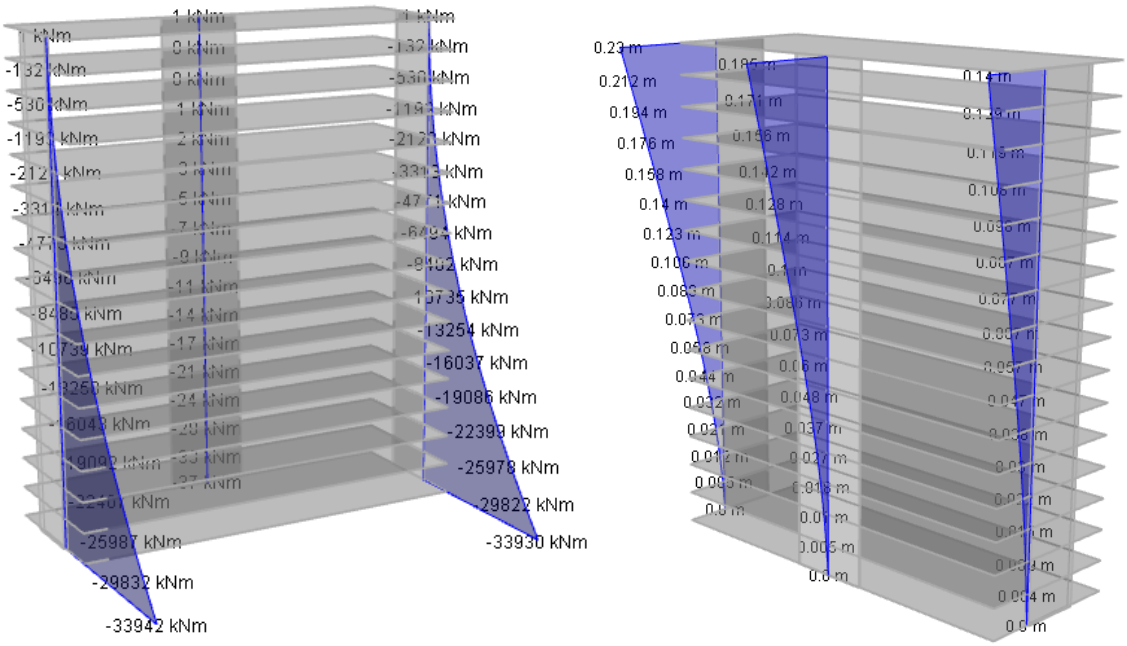
Master's thesis

StructuralComponents 6

An early-stage design tool for flexible topologies of mid-rise concrete buildings



by

Leah Dierker Viik

TU Delft

StructuralComponents 6:

An early-stage design tool for flexible topologies of mid-rise concrete buildings

by

Leah Dierker Viik

Master thesis submitted to the Delft University of Technology
in partial fulfilment of the requirements for degree of

Master of Science
in Building Engineering
Faculty of Civil Engineering and Geosciences

in coordination with White Lioness technologies

To be defended publicly on 27 September 2019

Student number: 4614224

Thesis committee:	Prof.dr.ir. J.G. Rots (chairperson)	TU Delft
	Dr.ir. J.L. Coenders	White Lioness technologies
	Dr.ir. P.C.J. Hoogenboom	TU Delft
	Ir. S. Pasterkamp	TU Delft

TU Delft

Preface

This report is the result of the thesis project of Leah Dierker Viik for a Master's degree in Building Engineering from the Delft University of Technology. This thesis is part of an ongoing project "StructuralComponents" focused on the development of early-stage design tools for buildings. This research was developed in collaboration with White Lioness technologies. The graduation committee consisted of the following members:

Prof.dr.ir. J.G. Rots

Faculty of Civil Engineering and Geosciences
Department of Structural Mechanics
Room 6.61, Stevinweg 1, 2628 CN Delft
J.G.Rots@tudelft.nl

Dr.ir. J.L. Coenders

White Lioness technologies
Founder and CEO
Van Diemenstraat 118, 1013 CN Amsterdam
jeroencoenders@white-lioness.com

Dr.ir. P.C.J. Hoogenboom

Faculty of Civil Engineering and Geosciences
Department of Structural Mechanics
Room 6.44, Stevinweg 1, 2628 CN Delft
P.C.J.Hoogenboom@tudelft.nl

Ir. S. Pasterkamp

Faculty of Civil Engineering and Geosciences
Department of Structural Design/Building Engineering
Room 6.46, Stevinweg 1, 2628 CN Delft
S.Pasterkamp@tudelft.nl

Acknowledgements

Creating the contents of this report was a challenging process, but ultimately a rewarding process that allowed me to learn many new things and grow as a person. There are many people I would like to thank in the writing of this thesis. I would like to thank my graduation supervisors, Jan Rots, Jeroen Coenders, Pierre Hoogenboom and Sander Pasterkamp for their continued support throughout this project. Especially thank you to Jeroen Coenders, who introduced me to the topic StructuralComponents and provided me the opportunity to be involved in the community at White Lioness technologies.

I would also like to thank Babette Hohrath, who took the time to review my report and discuss my project with me, and Dion Jansen, who helped me with the development of the tool. Also thank you to Nikoletta Christidi who helped me figure out how to get my SymPy script working properly. Finally, thank you to my friends and family for continually supporting and encouraging me throughout this process. Thank you especially to my Dad, who spent several hours reviewing my final report and providing valuable advice on improvements.

Summary

The process of designing a building is often inefficient. One reason for this is that the design process involves many different actors with conflicting interests that are difficult to resolve. Another reason is that little information is known about the design in the early stages, often leading to bad decisions made in these stages which are time-consuming and costly to fix later (MacLeamy, 2010). Computational tools have the power to integrate different actors of the building design process together and to provide designers with more information in early stages of design, if used properly. However, current computational tools in the building industry are not being used to their full potential (Coenders, 2011). Therefore, it is important to work towards developing computational tools that can improve the building design process, particularly at the early stages of design.

StructuralComponents is ongoing project that focuses on the development of early-stage design tools for buildings. The goal of the project is to develop a tool that allows an engineer to analyse and validate a conceptual building design, and gives the engineer confidence in their chosen design. Five versions of StructuralComponents have been previously developed. The most recent version, StructuralComponents 5 (Hohrath, 2018), focused on developing an early-stage design tool for mid-rise concrete buildings constructed of stackable "building blocks" made of pre-defined configurations of shear walls and cores ("stability elements"). A limitation of StructuralComponents 5 is that these building blocks cannot be connected horizontally, limiting the variation in building designs that can be modelled with the tool. The purpose of StructuralComponents 6 is to address this limitation by developing a new version of StructuralComponents that can be used for the early-stage design of concrete buildings with flexible configurations of stability elements on a horizontal plane.

To achieve this goal, a new "calculation method" is developed that can be applied to varying configurations of stability elements. In this calculation method, stability elements are viewed as beams which are fixed to the ground by rotational springs (with infinite translational stiffness) and free at the top. The floors connecting the stability elements are considered as infinitely rigid.

The calculation method consists of a system of three differential equations representing the force and bending moment equilibrium of a set of rigidly-connected flexural beams (representing the stability elements). Each equation in the system can be modified to include more or fewer stability elements as needed. The system of equations can be solved with the boundary conditions specified in the previous paragraph to determine the deflection, shear force and bending moment along the height of each stability element. The system of equations for "i" stability elements is shown below:

$$EI_{x1}\frac{d^4u_{x1}}{dz^4} + \ \ldots + \ EI_{xi}\frac{d^4u_{xi}}{dz^4} = \ F_{x,external}$$

$$EI_{y1}\frac{d^4u_{y1}}{dz^4} + \cdots + EI_{yi}\frac{d^4u_{yi}}{dz^4} = \ F_{y,external}$$

$$\left(a_1 \cdot EI_{y1}\frac{d^4u_{y1}}{dz^4}\right) - \left(b_1 \cdot EI_{x1}\frac{d^4u_{x1}}{dz^4}\right) + \cdots + \left(a_i \cdot EI_{yi}\frac{d^4u_{yi}}{dz^4}\right) - \left(b_i \cdot EI_{xi}\frac{d^4u_{xi}}{dz^4}\right) = M_{external}$$

In the system of equations above, a_i and b_i represent the distance of stability element "i" away from the rotational centre of the system in the x and y-directions, respectively.

