simulation compatible model often presents a problem, causing bottlenecks in the workflow, as discussed in Section 2.3. Anecdotally, when discussing the use of the LNG Module early on with an experienced CFD practitioner colleague, they recommended rebuilding the geometry from the ground up. They mentioned another PhD student they had helped previously, who had needed to spend several months reworking an existent model to get to the point where they could run a CFD simulation with it. These sorts of impediments make CFD costly to use, and limit the applications were it will be considered. Given the utility offered by CFD, this restriction to its use is a problem in need of a solution. The results achievable on an abstracted model that took only minutes to generate, on a model that was otherwise incompatible with CFD without extensive manual rework, supports the concept that model abstraction can help make CFD more widely accessible.

Looking at the LNG Module simulations, more work could have been done to examine the use of Geometry Abstraction given grreater resources. It would be good to be able to perform a simulation against the original geometry so the results could be compared. It would also have been useful to simulate the abstractions with the wind coming from multiple directions, to compare the impact of abstractions on simulations with cross winds. Simulating with a wider variety of abstractions, were only portions of the geometry were abstracted or a wider variety of Hide, Merge and Cube sizes were explored would also have been informative.

Unfortunately, there was not enough available resources, particularly in terms of person hours and computing time, to carry out all of the comparative simulations nescessary to systematically assess the impact of abstraction on this particular model. What was achieved was to take a model that could not be simulated at all and create a system that allows for an abstraction of the geometry to be imported and simulation ready within a day. From what is reported in the literature, that is exceptional, and does suggest Geometry Abstraction has viable use cases.

Using the data from the Pan Pipes simulations as a basis, it is expected that the values reported and the flow demonstrated in the LMS simulation will be relatively close to a simulation performed against the original geometry. A Simplified abstraction may seem like it should be straight forward to work with, but small edges and odd gaps from the original geometry can persist or be added when objects are removed, merged, or replaced with simple prisms. While the LMS simulation was still computationally expensive to run compared to the Pan Pipes simulations, it required vastly less time and effort than would have been necessary to take the original geometry through manual rework and eventual simulation.

The flow restriction and values exhibited in the LMC10 simulation show the 10cm Cubification abstraction would be less appropriate for a simulation at this scale, of a single module, but would likely be useful when examining the flow across several modules in combination. A user looking at simulating something on the scale of this module would be better off using a smaller cube size or Simplification. It is expect that a Cubified abstraction with cubes of 2cm or 5cm would reduce unintended flow restriction while still allowing for a computationally cheaper simulation, but there was insufficent time to test this.

Taking into consideration the person hours and computational time required to prepare and run a simulation against geometry like that found in the LNG Module, LODOS shows great potential to improve existening model to simulation work flow. The data from the LMC10 simulation highlights the need to approach abstraction with care, but also shows how it can support even minimally skilled users in setting up and running simulations in relatively short periods of time.

The final chapter presents research conclusions about LODOS in its present state, as well as discussing future research opportunities to extend its capabilities.

Chapter 6

Discussion and Conclusion

The preceding chapters present the research aims, the identified challenges, the proposed solutions, a proof of concept implementation, and two test cases used to evaluate the proposed solutions. This chapter provides a brief summary of the research that has been carried out, discussing the work to date with reference to the research aims (6.1). A conclusion based on the work discussed in this thesis is presented (6.2), along with avenues for extending this work through future research (6.3).

6.1 Discussion

This research has been conducted with four research aims, which are laid out in the Introduction chapter (1.2). This section presents each of the aims again, with a discussion of how the work to date has addressed each one. They are presented in the order:

- Research
- Design
- Proof of Concept
- Test and Evaluate

6.1.1 Research

Identify challenges in the current methods for moving from existent models to CFD Simulations.

A literature review was conducted at the outset of this research, along with informal discussions with CFD practitioners, to identify the challenges in going from Building Information Model (BIM) or CAD geometry to a CFD Simulation. The identified challenges were examined, with further literature review carried out to identify any existing solutions. Consideration was then given to which challenges could be addressed within the scope of a PhD, with the difficulty in using existing geometry without extensive manual rework identified as the area that could be best addressed.

The challenge in going from an existing model to a completed CFD Simulation is made up of several smaller challenges. Research identified the cost of performing a CFD Simulation as both an independent challenge, as well as an element that compounds the difficulty of addressing the other challenges. Even performing a short simulation against simple geometry in a small domain can take hours to setup and hours more to run, making the testing of any proposed solution resource intensive.

Another significant challenge is the gap between the geometry requirements of good CFD models, and the geometry detail common to models intended for fabrication. Models intended for fabrication can contain small gaps intended to allow for fitting during the build process, and will contain small features and complex arrangements of geometry. In a model for CFD, small features will typically be removed unless they are significant to the simulation, and a minimal approach to geometry will be taken to prevent overlapping geometry from introducing complex planes. These different approaches can mean it is cheaper to rebuild a model from the ground up for CFD simulation, rather than try to use the model built for fabrication, where geometry problems need to be manually found and addressed.

As presented in the Background chapter (2), other researches have examined these challenges, with some presenting novel solutions. The solutions presented in the literature tend to focus on an aspect of the challenge, meaning none of the reported solutions solve it for all use cases. The scope of the challenge, and diverse uses for CFD, are such that further work on novel solutions is justified, with it likely that several solutions will need to be used in conjunction to provide across the board improvements.

In terms of the stated research aim, the literature review and investigation performed provide a strong basis for the work that was then conducted. The research highlighted the potential of BIM and CFD individually, as well as the benefits of reducing the cost of moving from BIM/CAD to CFD Simulations. The research also identified the scope of the challenge, allowing potential paths for addressing it to be identified.

6.1.2 Design

Study the challenges identified to understand their cause, any existing processes for dealing with them, and new ways they might be addressed. Develop a solution or set of solutions based on this study, with an emphasis on extending and improving existing workflows.

From the research conducted, the difficulties in moving from existent models to full CFD simulation was identified as an ongoing challenge. Within the AEC industry, particularly with the advent of BIM, there is a drive towards having a single facility model which can be used throughout the design and life of the facility to help reduce costs. This means many companies already have model assets, or will need to develop them before they begin construction, which contain detailed geometry for their facility, but that cannot be easily used for CFD Simulations.

The literature review and informal discussions showed that CFD simulations are often carried out against models which have been optimised to balance their level of detail against the simulation requirements. That is, unnescessary features that would consume processing resources but have little real impact on the intended simulation are ”abstracted”. In the context of this research, abstracted is used as a catch-all term to describe removing, simplifying, or otherwise altering a geometric detail to reduce the overall cost of performing a CFD Simulation against the model.

This led to examining the geometry in existing models and considering ways in which the geometry could be optimised in an autonomous or semi-autonomous way. The variable use of CFD Simulations to achieve different outcomes makes a fully autonomous solution impractical, making a user guided semi-autonomous system the more logical approach. By developing a tool to automate the abstraction process under the guidance of a user, the person hour cost to prepare an existing model can be reduced without limiting potential applications.

Algorithms for performing Geometry Abstraction were developed, with refinements made to allow user input. Methods for applying these algorithms were examined, and it was discovered that there was no support for the direct import of BIM geometry into the common CFD software suites. Available solutions for BIM and CFD were compared, with some weighting given to how widespread their use was and how much support they had.