The calculation method is tested for accuracy by comparing results for deflection, shear force and bending moment against results from finite element analyses, for various "test configurations". It is determined that the calculation method produces sufficiently accurate results for buildings with minimal out-of-plane floor effects (buildings with pre-cast floors).

After the calculation method is developed, it is integrated into a Python script wherein the number and location of stability elements can be automated. A user interface for the tool is developed in Grasshopper. Four components are developed in Grasshopper: a "Construct shear wall" and "Construct floor" component to construct the floorplan, a "Calculator" component to calculate the results of the system, and a "Visualiser" component to visualise the building design and results. The construction of a model with three shear walls is shown in Figure 1.

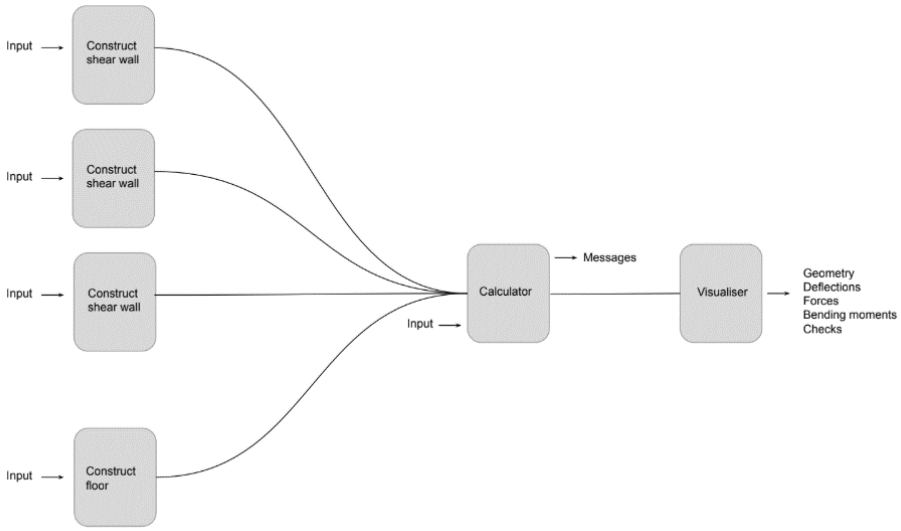


Figure 1 Model construction in StructuralComponents 6

The user can modify the number of shear walls in their floorplan simply by adding a new "Construct shear wall" component. This gives the user the flexibility to analyse building plans with various numbers and positions of shear walls.

To validate the tool, a case study is performed wherein the EWI building at the Delft University of Technology is modelled with the tool, with a simplified, rectangular floorplan. The floorplan used in the case study is shown in Figure 2.

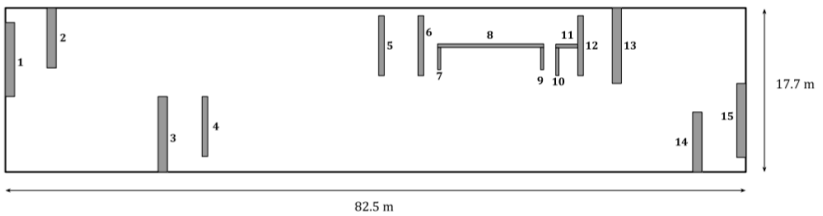


Figure 2 Simplified floorplan of EWI building used for case study

The case study shows that the tool can be successfully applied to a building with complex configurations of stability elements.

Table of Contents

1 Introduction

1.1 Motivation

The complete process of designing a building – from conception all the way to construction – is a complex procedure involving many different stakeholders acting in various roles and performing a wide range of tasks. Different stakeholders in the building design often have different interests, and resolving these conflicting interests can require a lot of correspondence and time. To make matters worse, there is often a disconnect between the flexibility of the design and the amount of information available for making design decisions. Early in the design, it is easy to make changes, however there is little information available to make informed design decisions. As the design progresses, more information becomes available, but the cost of making changes to the design increases As a result of this incongruity between the flexibility of the design and the amount of information available, poor decisions are often made in the early design stages which can be costly and time-consuming to correct later in the design. This means that the design team must work harder at a later stage in the design to fix bad decisions made earlier in the design. Ideally, the effort of the design team should be concentrated in the earlier stage in the design, to prevent these poor decisions from occurring in the first place (MacLeamy, 2010). This relationship is illustrated in Figure 1.1.

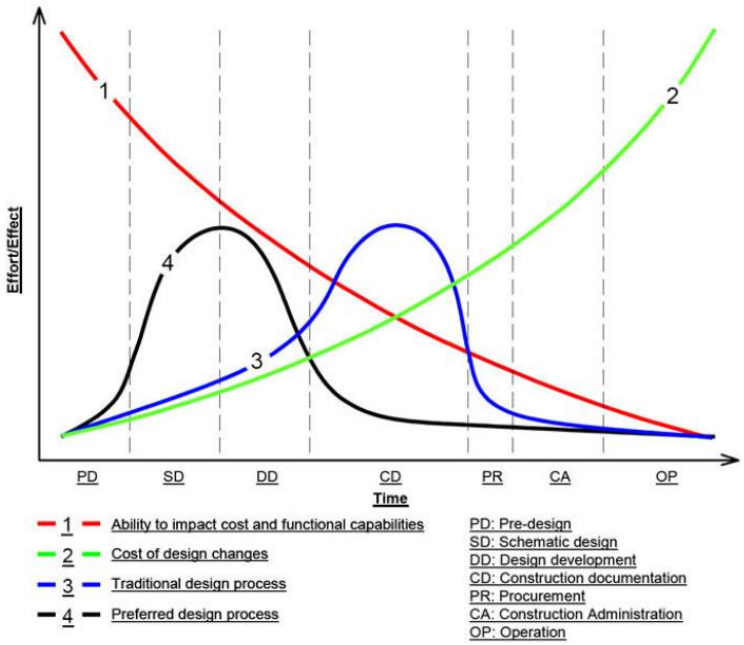


Figure 1.1 MacLeamy curve (CURT, 2004)

To improve the efficiency of the design process, some major changes to the traditional system are needed. Primarily, the efforts of different actors in the design process should be coordinated to avoid miscommunications that can negatively impact the design. Additionally, more information should be made available earlier in the design process to allow designers to make more informed decisions at this stage (MacLeamy, 2010). But how can these goals be achieved?

To answer this question, it is important to look at the role of computational technology in the building design process. The use of computation can lead to better control of data and information, better prediction of structural behaviour, and reuse of existing knowledge in the design process, all features which can greatly improve the quality of a design. Furthermore, computation has the potential to transform the building design process, if used effectively. Computational methods can be used to access more information in the early design phase, leading to better decisions made at this point, improving the quality of the design thereafter. The use of computation can also provide more flexibility in later design stages (Coenders, 2011).

Over the past few decades, a large number of different computational tools have been developed to aid engineers and architects with project coordination and the design and analysis of structures. These include analysis tools such as FEM software and graphics statics tools, and more integrated tools such as BIM software. However, despite the introduction of many new computational tools, there continue to be inefficiencies during building design (Coenders, 2011).

In his dissertation NetworkedDesign (2011), Coenders notes some observations about the use of computational technology in the building industry. Firstly, most programs that exist tend to support only one particular part of the design process, and few technologies have been developed that support the overall design process. Secondly, the adoption of computation in the building industry has been slow. Finally, other industries have overtaken the building industry in pioneering these new forms of technology. From these observations, it is clear that computational technology is not being used to its full potential in the building industry. The slow adoption of computation in this industry indicates that engineers are lacking motivation to adopt these new tools; this likely suggests that current computational tools are missing some inherent qualities or characteristics that they need or desire (Coenders, 2011).

It is therefore evident that research is needed in the field of computational tools for the building industry. New tools need to be developed that fully meet the needs of the industry and will be accepted by designers. In particular, research is needed in the field of early-stage design tools for the building industry. As expressed in the MacLeamy curve (CURT, 2004), early-stage design is particularly important because the decisions made in this stage have a great impact on the whole building design to follow. Decisions made at this stage can determine whether or not the project is successful.

This thesis is focused on contributing to the research field of tools for the early-stage design of buildings. The goal of the project is to develop a digital design tool that provides the structural engineer with an easy and intuitive method to design, analyse and compare conceptual designs for mid-rise concrete buildings. The tool should provide the engineer with insightful results that can be used for collaboration with other actors of the building design process, such as the architect and stakeholders. This project is part of an overarching project “StructuralComponents” with TU Delft and White Lioness technologies.

1.2 Background

StructuralComponents is an ongoing project that is focused on the creation of a conceptual design tool for building design. The main goal of StructuralComponents is to develop a tool that allows an engineer to easily create and structurally validate a conceptual building design, to support the

future detailed building design. Further goals of StructuralComponents are to provide a tool that is intuitive to the engineer to use, fully supports the varying logic of the design process, and importantly provides insight to the engineer to give them confidence in their design. Many existing design tools are like "black boxes", where the user enters input and receives output but does not completely understand what happens in between. StructuralComponents is meant to be insightful and allow the user to maintain a feeling of control over their design.

Another important goal of StructuralComponents is to aid with collaboration between different actors in the conceptual design process. By providing a clear and insightful model of the conceptual design, StructuralComponents can be used by the engineer and architect to integrate their design ideas together and articulate a clear, coordinated plan.

The first version of StructuralComponents was created in 2008 by Breider as part of a MSc thesis project at TU Delft. Since then, four other MSc projects have been done with TU Delft on further development of StructuralComponents. The following section will provide a summary of the previous work on StructuralComponents.

StructuralComponents 1 (2008)

The first version of StructuralComponents was developed by Breider in 2008 and focuses on the conceptual design of tall buildings. The tool was implemented as a plugin for Bentley GenerativeComponents (Bentley Systems, Incorporated, 2019)[1]. In Breider's tool, the user can create and analyse a two-dimensional model of a tall building composed of cores, columns and outriggers in various combinations. The tool includes a user interface where the user can develop their structural model and a dashboard where the user can visualise the performance results of their model (Breider, 2008 as cited in Rolvink, 2009). The user interface contains five different component types that are used to assemble and analyse the model, as follows:

- Structural Model Components (SMC): Define geometry of the structural elements
- Mechanical Assembly Components (MAC): Define boundary conditions and construct stiffness matrix for the structural elements
- Load Model Components (LMC): Apply loads
- Analysis Components (AC): Perform analysis
- Result Processing Components (RPC): Visualisation of model performance

Figure 1.2 shows a visualisation of the model assembly in Structural Components 1 (Breider, 2008 as cited in Rolvink, 2009).

[1] GenerativeComponents (Bentley Systems Incorporated, 2019) is a digital parametric and associative design (PAD) tool. PAD tools are described in more detail in Chapter 3.

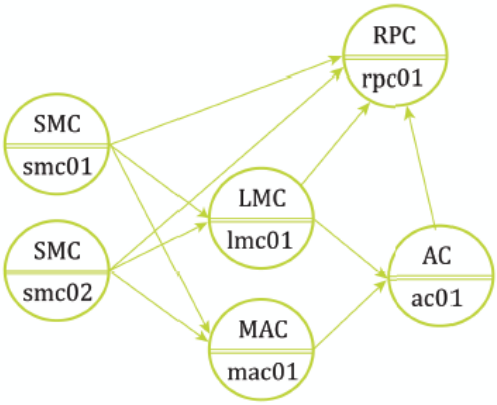


Figure 1.2 Assembly of structural components in StructuralComponents 1 (Breider, 2008 as cited in Rolvink, 2009)

After the model is assembled, the performance results of the model are shown to the user on a dashboard, as shown in Figure 1.3.

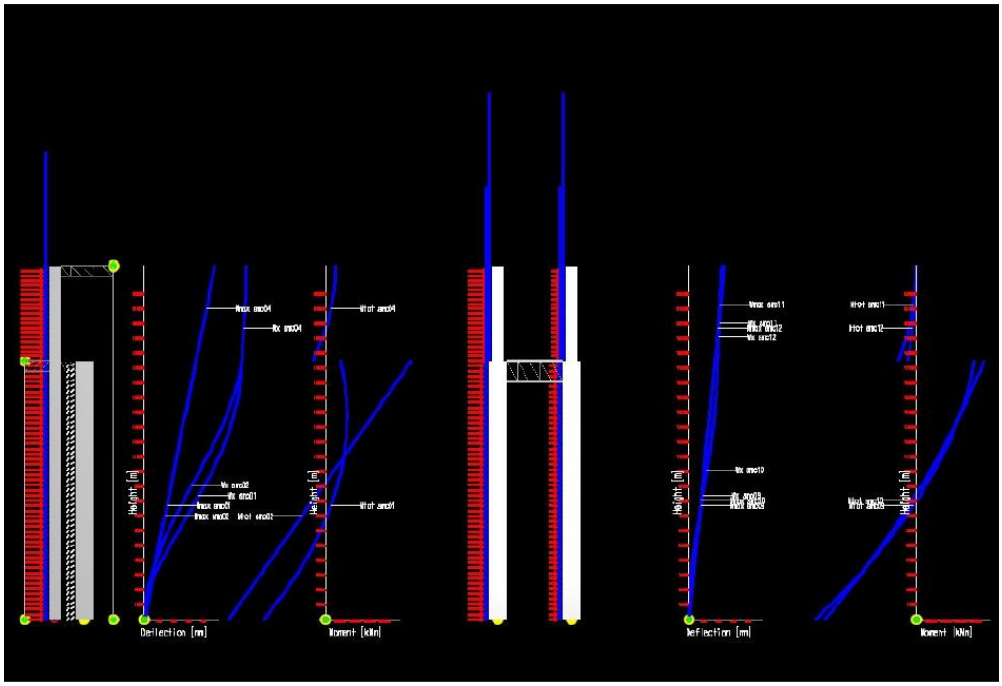


Figure 1.3 Dashboard view of StructuralComponents 1 (Breider, 2008 as cited in Rolvink, 2009)

StructuralComponents 2 (2009)

The second version of StructuralComponents was developed by Rolvink in 2009. The goal of this project was to improve the usability of StructuralComponents 1 by both allowing for the creation and analysis of conceptual designs in 3D and providing a new, more modular framework allowing for a more flexible workflow (Rolvink, 2009).

The toolbox itself consists of two parts: an interface model and a framework. The interface model allows the user to create the geometry of their conceptual design and visualise it. It is executed using PAD software such as Grasshopper (Davidson, 2019)[2] or GenerativeComponents (Bentley Systems Incorporated, 2019). The framework contains "the working code that corresponds to the elements declared in the interface" and is not visible to the user. It consists of a structural model, where the interface input is interpreted structurally, and an analysis model, which contains a solver and post-processing application. The performance results of the model are visualised on a dashboard in a similar manner to StructuralComponents 1. An image of the interface of StructuralComponents 2 is shown in Figure 1.4 (Rolvink, 2009)

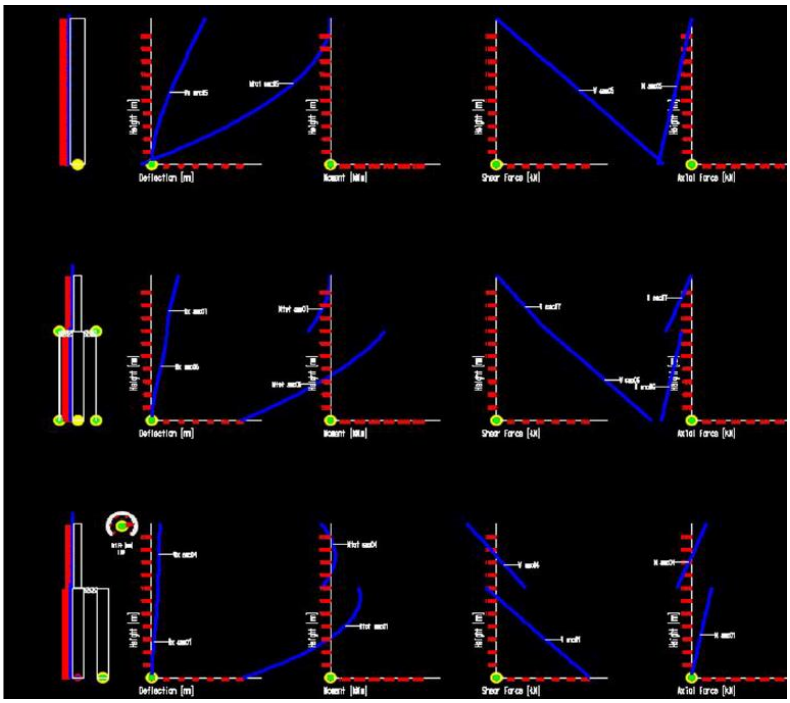


Figure 1.4 Dashboard results in StructuralComponents 2 (Rolvink, 2009)

An important feature of StructuralComponents 2 is that it provides the user with two different methods to create and analyse a conceptual model. This capability was implemented to reflect the bi-directional nature of the design process. The user can choose which of the two methods they would like to use to create their design, based on what aspects of design they find the most important. The two different methods are as follows:

1. The user creates a building model by adding/connecting components, the tool performs structural analysis of the design and then the user can change the design until it meets the structural design criteria
2. The user defines the structural design criteria and then the tool generates multiple structural models that meet those criteria, which the engineer can then evaluate and modify (Rolvink, 2009)

[2] Refer to Section 3.2 for a detailed description of Grasshopper (Davidson, 2009)

StructuralComponents 3 (2013)

The third version of StructuralComponents was developed in 2013 by van de Weerd. The goal of this version was to create a new prototype of StructuralComponents with similar capabilities of the previous versions, but with increased emphasis on design exploration and the generation of different design alternatives. Further goals of StructuralComponents 3 were to perform structural analysis using FEM, and to develop a new software architecture that would eliminate the external software dependencies that exist in StructuralComponents 1 and 2 (van de Weerd, 2013).