From this software analysis, Ansys® was selected as the best available CFD solution, with widespread use in the literature and industry. Given the lack of direct BIM integration, the Autodesk® product suite was selected as it offers BIM specific solutions (Autodesk® Revit®) as well CAD solutions (Autodesk® AutoCAD®), with strong interoperability between systems. At the time research began, Ansys® did not support plugin development, but Autodesk® featured a mature API, strengthening the case for the use of the two systems in conjunction.

During the design process, small scale simulations were carried out to understand how the two software suites worked and interracted. From these tests, difficulties in going from BIM to CAD to CFD ready meshes were found. Details on these challenges and how to work around them are documented in Section 3.1.

In terms of gemoetry abstraction, the first algorithm designed was “Simplification”, detailed in Section 3.3.2. Having examined CFD meshing, it was clear that spherical and cylindircal objects were particularly costly to simulare due to the large number of cells required to capture fluid flows around them. The simulation cost of such geometry remains high even when the dimensionality of the model is such that they will have little to no overall impact, introducing the idea of substituting the complex geometry for a simpler approximate.

Replacing a cylinder with a simple rectangular prism greatly reduces the meshing requirements, and thus the computational cost of realising the CFD Simulation. To maximise utility, the algorithm was designed to allow the user to set a maximum threshold, meaning a large scale model could have pipes and railing reduced to computationally cheaper prisms, while preserving larger objects such as silos, storage tanks, or smoke stacks. This algorithm allows a user to quickly reduce the geometric complexity of their model, while being able to keep complexity in the areas of interest.

While Simplification can greatly reduce the simulation costs, in complex environments objects such as taps and valves can introduce small features that, even Simplified to a prism, result in fine mesh requirements. The“Hide” algorithm, documented in Section 3.3.1, was designed to resolve this issue. In brief, it accepts user input to define a minimum threshold defined by either object volume or a set number of geometry dimensions. The algorithm then examines all objects, and removes those that do not meet the required threshold.

From studying models designed for use in industry, it was clear that logical objects within a model are often constructed from several smaller objects, such as a single pipe being made of several lengths of pipe with valves and taps. This meant the Hide algorithm, used in isolation, risked removing elements from the model that a user may not have intended, particularly when working with large complex models where manual checking is more difficult. Given the varied use of CFD and possible design intents of source models, this presented a challenge.

The use of several smaller objects to represent a single logical object also increases the difficulty of working with that model, increasing the memory footprint and number of operations required to manipulate it. This lead to the development of the "Merge" algorithm, where objects are compared to establish if they are adjacent or overlap each other, then if the size difference between them is within a user specified limit, they are combined into a single object with the combined dimensions of its components. This method is intentionally somewhat crude, the use of the combined dimensions meaning that a level of geometric complexity is removed along with the object count being reduced.

While these algorithms, particularly in conjunction, can greatly reduce the geometric complexity of a model and thus its cost to simulate, meshing can still be challenging in models with dense clusters of objects. Even once Simplified, a dense cluster of objects such as an array of pipes could result in small gaps, meaning a dense mesh may still be required to allow simulation. Further to this, if the pre-abstraction geometry was of varied sizes, the abstracted geometry may have small, difficult to mesh areas where the Simplified geometry now interact.

This lead to the development of the final absctraction algorithm, "Cubification". This algorithm represents the most extreme form of Geometry Abstraction, subdividing the domain into cubes of a user specified scale, then filling the cubes that overlap with objects in the original model. This algorithm allows a user to quickly convert a model into one with features of a known minimum size, allowing them to mesh it easily.

Each of the algorithms can reduce the geometric complexity of a model and thus the cost of working with it. Each algorithm also has trade offs, with the end user best placed to decide what level of dimensionality is required for their simulation and thus what level of abstraction is appropriate. The algorithms are designed so they can be used in conjunction, meaning a user could take a large and complex model and perform increasing levels of geometric absraction as they move away from the area of interest to their simulation, reducing the computational cost of elements in-line with their impact on the simulation outcome.

With reference to the research aim, the proposed algorithms represent a novel approach to reducing the computational costs of performing CFD Simulations, while also reducing the person hour cost of preparing an existing model for simulation. While the reliance on user judgement limits the current application to users able to make appropriate decisions, it also means the proposed solution can be used with the widest range of problems.

6.1.3 Proof of Concept

A proof of concept system that partially or fully implements the designed solution(s) will need to be built. The implementation needs to be sufficient to allow for the evaluation of the proposed solution(s).

During the initial scoping of this research, the Ansys® Workbench® suite did not support plug-in development. This meant that in order to develop a proof of concept implementation of the algorithms, either a stand alone system or a plug-in for an existing BIM/CAD software package would need to be built. Due to its prominence in industry and literature, development of the algorithms as a plug-in within Autodesk® AutoCAD® was deemed the more efficient approach, with the process detailed in Chapter 3.

The proof of concept was dubbed the LODOS System, standing for Level Of Detail and Object Simplification System. It was developed as a plug-in using the Autodesk® AutoCAD® API using C# on a Microsoft® Windows® desktop, and has been tested and shown to work with Autodesk® AutoCAD® 2016 and Autodesk® AutoCAD® 2018. In its present state it can process models on the scale of the LNG module, approximately 3000 objects, in under a minute.

The current implementation of the LODOS system only provides a partial implementation of the algorithms as designed. Due to limitations in what information is exposed via the Autodesk® AutoCAD® API to the plug-in, only basic information on the geometry of objects could be accessed. This means that the implementation of the algorithms in LODOS are restricted to only using the bounding box information to determine the size and shape of an object.

The bounding box of an object is the smallest box capable of containing the object. Because no field on object type is available, the current LODOS system cannot differentiate between a cylinder, a box, or a complex object made up of several geometric objects. This means all objects must be treated as a simple prism with an equivalent size to their bounding box, meaning a collection of adjacent pipes could all be merged, before the prism that subsumed them is merged with the junction box they all went into.

While the present implementation does introduce limitations, it is sufficient to allow testing of the impacts of Geometry Abstraction on CFD Simulations, inline with the research aim.

6.1.4 Test and Evaluate

Finally, a set of simulation experiments need to be designed and carried out to examine the impact of the designed solution(s). Data from these experiments will need to be collected, analysed, and then presented as a thesis.

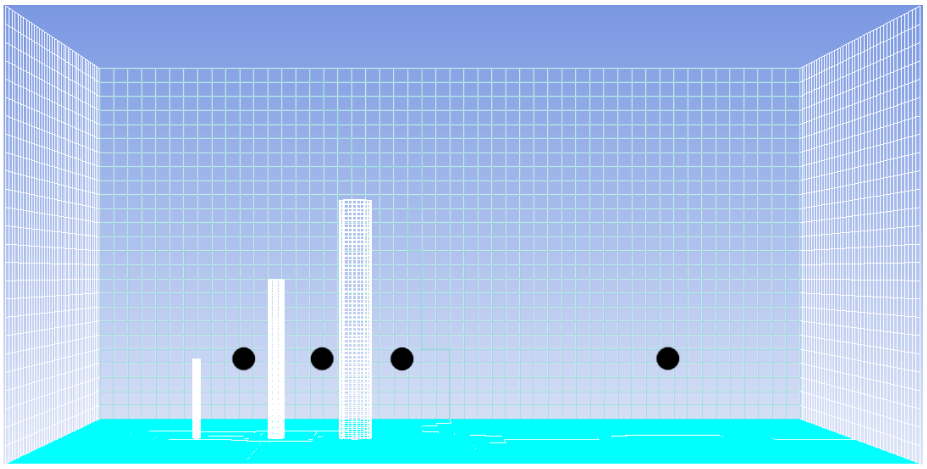
To evaluate the utility of Geometry Abstraction, a set of tests are needed. Geometry Abstraction can impact the difficulty of meshing a model, the computational cost of running the simulation, and the accuracy of the results compared to the original geometry. Performing CFD Simulations, even against abstracted geometries, can be a costly process meaning extensive and exhaustive testing would likely not be possible.