To develop design alternatives, StructuralComponents 3 proposes a system that moves between composition and abstraction of a design. The user first creates a parametric model using functional components and defines the upper and lower limits of the parameters. The user also defines performance metrics that they would like their building model to meet. The tool then "abstracts" the user's model through the range of the defined parameters and repeatedly produces samples with varying parameter values which are analysed and compared against the performance metrics until the best parameter values are found; this model is then returned to the user. If the user wants to change a parameter, this process in repeated to find new optimum parameter values (van de Weerd, 2013).

The new software is based on a client-server setup, wherein the client-side system includes the user interface and the visualisation of output, and the server-side system is where the generation of alternatives and structural analysis is performed. Only the server-side system was developed for StructuralComponents 3 (van de Weerd, 2013).

StructuralComponents 4 (2015)

The fourth version of StructuralComponents was developed in 2015 by Bovenberg. This version of StructuralComponents moves away from the design of high-rise buildings and seeks instead to provide a tool for the conceptual design of more general building typologies. The main aim of the StructuralComponents 4 was to develop a tool to encompass the entire "design story" of a building by including all the "models, simplifications, reasoning, alternatives, and scenarios used to develop and justify the design" (Bovenberg, 2015).

The software architecture of StructuralComponents 4 consists of four interrelated components: a user interface, a conceptual building model, analysis tools and a computational engine. The conceptual building model defines how the information is structured, the analysis library provides a number of different options to analyse the building model, the computational engine ensures all information in the system is constantly up-to-date and the user interface defines how the user interacts with the components (Bovenberg, 2015).

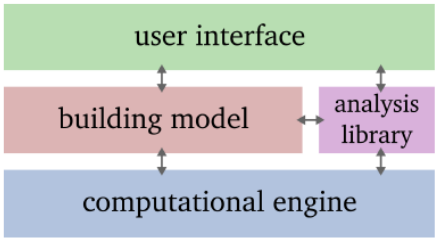


Figure 1.5 Four main components of StructuralComponents 4 (Bovenberg, 2015)

StructuralComponents 4 aims to model the conceptual design of a building by incorporating aspects of the conceptual design process into its framework. The tool is designed to allow the user to easily create and compare different structural models at different levels of abstraction, and to allow the user to have as much design freedom as possible. At the basis of the structural model are "blank slate" components: blank components to which the designer can assign attributes (such as length, material, stiffness, etc.) and connect together like Lego blocks. Alternative versions of the same component can be used in the model (allowing for parallel versions of the same model). Individual components can be related to each other hierarchically to create larger components. Different "analysis models" can be applied to the components/groups of components (for example, to calculate the maximum moment on a beam or to calculate the required thickness of a floor, given a certain load). In this way, the user can choose the analysis method that is most appropriate for their needs (Bovenberg, 2015).

StructuralComponents 5 (2018)

The fifth and most recent version of StructuralComponents was developed by Hohrath in 2018. This version of StructuralComponents is focused specifically on the conceptual design of mid-rise concrete buildings consisting of cores, shear walls and floors. The tool was implemented as a group of components in Grasshopper, scripted in Python (Python Software Foundation, 2019; Hohrath, 2018).

At the core of StructuralComponents 5 are three "building blocks" which represent three different topologies of a concrete mid-rise building containing shear walls, cores and floors (shown in the image below). The topology of each building block is fixed, but the user can define specific attributes for each building block, such as the thickness of the walls/cores or the distance between the stability elements. The user can create more complicated building forms by stacking these building blocks on top of one another. The structural analysis of the building blocks is done by considering the building blocks as super elements (see Appendix A) and calculating the structural behaviour using symbolic differentiation (which allows for real-time visualisation of the structural behaviour; Hohrath, 2018).

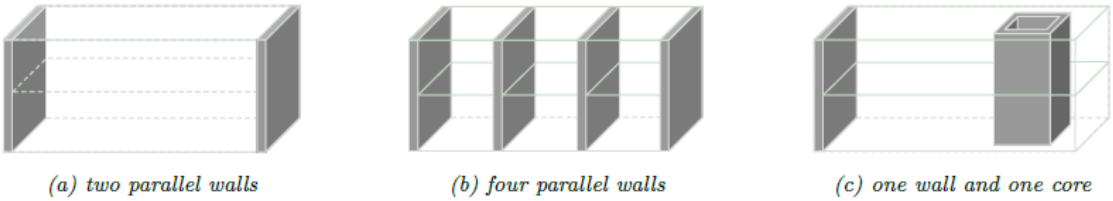


Figure 1.6 Structural building blocks from StructuralComponents 5 (Hohrath, 2018)

The framework of StructuralComponents 5 includes several different Grasshopper components that the user can connect together to create and analyse their conceptual model. The user begins with a "start" component where they can define the position and orientation of the stability elements. They then use this start component as input into a building block component, which creates one of the three building blocks shown above. They then use their geometry as input into an analysis component which outputs results to a viewer (Rhinoceros; Robert McNeel & Associates, 2019). An outline of this framework is shown in Figure 1.7 (Hohrath, 2018).

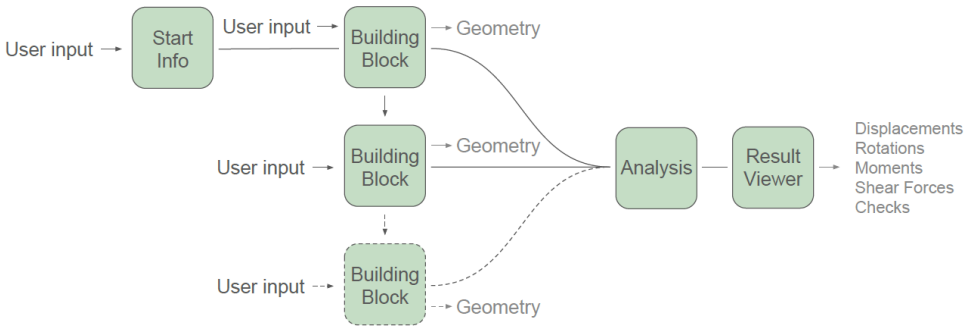


Figure 1.7 Overview of framework for StructuralComponents 5 (Hohrath, 2018)

2 Problem Definition

2.1 Overview

The principal goal of this Master's thesis is to develop a new version of StructuralComponents 5 with increased flexibility of use. As mentioned in the previous section, StructuralComponents 5 involved the creation of three "building blocks", which the user can modify and stack vertically together to create more complex structures. A limitation of StructuralComponents 5, however, is that the building blocks cannot be connected horizontally. This means that the number of different building topologies that can be represented in this tool is limited.

To overcome this limitation, StructuralComponents 6 proposes a different approach. Rather than developing discrete building blocks made of pre-arranged core/shear walls as was done in StructuralComponents 5, this project focuses on determining a way to allow the user to create their own custom arrangements of shear walls and cores (referred to collectively as "stability elements") on a building plan. In this approach, the individual stability elements and floors are considered as discrete elements that can be modified by the user and connected together to create a complete structure.

In order to give proper validation of the design that the user has created, the tool needs to have certain capabilities. Primarily, the tool must provide structural validation to ensure that the structure is adequate for the loads that are applied on it.