Taking these factors into account, a set of tests were designed to maximise comparison of results between tests, while minimising the necessary number of simulations required for testing. Going from a cylinder to a prism represents one of the largest losses in geometric detail, and thus an abstraction most likely to affect simulation cost and result accuracy. The first test case was designed around this, using a small set of cylinders with known separation and dimensions to allow comparison between the original model and a set of abstractions.

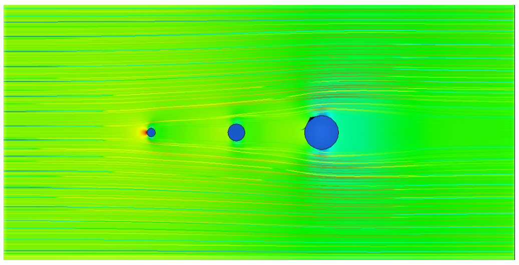
This test is covered in detail in Chapter 4, with details on the layout of the geometry, the abstractions performed, the simulation parameters, and the results generated. In brief, the model is composed of three cylinders of increasing size aligned alone a line which were nick named the ”Pan Pipes”, with Simplification and Cubification abstractions generated. The initial model was used to develop appropriate mesh and CFD Simulation settings, with the same values used on the abstraction simulations to allow for comparison between simulations.

These initial tests were very informative, showing the potential for prism abstractions to introduce flow obstructions as can be seen in Figure 6.1b/c, as well as providing an idea of the impact of Geometry Abstraction on the computational costs and result accuracy. The abstracted geometries require approximately 1/5th the mesh nodes of the original geometry, and approximately 1/3rd of the simulation time. Analysis of the methane levels at the probe locations, such as that shown in Figure 6.1d, show the abstractions tend to register higher methane values with a roughly 25% variance in the values in the abstracted simulations, with the less abstracted models measuring closer to the original models values.

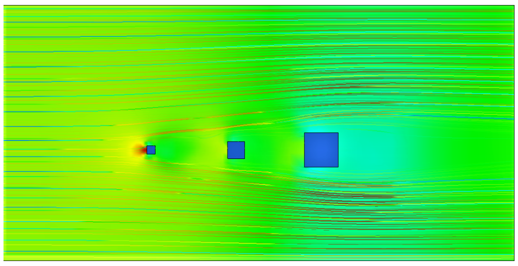
The lessons learned from the Pan Pipes simulations were used to inform the second phase of testing, against the more complex geometry of the LNG Module. The LNG Module is an interesting test case, as the complexity of the geometry and a small number of non-manifold objects present in it means the original geometry can not be successfully meshed. This means the LNG Module provides an opportunity to test Geometry Abstraction as a means for quickly adapting difficult models to something CFD Simulation compatible, but also means there is no base simulation to compare results to.



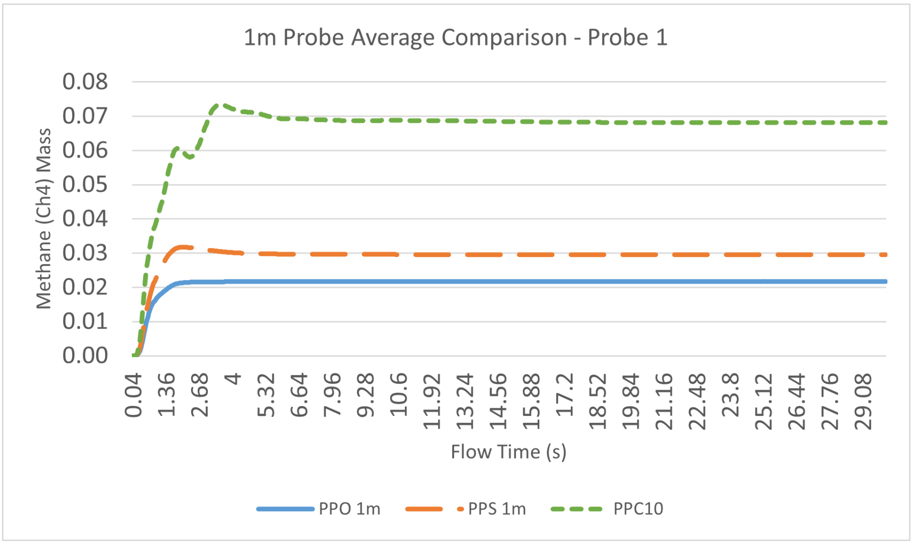
(a) Pan Pipes domain, black dots indicating methane measurement locations



(b) Flow analysis, Original geometry



(c) Flow, 5cm Cubified Geometry



(d) Results from the first probe location, across the four comparative simulations

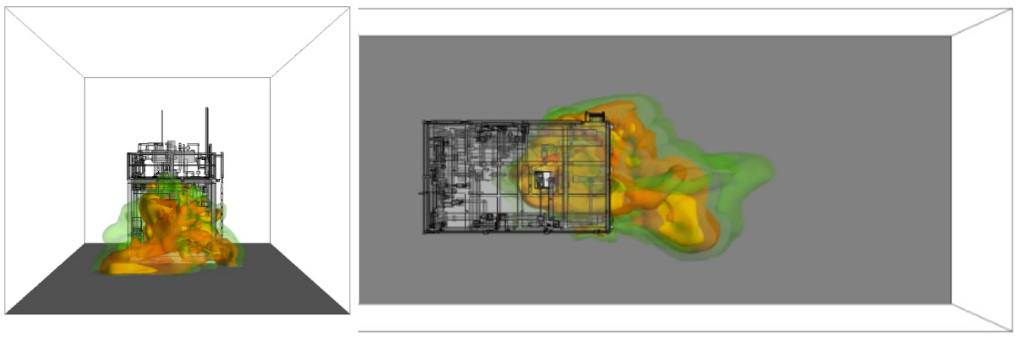
Figure 6.1: An example of the layout of the Pan Pipes simulations and comparative data generated by them

The LNG Module also provided a chance to examine Geometry Abstraction in terms of a real world use case. Two abstractions were generated, a Simplified and a Cubified version, which are detailed in Section 5.1. The Simplified abstraction had nearly 5 millions nodes, and a 10cm Cubified abstraction had nearly 1.5 million nodes, suggesting a successful mesh of the Original geometry would require somewhere in the realm of 15 - 50 million nodes.

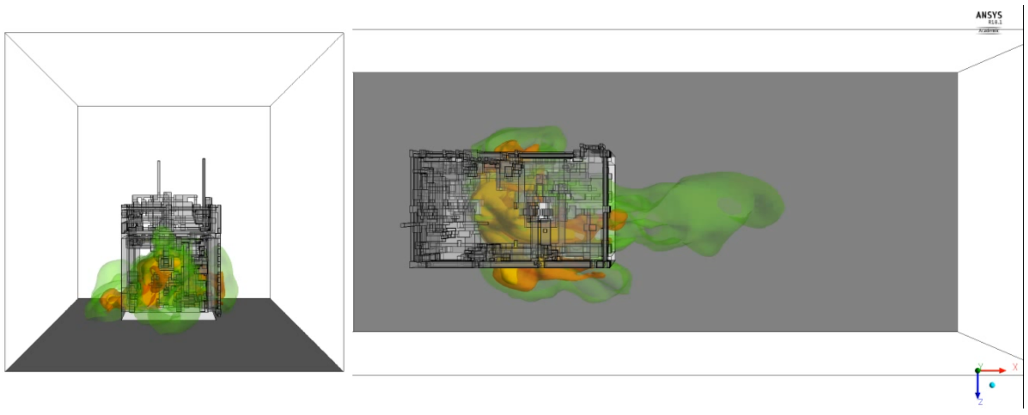
The simulations for the Abstracted LNG Modules were run against a computing cluster, using three nodes with 16 logical cores. Simulating 30 seconds of flow time on the Simplified abstraction took just over 18 hours of real world time, which is 864 hours of compute time. Using the Pan Pipes metrics as a basis, this means a simulation against the original geometry would require roughly 54 hours of real world simulation time, assuming similar hardware availability, once a successful mesh was generated.

The massive computational cost of a single LNG Modules geometry, even after abstraction, highlights the potential for Geometry Abstraction to make complex CFD Simulation more accessible. It also demonstrates the need for caution when using Geometry Abstraction on complex models, with visualisations of the flow from the 10cm Cubified abstraction noticeably different to those from the Simplified abstraction, due to Geometry Abstraction introducing obstructions to the flow. This can be seen in Figure 6.2.

To minimise unintended flow obstructions, the LODOS system at present must rely on the end user to perform inspection and make judgement calls. As discussed previously, the large problem space that CFD Simulations can be applied to makes it difficult to automate too much of the abstraction process, for fear of removing features that are necessary to a particular simulation. However, if Geometry Abstraction is adopted for wider use, it would certainly be possible to develop domain specific default values, such as recommended abstraction values for when dealing with an LNG plant or whole city simulations.



(a) Visualisation of the vapour cloud with the Simplified abstraction

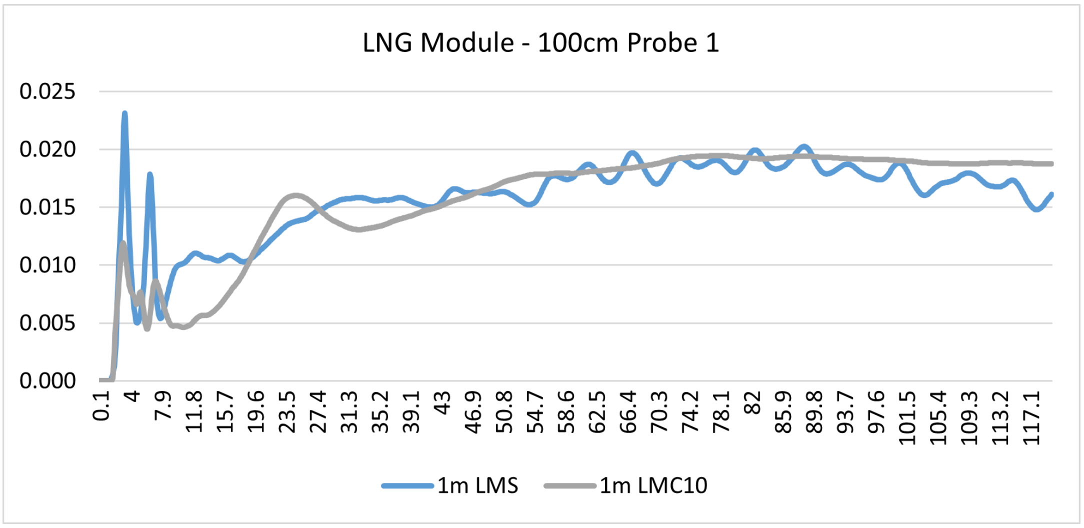


(b) Visualisation of the vapour cloud with the 10cm Cubified abstraction

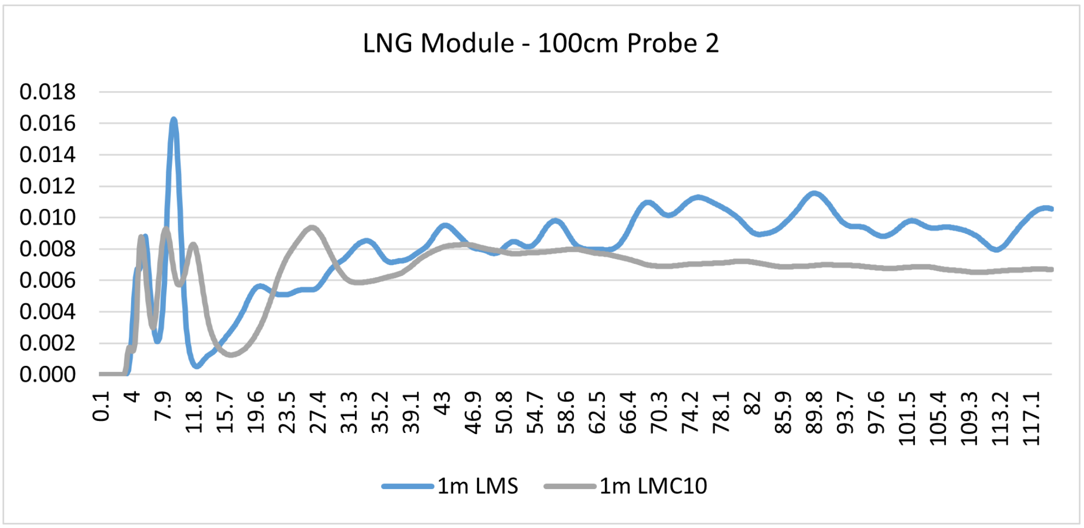
Figure 6.2: A comparison of the methane cloud at 5s flow time between the Simplified and 10cm Cubified LNG Module abstractions.

The results from the LNG Module simulations, see Figure 6.3, show overall good agreement between the two simulations for methane readings. The Simplified abstractions readings show more variance, which is to be expected given both the easier flow through the geometry and the greater number of nodes in the simulation causing increased mixing, but the general behaviour and peaks are similar. It was noticed that the values tended to be higher in the LMS simulations, particularly later in the flow time, which is explained by the location of the probes and the greater flow through the module in the LMS, as shown in the flow visualisations.

The first test case, the Pan Pipes, shows that abstracted geometry can produce results within a reasonable margin of the original geometry, with a tendency towards over prediction which is useful both in terms of safety as well as estimating the results in comparison to a non abstracted model. The second test case showed



(a) Probe 1 results



(b) Probe 2 results

Figure 6.3: Graphed values from the methane probes for the LNG Module abstractions.

that the system can be used to quickly make complex models CFD compatible, with caution needing to be taken to minimise the introduction of flow obstruction. Both test cases suggest that Geometry Abstraction can reduce the bottleneck of going from existent models to CFD Simulations with useful results, though the limited resources available within a PhD mean further testing is needed to better quantify the cost.

With regard to the final research aim, the testing and analysis are sufficient to demonstrate the utility of Geometry Abstraction. This thesis presents the relevant details of the work, testing, analysis and conclusions, addressing the research aim in full.

6.2 Conclusion

CFD simulation allows for many elements of a design, from safety to efficiency, to be explored without the resource cost of full production. As a technology it has huge potential to assist in numerous fields, from crowd modelling for safety to car designs for fuel efficiency, and many more. At present this potential is hampered by the current resource costs to take any design from imported geometry to evaluated simulation data.