For structural validation of the design, the tool must perform the following checks:

1. Strength: The stress in the building must not exceed the available strength
2. Stability: There must be no tension in the foundation
3. Stiffness: The deformation of the building must be limited to allowable limits

The tool also needs to provide the user with visualisation of their design and its structural behaviour. The tool should provide the user with a clear visual representation of what their design looks like. After structural analysis is performed, the tool should show graphs of the deflection/moment/shear force/normal force along the structure. The tool also needs to visualise warnings and failures to alert the user to problems with their design.

As in StructuralComponents 5, this new tool will be developed as a group of components in Grasshopper. The method of structural analysis will be developed in Python, and incorporated into the Grasshopper components.

2.2 Research Questions

The main research question that the project addresses is the following:

> How can the tool StructuralComponents 5 be modified to allow the user to design buildings with custom configurations of stability elements?

More specific research questions are shown as follows:

1. What requirements are needed for the tool for the conceptual design phase?

- What does the conceptual design process look like, from the engineer's perspective?
- How much information is needed by the engineer at the conceptual design stage? What simplifications are made to the building design during the conceptual stage?
- What conceptual design tools already exist, and what characteristics do they have?

2. How can the tool perform structural validation of the custom configuration?

- Is the method of structural analysis employed in StructuralComponents 5 (super-element method) appropriate for this new tool?
- What are the limitations to the accuracy of analysis when the floors are considered to be infinitely rigid?
- How can the structural analysis be automated using Python?

3. How should the user interface be developed for the tool?

- What assumptions should be made about the knowledge of the user? How does this affect the capabilities/limitations of the tool?
- How can the user easily define their desired floorplan? How are stability elements and floors defined in Grasshopper/Rhinoceros, and how do they interact with each other?
- How are checks and warnings visualised?

2.3 Objectives

Based on the research questions, objectives for the project are developed. The main objective of the project is as follows:

> *Develop a tool, based on StructuralComponents 5, that provides early-stage structural validation for flexible topologies of concrete mid-rise buildings made of shear walls, cores and floors.*

Flexible topologies are defined by the ability to place stability elements freely, as discussed before. To narrow down this main objective, four sub-objectives are also developed, which are listed below:

1. Investigate the requirements of the tool to allow for the conceptual design of a mid-rise building

This objective intends to develop a complete framework that the tool will be based on. First, the conceptual design process is studied. Additionally, other similar technologies in this field are studied in order to gain a good idea of the typical goals of the software and how the user interacts with the software. At the end of the phase, a rough framework of conceptual design is developed, and key characteristics of the framework are used as criteria in further development of the tool.

2. Develop a flexible calculation method to determine the forces and deflections to a sufficient degree of accuracy for varying arrangements of shear walls, cores and floors

The aim of this objective is to develop a calculation method for analysing mid-rise buildings with custom configurations of stability elements. A method must be developed to calculate the deflection, shear force, bending moment and normal force on the stability elements, for any given configuration of stability elements. Different calculation methods are proposed and judged based on criteria developed in the previous objective.

3. Develop an intuitive user interface for the tool

The focus of this objective is to develop the user interface for the tool. In this stage, the inputs and outputs of the tool are determined, and the calculation method developed in the previous objective is incorporated into the tool. A method to visualise checks and warnings to the user is also developed.

4. Examine the applicability of the tool to a real building design

The final objective of the project is to test the applicability of the tool to an actual building design. To achieve this objective, a case study is performed, wherein an existing building is modelled in the tool. Based on the case study, the suitability of the tool to actual building design can be examined, and necessary improvements for the tool can be determined.

2.4 Scope

To contain the complexity of the project, the scope is limited to similar constraints of StructuralComponents 5. The project focuses specifically on the conceptual design of mid-rise concrete buildings, and the tool will model only the main elements of a horizontal load-bearing system which consists of concrete shear cores, shear walls and floors. Other stability systems that could be used for a mid-rise concrete building, such as a frame structure, will not be considered. Because the building is considered to be mid-rise and not high-rise, dynamic effects on the building's behaviour will be ignored.

To simplify the analysis, the tool will only consider buildings that are prismatic in height.

Due to the time limitations of the graduation project, the development of the user interface is limited to functionality only (not aesthetics). A fully-developed tool is not created, but a set of components that allow the user to create and validate custom building blocks and simple building designs made from these blocks.

2.5 Methodology

This project has been split into four phases based on the objectives:

- Phase 1: Investigate requirements of conceptual design
- Phase 2: Determine method of analysis
- Phase 3: Develop the user interface
- Phase 4: Test the applicability to a real building design

Phase 1 focuses on the initial research and literature review. In this phase, the process of engineering design is investigated to determine the requirements of the building industry for conceptual design tools. Additionally, existing conceptual design tools in the building industry are investigated to determine what options have already been explored, and what the current needs are for the development of new conceptual design tools. At the end of this phase, a rough framework of a conceptual design tool is outlined. This phase is discussed in Chapter 3.

Phase 2 is focused on developing a calculation method for the structural analysis of the tool, and assessing the limitations of this calculation method. In this phase, a calculation method is developed to determine the deflection, shear force, bending moment, and normal force on each stability element, for any given configuration of stability elements. A simplified analysis method is presented, in which the floors are assumed to be infinitely rigid. Three different "test" cases are developed with this assumption, and each test case is validated against a finite element analysis. The limitations of the different test cases are discussed, and the most favourable test case is chosen for use in the tool and further developed. This phase is discussed in Chapter 4.

Phase 3 is focused on developing the user interface of the tool. In this phase, the calculation method developed in the previous phase is incorporated into a Grasshopper component. In this phase, the interaction between the user and the tool is developed. It is determined exactly how the user defines their model, how the user visualised their model and how results and warnings are displayed to them. Because the tool is meant for conceptual design, the user interface must also include qualities that facilitate the conceptual design process. This phase is discussed in Chapter 5.

The final phase, Phase 4, involves the creation of a case study to test the applicability of the tool to a real building design situation. In this phase, the EWI tower at TU Delft is modelled in the tool. The building is analysed in both SLS and ULS for two different wind directions, and requirements for stiffness, strength and stability are checked for the appropriate analyses. The tool is validated on the basis of the ease of model construction, speed of analysis and visualisation. Phase 4 is described in Chapter 6.

3 Conceptual Design

3.1 Conceptual design process

The RIBA Plan of Work 2013 (RIBA, 2013) is the standard model for the building design and construction process in the UK, created by the Royal Institute of British Architects. The RIBA Plan of Work 2013 outlines the general steps that define a typical building construction process. Eight stages of the building design and construction process are defined in the RIBA Plan of Work 2013 as follows:

0. Strategic Definition
1. Preparation and Brief
2. Concept Design
3. Developed Design
4. Technical Design
5. Construction
6. Handover and Close Out
7. In Use

Stage 0, Strategic Definition, is a preliminary stage concerned with the initial orientation of the project. The goal of this stage is to identify the client's rationale in initiating a new building project and the client's strategic objectives and desired outcomes for the project. An initial Project Programme is developed. In this stage, initial considerations of the project team are also made. In Stage 1, Preparation and Brief, the Initial Project Brief is formally compiled. This brief outlines the objectives of the project, the desired outcomes and the budget. In this stage, a feasibility assessment of the site is also performed. The project team is assembled and the roles and responsibilities of each member of the team is defined (RIBA, 2013).

In Stage 2, the Concept Design is created. The Concept Design includes proposals for the structural design, building services, specifications and initial cost estimation. At the end of this stage, the Initial Project Brief is reviewed and redefined as the Final Project Brief. Following the Concept Design is the Developed Design, or Stage 3. The Developed Design involves the development of the Concept Design. At the end of this stage, the architectural, structural and service plans for the project are developed in coordination with the project budget. This process may involve several iterations. In both Stage 2 and Stage 3 of the design, the design team should work in close coordination to make sure all aspects of the design are properly integrated (RIBA, 2013).

Stage 4, the Technical Design, involves further refinement of the Developed Design. In this stage, different members of the design team begin to work independently to develop separate technical designs. Once enough detail is included in the designs of the original design team, specialist subcontractors or suppliers may contribute to the design. In a traditional building project, the building design is completed at the end of this stage (RIBA, 2013).

After Stage 4, the building is constructed. Following the construction of the building is the Handover and Close Out, wherein the building will be handed over by the project team and Building Contract is usually ended. Some tasks may be continued into the In Use stage of the

building, such as reviewing of the project performance and research and development for future projects (RIBA, 2013).