The presented collection of solutions dubbed LODOS provide a path for taking existing models through to CFD ready geometry with reduced time and effort by the end user. The solutions are somewhat limited by their current implementation, but the data gathered shows that even in a limited implementation they can assist users. The results presented in this thesis support the idea that a Geometry Abstraction system can help remove the bottleneck between existing geometry and CFD based analysis, as well as lowering the resource overheard to achieve an analysis while still generating useful results.

Further study is needed to understand how much of the observed margins of error between original and abstracted geometry are from the changes in geometry, and how much are from the resulting lower mesh density. Further research can also establish best practices for minimising the introduced error from abstraction, while maintaining most of the gains in ease of use and reduced computation. The data provided in this thesis shows Geometry Abstraction can aid in achieving useful CFD simulations, even on complex or incomplete models, though further analysis is needed to better quantify the cost to benefit of this approach.

Geometry Abstraction as presented in this thesis provides a novel approach to reducing the resource cost of achieving CFD simulations. It can greatly reduce the cost of performing a simulation, particularly in any scenario were a millimetre accurate result is less important than a general understanding of the likely flow, such as early site layout checking. Ideas for extending this work will be presented below, in Future Work (6.3), with the work presented above providing a working foundation, as well as providing evidence to justify further investigation.

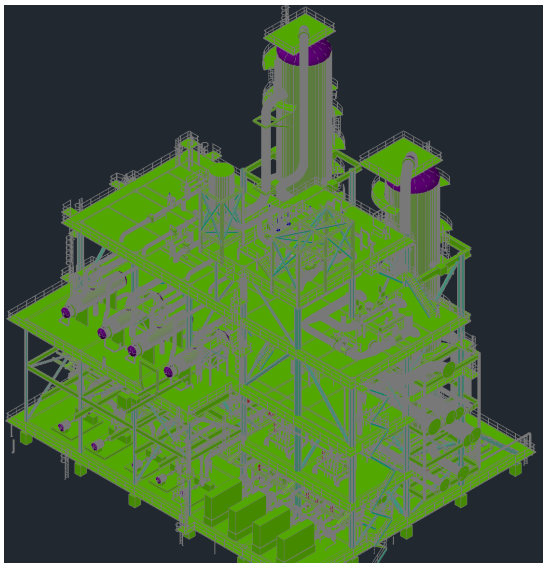
6.3 Future Work

This final section will present research ideas that build upon or vary the concepts presented in this thesis. The time available for conducting research is always finite, and some ideas only occur when it is too late to justify the cost of pursuing them. This section presents several ideas that could not be explored during the research to date, but are likely to be useful during further development of the presented concepts.

6.3.1 Further validation and quantification

Due to time limits, only a fairly small set of CFD simulations were carried out with primarily just the two models. While a larger geometry was acquired and the abstraction algorithms were tested against it, insufficient time and resources were available to attempt a simulation against it. This third geometry, an LNG Train viewable in Figure 6.4, would have required orders of magnitude more computational resources and also presented issues with the abstraction systems.

An unknown bug caused extra geometry to be added to the top of the LNG train when it was abstracted, see Figure 6.5. The cause of this error is unclear at this time, likely resulting from the Merge algorithm combining two or more objects into a single unified plane. Identifying the cause of this bug and resolving it was not given a high priority, as the LNG train was the least important of the three available test cases, with its scale making it unlikely sufficient time would be available to properly


(a) LNG Train

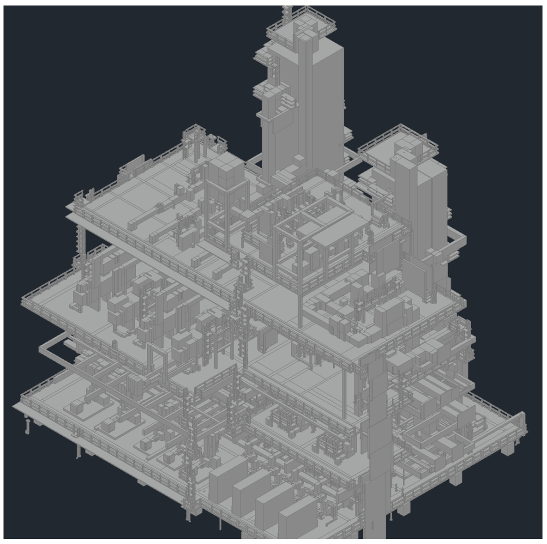
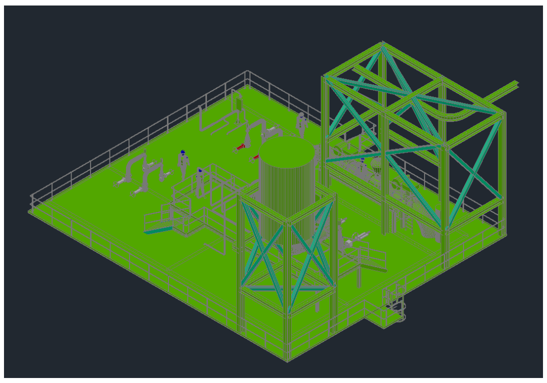

(b) 10cm Cubified abstraction of an LNG Train

Figure 6.4: LNG Train abstraction example.

simulate and assess it. The bug does not seem to present in any of the other models.


(a) LNG Tower, with most of the model hidden

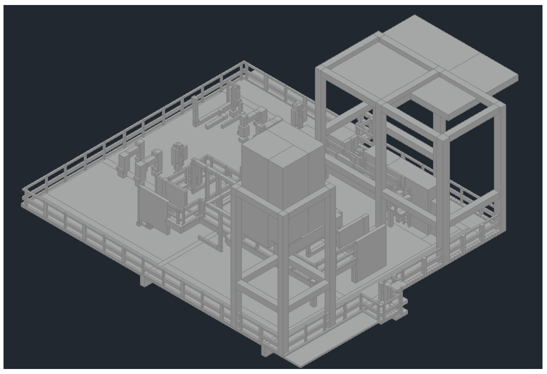

(b) 10cm Cubified abstraction of the LNG Tower

Figure 6.5: Isolated versions of the LNG Trains tower section, showing the unexpected additional geometry after abstraction.

Beyond the LNG Train, it would be useful to run further simulations against a range of geometries under a set of scenarios. While the simulations carried out to date give some idea of the loss of accuracy from abstraction, it would be useful to perform further testing to fully capture the behaviour of a CFD simulation using various degrees of assistance from LODOS. A set of two or three geometries bridging the gap between the Pan Pipes and LNG Module in terms of scale and complexity would be very useful for this purpose.

In general, the LODOS system provides a user with many possible methods for abstracting their geometry, and a more comprehensive examination of the impacts of those methods would be useful.

6.3.2 Simplification - Edge correction

Because of the current state of LODOS, it is possible when abstracting complex and asymmetrical geometry to end up with small, odd gaps. These small gaps can result in edges with very small, sub-millimetre lengths. Such edges present issues when attempting to import and mesh the geometry.

These problems can be addressed with existing tools, such as Ansys® SpaceClaim®, but that process is largely manual and requires the use of additional software and person hours. If LODOS undergoes further development, adding algorithms to identify small edges and correct them to user specified sizes would be a worthwhile application of time. This issue does not effect Cubification due to its grid based nature and inherent merging algorithms, but does seem to present issues for the other abstraction algorithms.

6.3.3 Simplification - Many face simplification

At present, the implementation of LODOS through the Autodesk® AutoCAD® API provides several limitations to functionality. Despite best efforts, no method for identifying the type of object being accessed could be identified. While Autodesk® AutoCAD® may internally track the type of object, such as Cylinder or Sphere, it does not appear to expose this information to developers through the API.

This means that all objects had to be abstracted to their most basic forms, simple prisms based on their bounding boxes. While this did present substantial savings in terms of mesh density required to capture the new shape, a loss of air flow fidelity was inevitable and beyond what was desired. Given access to better identification of the object to be abstracted, it would be possible for LODOS to replace objects with lower complexity alternatives while maintaining a better approximation of the airflow dynamics.

For instance, when replacing a cylinder, such as a pipe, rather than a simple rectangular prism a user could be presented with options for replacing cylindrical objects with a pentagonal, hexagonal, or octagonal based prism. This would reduce the meshing requirement of trying to capture a round shape, without introducing the flow obstruction of a flat face that is currently unavoidable with simple rectangular prisms.

A possible method for implementing this within the Autodesk® AutoCAD® API is discussed below.

6.3.4 Autodesk® AutoCAD® - Exploding geometry back to basics

A concept that occurred after data collection had been completed and writing of this thesis had begun, was that it may be possible to get better information on the type of geometry LODOS is abstracting by breaking it down to basic components. As covered above, the Autodesk® AutoCAD® API provides access to only very limited information about a 3D Solid, preventing some forms of more intelligent abstraction. However, it is possible within Autodesk® AutoCAD® to "explode" an object, converting it from a single object into the collection of more basic objects that composed that object.

It appears to be possible via this method to revert a cylinder from a 3D Solid to a collection of entities, then interrogate these entities to identify their type. An algorithm could be developed to explode an object back to primitives, and then use analysis of those primitives and their locations with relation to each other to more intelligently identify what the object was. This would allow LODOS to classify objects as cylinders, cones, etc., and then apply the relevant abstraction or flag the object for user review.

It must be emphasised this idea occurred very late in the timeline of the PhD, so only basic testing has been performed. Exploding a cylinder does seem to produce an identifiable set of basic shapes, but testing would need to be carried out on other 3D solids, in particular composite objects, to assess how universally applicable the idea is. Based on the initial investigation, it does seem promising.

6.3.5 Automated Domain Assessment

To assist users in achieving an optimal level of abstraction, it could be useful to add a tool that allows them to define their intended domain volume before performing abstractions. This way the original geometry could be compared to the intended domain, and feedback given to the user on what volume of the domain is obstructed. Then, as a user applies abstractions, this value could be calculated and compared back to the original, providing a user with ongoing feedback of how much of their domain has been obstructed or cleared by each abstraction.

This could reduce the users need to manually check the geometry to establish if flow has been impeded in a particular area. While more difficult, it should also be possible to have a user define their inlet locations and perform a simplified flow analysis using that as an input. This would allow an Automated Domain Assessment system to provide feedback to the user on any large changes to model flow characteristics resulting from an abstraction, potentially allowing a user to avoid issues similar to those encountered with the LMC10 flow compared to the LMS flow in Chapter 5.

6.3.6 Cubification - Smoothing

Similar to the goal of the work suggested above, in **Simplification - Many face simplification**(6.3.3), it may be desirable to provide a more aerodynamic surface to Cubified abstractions. Due to the higher level of abstraction involved in Cubification, this process carries a greater risk of introducing unintended geometry, depending on the inputs used. An additive or subtractive method of smoothing could be added, most likely requiring a user interactive approach for best results.

To smooth the transition between cubes, this method would introduce angled planes. This would need to be applied with a degree of intelligence, so most likely only in situations where a collection of cubes meets a given set of parameters. While the proposed smoothing may prove useful for some applications, it could also subtract from one of the key benefits of Cubification, that of providing a known minimum grid size. As such the cost to benefit would need to be examined for this new method.

As an example of how Cubification Smoothing would work, please see Figure 6.6. In Figure 6.6a, upon detection of cubes conforming to a prescribed shape, additional geometry is inserted to smooth the 'step' of the cubes. In Figure 6.6c, cubes would need to be marked in the matrix so that they can be replaced with triangular prism representations, effectively subtracting geometry to smooth the steps that way.

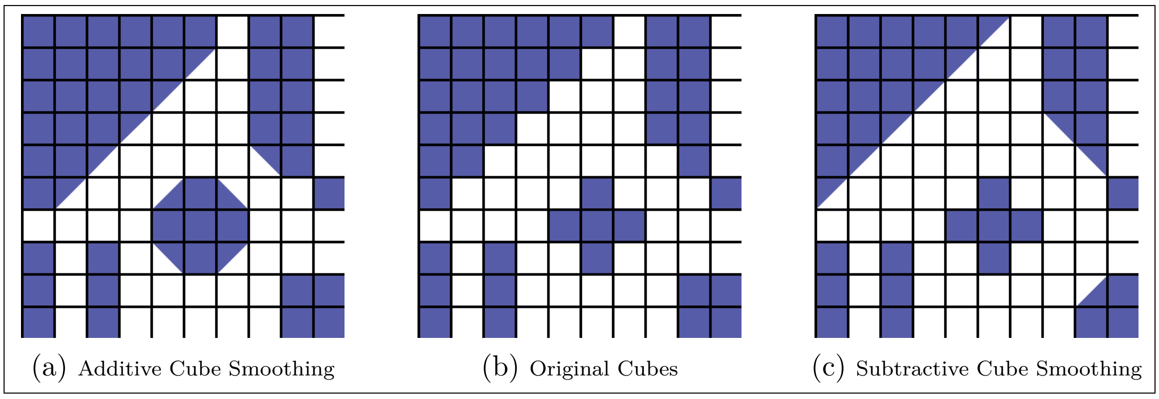


Figure 6.6: These images provide a simple visual example of how additive and subtractive cube smoothing would work, with reference to an example cube grid.

6.3.7 Cubification - Sub-Cubification

Because the current implementation of Cubification generates cubes based on which grid cells are impeded by the bouncing box of objects, it will typically block more of the flow domain than a Simplification would. Ideally Cubification would only generate cubes based on where the grid cells are impeded by the actual geometry, allowing for the benefits of known minimum edge length while minimising unnecessary flow obstruction. The images in Figure 6.7 illustrate an example of this 'sub-Cubification' method, compared to the current Cubification algorithm.

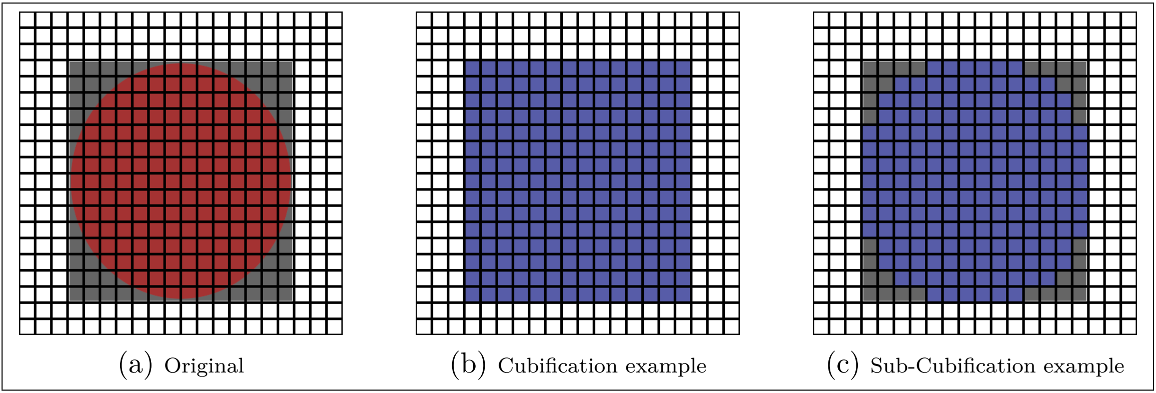

(a) Original (b) Cubification example (c) Sub-Cubification example

Figure 6.7: An example geometry (red) with its bounding box (grey), accompanied by depictions of Cubification with current methods or the proposed Sub-Cubification method.