The RIBA Plan of Work 2013 emphasises planning and collaboration in the early design phases of the building project. As previously mentioned, the conceptual design (or Concept Design) phase is particularly important because it defines the path of the project to follow. It is thus important to use disciplined design methodologies supported by effective tools to help ensure that a good design is produced in this phase. Some aspects of the conceptual design phase are outlined as follows.

The conceptual design process varies between different engineers and often does not follow one specific method. Generally, though, this process begins with several rough sketches and notes, wherein the engineer defines important characteristics of the project and considers a number of different solutions to meet these characteristics. From this initial process of sketching, the engineer can identify the key drivers to the design. These key drivers define a specific "design logic" that serves as a framework for the design; it is important to keep track of this design logic because if drivers change later in the procedure, then this design logic may also change. While developing different alternatives, the engineer must concurrently think about how these alternatives should be assessed and ranked. Options are usually ranked based on assessment criteria as specified in the project brief, and this may lead to changes in the brief if the criteria are too restrictive. The engineer may additionally specify further assessment criteria that he/she feels is necessary (Institution of Structural Engineers, 2011).

The development of the conceptual design is a highly iterative process and often very time-consuming. The structural designer often goes through many different sketches and designs before a suitable option is found. Frequently, existing solutions are broken down and reworked to develop improved, unique solutions. Furthermore, during this process there is often a lot of exchange of ideas between the structural engineer and other designers. Depending on the complexity and requirements of the project and the desires of the client, different strategies may be used to select the best design from a group of options. For example, it is possible that only a few "superior" options be developed, and less desirable options be discarded, or alternatively all options may be developed and reviewed (Khandani, 2005; Institution of Structural Engineers, 2011).

From the perspective of the structural engineer, the conceptual design process is difficult to fully characterise because it is complex, subject to change, and varies highly amongst different engineers, teams, and projects. However, several aspects of design have been noted by scientists and engineers.

Braha and Maimon state that evolution is a key element of the design process. As the design progresses, the designer continuously modifies either the current design or the design specifications in accordance to new information/ideas that arise in the design cycle. The design process continues to converge until an acceptable design is determined (Braha et. al, 1997).

Braha and Maimon additionally observe four characteristics of design, as follows:

1. Designers act under "bounded rationality"[3]
2. Design problems are initially ill-structured, and must be further defined through a research process
3. Designers often do not seek the most optimal design but instead look for a design that is satisfactory
4. Design problems are generally so complex that even if an optimal solution exists, it is difficult for the designer to access (Braha et. al, 1997)

One of the challenges in describing the design process is the ubiquity of design; there are often several different ways of designing depending on the design problem and/or designer. Sriram et al. (1989) note four different types of design:

1. Creative design: There is no pre-existing plan for how to solve the design problem. The designer must create a plan by decomposing the problem into a set of requirements and making decisions. The designer uses a divergent thought process.
2. Innovative design: The required steps in solving the design problem are known, but the alternatives at each of these steps is not. The designer must develop these alternatives using principles of their domain of knowledge.
3. Redesign: An already-existing design is changed based on its functional needs.
4. Routine design: There is a pre-existing plan for how to solve the design problem and the alternatives at each step are known. The designer must determine which alternatives best meet the constraints of the problem (Sriram et al., 1989).

In NetworkedDesign, Coenders (2011) notes that exploration of design ideas often occurs on multiple levels of abstraction or composition. Additionally, conceptual design consists of many processes of a different nature; these may be linear, non-linear, iterative, cyclic or chaotic. Some different processes that occur during conceptual design can be listed as follows:

- Parallel investigation of alternatives
- Changing granularity (switching between diversity and convergence)
- Changing focus and goals
- Freezes and states (for example milestone points to synchronise different disciplines)

There are also many ways in which the conceptual design may change. The conceptual design may undergo changes of data, structure, logic or process (Coenders, 2011).

There are several important values in structural engineering that drive the conceptual design process of a structural engineer, as have been outlined in NetworkedDesign (Coenders, 2011). A selection of these values is summarised below:

[3] Bounded rationality is a concept developed in 1956 by Herbert Simon. It refers to the idea that the designer's ability to make decisions is constrained by his/her limitations in time, knowledge, and resources. Bounded rationality posits that true optimisation in a design problem is impossible, since the decision-making ability of the designer is limited (Gigerenzer et al., 2001).

Responsibility

Structural engineers have the primary responsibility for safety in the building. Failure in structural design can lead to devastating and even fatal consequences, generally unlike failure in other disciplines which has smaller consequences. Thus, the structural engineer must ensure that their design is safe.

Confidence

The engineer needs to have confidence in the design before he/she is willing to approve it. In order to have confidence the engineer requires the following:

- Comprehension: The engineer should have full understanding of the methods, calculations, process used to develop a solution
- Insight: The engineer should understand how and why the applied methods work, and the significance of these methods in the overall building design
- Control: The engineer should have control over the design process, and should have control over the knowledge and logic that support the developed designs

If the engineer has confidence, this leads to *trust* in the design.

Justification

The engineer should be able to fully justify the design. This justification should explain "the workings of the structure, how it deals with different scenarios and how it deals with unforeseen situations". All aspects of the full justification story are indicated in Figure 3.1 (Coenders, 2011).

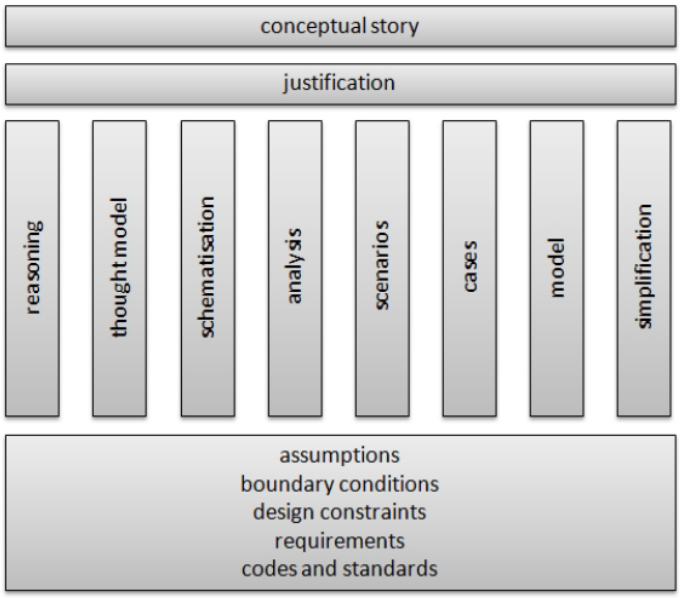


Figure 3.1 Justification story of a conceptional design (Coenders, 2011)

Effective tools can support the conceptual design process, leading to a safer design (responsibility), and better controlled design (confidence) and a well-understood design (justification).

3.2 State-of-the-art in conceptual design

It is important to review the current state of conceptual design tools used in structural engineering and related disciplines. This will help identify challenges, needs, gaps, problems and areas for improvement. Innovation is a gradual process that builds on both a good understanding of the design process and the current state-of-the-art in technology.

Several conceptual design software tools have emerged over the past years. Conceptual design tools differ from traditional design/analysis tools in that they focus on the quick exploration of different design alternatives, rather than detailed calculations and measurements. The purpose of these conceptual design tools is to provide designers with an easy way to brainstorm and review different ideas; they are essentially a digital replacement for the sketching paper traditionally used by designers. In order to provide as much flexibility as possible, conceptual design tools generally focus on high-level representations of structures/objects. By eliminating the constraints of a detailed design, the user of these tools can easily create/modify designs and has more freedom to be creative (Rolvink et al., 2011; Pal, 2014).

Conceptual design tools can lead to a better design by providing the designer with more knowledge to make informed decisions at early stages in design. As already discussed, bad decisions in the conceptual design can lead to huge consequences later that are difficult and expensive to change; conceptual design tools reduce this problem. Conceptual design tools are also very useful for collaboration because they provide a way to visualise the conceptual design. Visualisation enables the designer to easily discuss their design with other actors in the design process (such as the architect, client, or investors) and make changes as needed. Another benefit of conceptual design tools is that they help speed up the conceptual design process. In recent years, the design process has sped up considerably, so the time allotted to create a conceptual design has shrunk. There is a need to make the conceptual design phase as efficient as possible, and conceptual design tools can increase the efficiency of the design process (Rolvink et al., 2011; Pal, 2014).