6.3.8 Cubification - Minimum overlap

At present, when performing Cubification, any amount of overlap by the bounding box with the Cubification grid is sufficient to generate a cube. This means that if the bounding box of a pipe impinges upon the defined space of a 10cm^3 by even a single millimetre it will generate an entire cube. While a user can select a smaller cube size to minimise the airflow obstruction caused by minor overlaps, this would still result in the same issue at a smaller scale while increasing overall simulation requirements.

To reduce the impact of cubes being generated where only a small amount of overlap would exist, a user input could be added to allow the specification of a minimum overlap. The Cubification algorithm could then be reworked to consider how much of a given potential cube's space is taken up, and only mark the grid for generation if it exceeds the specified minimum. This method, particularly in conjunction with some form of sub-Cubification, would allow a researcher to quickly convert a model into an abstraction with a similar air flow profile to their original geometry while also having a known minimum edge length.

Bibliography

Al-Kodmany, K. (2013). Crowd management and urban design: New scientific approaches. *Urban Design International*, **18**(4), 282–295.

Aliabadi, S., Tu, S., Watts, M., Ji, A., and Johnson, A. (2006). Integrated high performance computational tools for simulations of transport and diffusion of contaminants in urban areas. *International Journal of Computational Fluid Dynamics*, **20**(3-4), 253–267.

Autodesk.help (2020). Revit Products - Import a CAD File. From https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/Revit-Model/files/GUID-02E2B1A7-AA87-466A-854C-728296A84628-htm.html, Accessed on 2020-04-07.

Azhar, S., Hein, M., and Sketo, B. (2008). Building Information Modeling (BIM): Benefits , Risks and Challenges. In T. U. o. S. M. Sulbaran and C. U. o. S. M. Sterling, editors, *44th Associated Schools of Construction National Conference*, Auburn, Alabama.

Bernstein, H. M., Jones, S. A., Russo, M. A., and Laquidara-Carr, D. (2014). The Business Value of BIM in Australia and New Zealand: Smart Market Report. Technical report, McGraw-Hill Construction, Bedford, MA.

Chang, J. C. and Hanna, S. R. (2004). Air quality model performance evaluation. *Meteorology and Atmospheric Physics*, **87**(1-3), 167–196.

Cheng, B. and Wang, Y. (2010). BIM's content and its application in contemporary architectural design. In *2010 International Conference on Management and Service Science : MASS 2010, Aug. 24-26, 2010, Wuhan, China / sponsors, IEEE Wuhan Section [and others].*, Wuhan, China. International Conference on Management and Service Science (2010 : Wuhan, China).

Cormier, B. R., Qi, R., Yun, G., Zhang, Y., and Sam Mannan, M. (2009). Application of computational fluid dynamics for LNG vapor dispersion modeling: A study of key parameters. *Journal of Loss Prevention in the Process Industries*, **22**(3), 332–352.

Dawes, W. N., Dhanasekaran, P. C., Demargne, a. a. J., Kellar, W. P., and Savill, a. M. (2001). Reducing Bottlenecks in the CAD-to-Mesh-to-Solution Cycle Time to Allow CFD to Participate in Design. *Journal of Turbomachinery*, **123**(3), 552.

Eastman, C., Fisher, D., Lafue, G., Lividini, J., Stoker, D., and Yessios, C. (1974). An Outline of the Building Description System (Research Report 50). Technical report, Institute of Physical Planning, Carnegie-Mellon University, Pittsburgh, PA.

Fothergill, C. E., Roberts, P. T., and Packwood, A. R. (2002). Flow and dispersion around storage tanks. A comparison between numerical and wind tunnel simulations. *Wind and Structures, An International Journal*, **5**(2-4), 89–100.

Franke, J., Hirsch, C., Jensen, A. G., Krus, H., Schatzmann, M., Westbury, P., Miles, S., Wisse, J., and Wright, N. G. (2004). Recommendations on the Use of CFD in Wind Engineering. Technical report, European Science Foundation COST Office.

Gao, J. and Fischer, M. (2008). Framework and Case Studies Comparing Implementations and Impacts of 3D/4D Modeling Across Projects. *Environmental Engineering*, (March).

Gavelli, F., Bullister, E., and Kytomaa, H. (2008). Application of CFD (Fluent) to LNG spills into geometrically complex environments. *Journal of hazardous materials*, **159**(1), 158–68.

Ghebreyesus, T. A. (2020). WHO Director-General's opening remarks at the media briefing on COVID-19 - 11 March 2020. From https://tinyurl.com/62jcf8s, Accessed on 2020-04-21.

Gopalaswami, N., Kakosimos, K., Zhang, B., Liu, Y., Mentzer, R., and Mannan, M. S. (2017). Experimental and numerical study of liquefied natural gas (LNG) pool spreading and vaporization on water. *Journal of Hazardous Materials*, **334**, 244–255.

Hanna, S. R., Hansen, O. R., and Dharmavaram, S. (2004). FLACS CFD air quality model performance evaluation with Kit Fox, MUST, Prairie Grass, and EMU observations. *Atmospheric Environment*, **38**(28), 4675–4687.

Hansen, O. R., Gavelli, F., Ichard, M., and Davis, S. G. (2010). Validation of FLACS against experimental data sets from the model evaluation database for LNG vapor dispersion. *Journal of Loss Prevention in the Process Industries*, **23**(6), 857–877.

Hightower, M., Gritzo, L., Luketa-hanlin, A., Covan, J., Tieszen, S., Irwin, M., Kaneshige, M., Melof, B., Morrow, C., and Ragland, D. (2004). Guidance on Risk Analysis and Safety Implications of a Large Liquefied Natural Gas (LNG) Spill Over Water. Technical Report SAND2004-6258, Sandia National Laboratories, Albuquerque, New Mexico.

Ivings, M. J., Lea, C. J., Webber, D. M., Jagger, S. F., and Coldrick, S. (2013). A protocol for the evaluation of LNG vapour dispersion models. *Journal of Loss Prevention in the Process Industries*, **26**(1), 153–163.

Ji, Y., Cook, M. J., Hanby, V. I., Infield, D. G., Loveday, D. L., and Mei, L. (2007). CFD modelling of double-skin façades with venetian blinds. In *Building Simulation*, pages 1491–1498, Beijing, China. International Building Performance Simulation Association.

Kawaguchi, T. and Oguni, K. (2016). Automatic Conversion of Visually Consistent Digital Maps to Conforming Geometry for Computational Fluid Dynamics. *Journal of Computing in Civil Engineering*, **30**(2), 1–10.