As mentioned previously, the adoption of computation in the building industry has been slow, and this applies also to the development of conceptual design tools for structural design. Early computational tools for structural include CAD software for drafting, and FEM programs to perform structural analysis (Rolvink et al., 2011). One recent development in computational tools for the building industry has been the introduction of Building Information Modelling (BIM). BIM is defined by the National Building Information Model Standard Project Committee (United States) as “a shared knowledge resource for information about a facility forming a reliable basis for decisions during its life-cycle; defined as existing from earliest conception to demolition”. The main goal of BIM is to improve collaboration between different actors/stakeholders in the design process by providing them with an integrated digital platform to represent all aspects of the building design (National Institute of Building Sciences, 2014).

Actual conceptual design software in the building industry is still limited. However, there have been a number of developments towards conceptual design tools for structural design. Several of these developments have been described in Rolvink et al. (2011) and are summarized below:

Graphics statics tools

Graphics statics tools allow the user to analyse shapes via the force polygon method. They are useful because they provide a clear, simple overview of structural behaviour. However, they are generally limited to simple problems with two-dimensional and statically-determinate structures, which limits their usability. Examples of graphics statics tools are eQUILIBRIUM (BLOCK Research Group, 2012) and RhinoStatics (Shearer, 2010).

Form-finding tools

Form-finding tools use algorithms to find shapes with little or no bending using constraints defined by the user. These tools are often used to find shapes with minimized bending moment. The limitation of these tools is that they can only be used to design a very small range of structures (such as membranes or shells) and cannot be used to design more general building types.

Design optimisation

In design optimisation, the designer aims to find the best possible design to meet several criteria (for example, stiffness, strength, weight, etc). So far, the use of design optimisation in the building industry has been limited. A possible explanation for this is that the process of optimisation is a convergent process that seeks one optimal solution, whereas conceptual design is generally more divergent and considers a number of different options. Additionally, often it is difficult for the designer to formulate all of the requirements and constraints of the design from the outset, making design optimisation difficult.

Interactive evolutionary exploration tools

Interactive evolutionary exploration tools follow a similar approach to design optimisation with one key difference: the user contributes to the selection process. This type of tool uses an algorithm to develop new design alternatives at every iteration, and then the user chooses the parent alternatives that will go into the next iteration to develop new design alternatives. The problem with these tools is that they can be slow for complex problems, and they often output similar design alternatives instead of providing a wide range of alternatives. An example is IGDT (Intelligent Genetic Design Tool; von Buelow, 2008) and ParaGen (von Buelow, 2011).

Parametric and associative design (PAD) tools	PAD tools allow the user to create and analyse a design using parameters that can be changed to automatically update and regenerate the design. A disadvantage of these tools is that it is difficult to change the design logic at the end of the design process. Furthermore, the model created using PAD tools is the "perfect" building model and doesn't contain any imperfections, which limits the application of structural analysis. Examples of PAD tools are Grasshopper (see below) and Dynamo (Autodesk, 2016).
Dashboard-based design tools	Dashboard-based design tools help the design process by providing a dashboard of tools that the user can use to develop and analyse design alternatives. This type of tool aims to incorporate the actual conceptual design process into the development of the design. The user defines requirements, constraints, boundary conditions and then design alternatives are generated based on "reasoning, thought, schematisation, modelling, analysis and defining scenarios". Then based on these factors, the design alternatives can be evaluated. An example of a dashboard-based design tool is StructuralComponents 2 (Rolvink et al., 2011).

A PAD tool of note is Grasshopper. Grasshopper is a modular visual programming engine, designed by David Rutten that connects to the Rhinoceros CAD application (Robert McNeel & Associates, 2019). Grasshopper is used to create algorithmic models, based on specified design parameters, which can be visualised in the Rhinoceros 3D environment. Grasshopper contains a number of "components" containing predefined logic (for example a Multiplication component defines that inputs 'a' and 'b' result in the product 'c') which can be connected together to create an overall design logic. The benefit of Grasshopper is that it allows users to create complex algorithmic designs without prior knowledge of programming, which makes this design platform accessible to regular designers and engineers. Grasshopper is also extensible, as users can write logic for new components in C#, VB.NET or Python. This extensibility allows for a wide range of uses, as the number of Grasshopper components constantly grow to meet more and more applications (Davidson, 2019). An image of a simple Grasshopper model and its visualisation in Rhinoceros is shown in Figure 3.2.

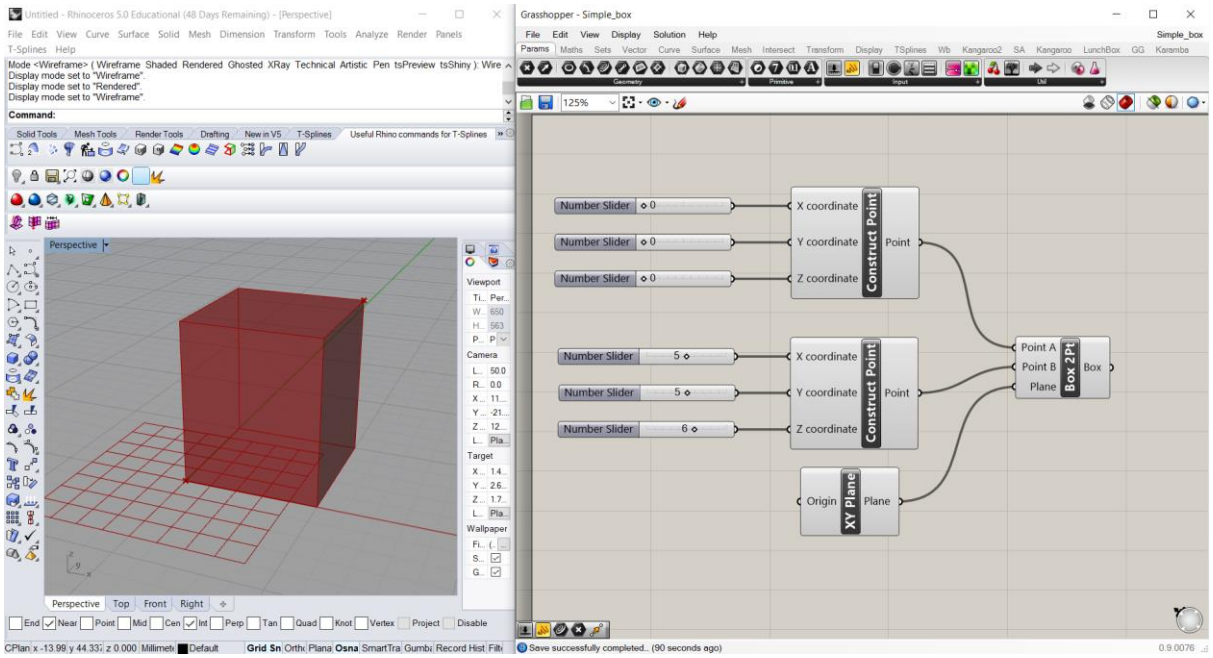


Figure 3.2 Simple Grasshopper model

An additional development related to conceptual design software in the building industry is the project Structural Design Tools by Coenders and Wagemans (2006). The goal of this project was to develop a computational tool to support an "efficient, fast, realistic and practical design process", while remaining as a generalized toolbox that could be used to build other software. Structural Design Tools was developed out of a desire to create a computational tool that would solve the disconnect that currently exists between structural design and computation. The Structural Design Tools concept views design as a complex, cyclic, creative, multi-disciplinary process. To support such a complex design process, Structural Design Tools uses the "openStrategy Form Finding" framework, which is a flexible framework that allows the designer to choose their desired optimisation strategy from several options. Using this framework, a large variety of different problems and algorithms can be implemented, thus supporting the variable nature of design. Structural Design Tools was implemented using PAD software as a basis; custom components were developed using C# (Coenders et al., 2006).