Kim, J., Jung, E., and Kang, S. (2015). Large eddy simulation of hydrogen dispersion from leakage in a nuclear containment model. *International Journal of Hydrogen Energy*, **40**(35), 11762–11770.

Koopman, R. P. and Ermak, D. L. (2007). Lessons learned from LNG safety research. *Journal of Hazardous Materials*, **140**(3), 412–428.

Li, J., Abdel-jawad, M., and Ma, G. (2014). New correlation for vapor cloud explosion overpressure calculation at congested configurations. *Journal of Loss Prevention in the Process Industries*, **31**(1), 16–25.

Li, J., Hernandez, F., Hao, H., Fang, Q., Xiang, H., Li, Z., Zhang, X., and Chen, L. (2017). Vented Methane-air Explosion Overpressure Calculation—A simplified approach based on CFD. *Process Safety and Environmental Protection*, **109**, 489–508.

Li, L., Ma, Y., and Lange, C. F. (2016). Association of Design and Simulation Intent in CAD/CFD Integration. *Procedia CIRP*, **56**(2014), 1–6.

Luketa-Hanlin, A., Koopman, R. P., and Ermak, D. L. (2007). On the application of computational fluid dynamics codes for liquefied natural gas dispersion. *Journal of hazardous materials*, **140**(3), 504–17.

Oberkampf, W. L. and Trucano, T. G. (2002). Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, **38**, 209–272.

Porter, S., Tan, T., Tan, T., and West, G. (2014). Breaking into BIM: Performing static and dynamic security analysis with the aid of BIM. *Automation in Construction*, **40**, 84–95.

Porter, S. R. (2014). *Breaking Into BIM - Physical Security Simulation Utilising Building Information Models*. Master's thesis, Curtin University, Perth, Western Australia.

Qi, R., Ng, D., Cormier, B. R., and Mannan, M. S. (2010). Numerical simulations of LNG vapor dispersion in Brayton Fire Training Field tests with ANSYS CFX. *Journal of hazardous materials*, **183**(1-3), 51–61.

Rong, L., Nielsen, P. V., Bjerg, B., and Zhang, G. (2016). Summary of best guidelines and validation of CFD modeling in livestock buildings to ensure prediction quality. *Computers and Electronics in Agriculture*, **121**, 180–190.

Sun, B., Utikar, R. P., Pareek, V. K., and Guo, K. (2013). Computational fluid dynamics analysis of liquefied natural gas dispersion for risk assessment strategies. *Journal of Loss Prevention in the Process Industries*, **26**(1), 117–128.

Sun, B., Guo, K., and Pareek, V. K. (2014). Computational fluid dynamics simulation of LNG pool fire radiation for hazard analysis. *Journal of Loss Prevention in the Process Industries*, **29**(1), 92–102.

Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., and Shirasawa, T. (2008). AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, **96**(10-11), 1749–1761.

van Hooff, T. and Blocken, B. (2010). Coupled urban wind flow and indoor natural ventilation modelling on a high-resolution grid: A case study for the Amsterdam ArenA stadium. *Environmental Modelling and Software*, **25**(1), 51–65.

Versteeg, H. and Malalasekera, W. (2007). *Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education.

Vílchez, J. A., Villafañe, D., and Casal, J. (2014). Predicting the flammable region reach of propane vapor clouds. *Journal of Loss Prevention in the Process Industries*, **29**(1), 1–7.

Vuorinen, V. A. U., Hellsten, A. F. M. I., Karvinen, A. V. T. R. C. o. F., Sironen, T. U. o. H., and Råback, P. C. . F. I. e. f. S. L. (2020). Researchers modelling the spread of the coronavirus emphasise the importance of avoiding busy indoor spaces. From https://www.aalto.fi/en/news/researchers-modelling-the-spread-of-the-coronavirus-emphasise-the-importance-of-avoiding-busy, Accessed on 2020-04-21.

Yang, P., Tan, X., and Xin, W. (2011). Experimental study and numerical simulation for a storehouse fire accident. *Building and Environment*, **46**(7), 1445–1459.

Zhang, S., Wu, Z., Zhang, R., and Kang, J. (2012). Dynamic numerical simulation of coal mine fire for escape capsule installation. *Safety Science*, **50**(4), 600–606.

Appendix A

LMSimp Journal File

```
define us-de compiled-functions compile,,"new_velocity.c",,
define us-de compiled-functions load,
;define boundary-conditions velocity-inlet north no no yes yes yes yes udf inlet_x_velocity::libudf no 0 no 306.65 no yes 1 0.1 yes no 0

file read-cas "LMSimp.cas.gz"
file read-data "LMSimp.cas.gz"

;Should cause it to overwrite older files.
;file set-batch no yes no
file confirm-overwrite no
file auto-save retain yes
file auto-save case-freq if-case-is-modified
file auto-save data-freq 10
file auto-save root-name "/home/12645503/LMSimp/data/LMSimp.cas.gz"
file auto-save append-file-name-with flow-time 2

;; Set dynamic refine
/adapt set min-number-cells 0
/adapt set max-number-cells 2000000
/adapt set max-level-refine 3

/adapt set coarse yes
/adapt free yes

/adapt mwg viscosity-ratio iso-value 0 5000 0 yes 20

solve set reporting-interval 2
solve report-files edit report-mass-rfile print yes q
solve report-files edit report-mole-rfile active no q

solve report-files edit report-mass-rfile freq 10 q
solve set time-step 0.01
solve du 1000 1000

file write-case-data "/home/12645503/LMSimp/data/LMSimpTS0.01.cas"

solve report-files edit report-mass-rfile freq 5 q
solve set time-step 0.02
solve du 1000 100

file write-case-data "/home/12645503/LMSimp/data/LMSimpTS0.02.cas"

solve report-files edit report-mass-rfile freq 2 q
solve set time-step 0.05
solve du 600 100

file write-case-data "/home/12645503/LMSimp/data/LMSimpTS0.05.cas"

solve report-files edit report-mass-rfile freq 1 q
solve set time-step 0.1
solve du 600 100

file write-case-data "/home/12645503/LMSimp/data/LMSimpTS0.10.cas"

```

Appendix B

LMSimp PBS Batch File

```
#!/bin/bash
# This is the name that appears in qstat
#PBS -N SRP_LMSimp
# The number of nodes.
#PBS -l nodes=3:ppn=8

# Time requested.
#PBS -l walltime=1800:00:00
#
###
#PBS -M stuart.r.porter@postgrad.curtin.edu.au

### send me email when job begins
#PBS -m abe
### send me email when job end
### PBS -m e
### send me email when job aborts (with an error)
### PBS –m a


echo Working directory is $PBS_O_WORKDIR
cd $PBS_O_WORKDIR

. /etc/bashrc

# Get the path to Fluent
module load fluent/v190

# Tell Fluent where the licence server is.
export ANSYSLI_SERVERS=2325@curtin-swlic01.curtin.edu.au
export ANSYSLMD_LICENSE_FILE=1055@curtin-swlic01.curtin.edu.au
# export ANSYSLM_LICENSE_FILE=1055@curtin-swlic01.curtin.edu.au
# Run fluent
# PBS_NODEFILE is set by PBS.
# PBS_NODEFILE contains a list of the hosts for MPI Fluent.
cpus=`cat ${PBS_NODEFILE} | wc -l`
#

fluent 3ddp -ssh -t${cpus} -cnf=${PBS_NODEFILE} -g -i LMSimp.jou -cflush > output_file

```