Another development in the field of conceptual design tools for the building industry is the dissertation project *NetworkedDesign* by Coenders (2011). This project focuses on the development of a conceptual infrastructure that supports computational tools for the design of buildings and other structures. NetworkedDesign is an object-oriented infrastructure which is based on the concept of modularity. NetworkedDesign incorporates many concepts that exist in current PAD, but improves on and adds to these concepts to provide a new infrastructure for a broader use in design. Its aim is to improve usability of PAD by replacing the existing low-code approach to parametric design by a no-code approach (Coenders, 2011). Several important concepts of NetworkedDesign are described below:

Change propagation	A key concept in PAD is change propagation; if a parameter is changed, this change propagates through the entire model. NetworkedDesign extends the definition of change propagation by allowing not just changes in parameters, but also changes in the model logic and solving methods. This allows many different problems to be modelled in different ways.
Pluggability	NetworkedDesign is extensible by the user; the user can add plugins or extend its capability through scripting.
Multi-directionality	NetworkedDesign supports not just bi-directionality, but multi-directionality. In bi-directionality, the direction of logic between two objects is reversed. Multi-directionality is a new concept developed in NetworkedDesign, and the idea behind this concept is to allow the user to create multiple paths of logic between objects.
Solving by choice	NetworkedDesign is "solver-indifferent" meaning that it does not contain a pre-prescribed method to resolve changes made to the model. Instead, the user can choose the change propagation process.
Meta-process and meta-knowledge	NetworkedDesign can provide users with insight into their design by providing them with meta-knowledge about the problem. When the user defines a problem, NetworkedDesign can search for pre-defined solutions that solve this problem based on patterns in the problem. Pre-defined solutions are defined in a library, which can be a plug-in or on the internet. The multi-directional aspect of NetworkedDesign allows many different solutions for a single model to be found in this way (Coenders, 2011).

A new development in the field of computational tools for structural design is the platform Packhunt.io developed by White Lioness technologies. Packhunt.io is an online digital twin and parametric design platform intended to make advanced technology available to designers and engineers. Packhunt.io allows users to develop designs/design logic using no-code visual programming tools such as Grasshopper and upload this logic online. The uses of Packhunt.io are flexible; users can create digital twins of structures/objects, modify their designs based on changeable parameters or a customised design logic, automate structural design calculations and cost calculations, and create manufacturable digital models. Packhunt.io facilitates the integration of different parts of the design process, allowing the user to see the impact of design parameters on their final design (White Lioness technologies, 2019).

Outside the building industry, many conceptual design tools already exist, particularly for the mechanical and industrial design industries. Some of these tools are integrated-solution tools, meaning they include not just conceptual design but all aspects of design until manufacturing, and

some tools are standalone conceptual design tools (Pal, 2014). The following paragraphs describe a few conceptual design tools that exist outside of the building industry.

PTC Creo 5.0

PTC Creo 5.0 is a CAD-based software intended for product design. It is an integrated-solution tool that allows the user to create a design from scratch then directly manufacture it. Additionally, Creo 5.0 uses topology optimisation to optimise the design. The user can visualise what their product looks like in real life through augmented reality. Creo 5.0 also provides real-time simulation of the product's behaviour using computational fluid dynamics (PTC, 2018a; BusinessWire, 2016).

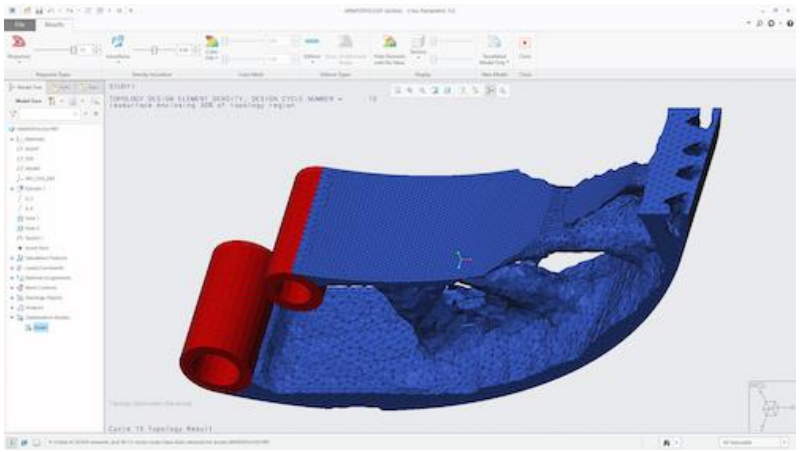


Figure 3.3 Topology optimisation in PTC Creo 5.0 (PTC, 2018b)

Siemens NX

Siemens NX provides an integrated, multidisciplinary platform that integrates electrical, mechanical and control systems. It is meant to help integrate electrical and mechanical industries together, to avoid electromechanical issues in the detailed design. It is an integrated-solution tool that allows the user to create a design and 3D-print it. NX contains generative design software, which generates several alternatives that the designer can then choose from. It also contains validation checks, that validate that the product complies to standards and requirements. NX also provides simulation to check the product's performance, with motion, structural and thermal simulation tools. NX contains a library of pre-designed parts that the designer can reuse in their design (Reyes, 2017; Siemens, 2018).

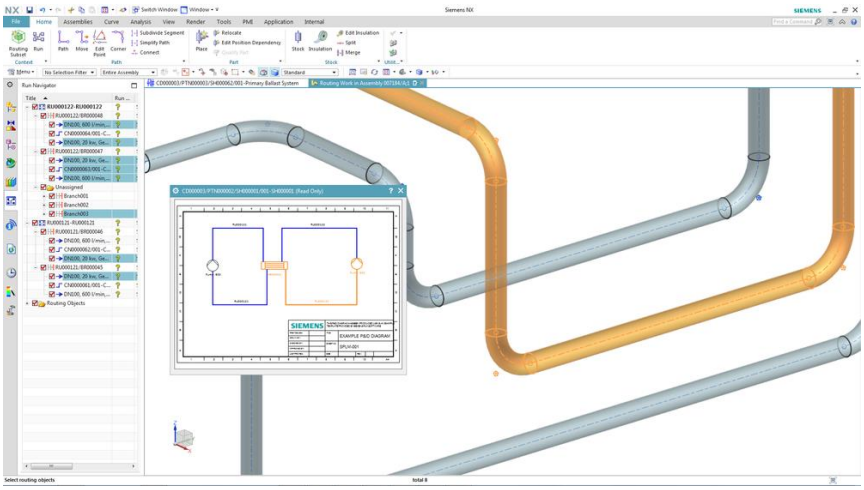


Figure 3.4 Siemens NX (Reyes, 2017)

Solidworks Mechanical Conceptual

Solidworks Mechanical Conceptual (SWMC) was created for the conceptual design of mechanical systems. The user interface is meant to replicate the traditional way of designing on paper, from first developing a 2D sketch then developing this into a 3D concept. In SWMC, the user starts by creating a 2D concept drawing, then adding dimension later to make it into a 3D conceptual design. SWMC provides real-time motion simulation of the design, allowing the user to see the motion paths of the product and determine if there are any overlapping movements. The user can define design requirements and SWMC will alert the user if their design falls outside these requirements. The user can also perform structural FEM analysis of a 3D design. SWMC includes a built-in cloud-based social network, which allows designers to collaborate on the design remotely. There is also an online SWMC community where users can discuss their designs (Dassault Systèmes, 2012; Dassault Systèmes, 2014).

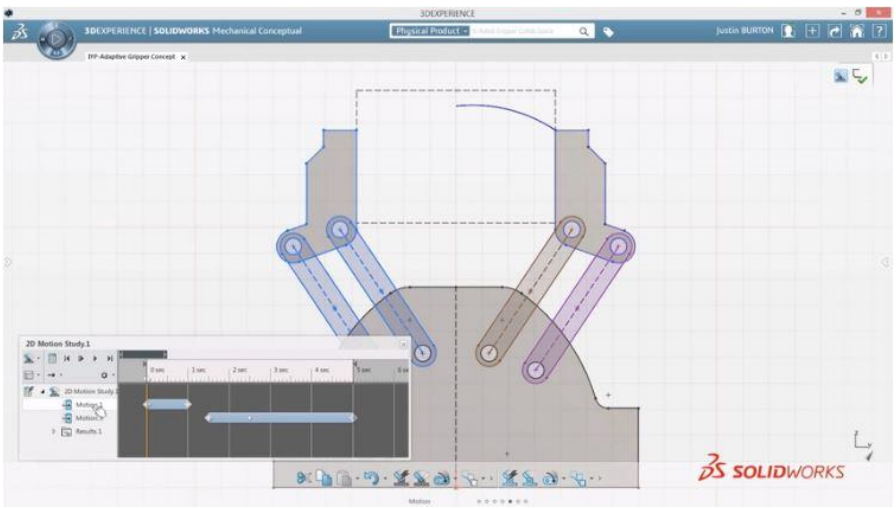


Figure 3.5 SWMC 2D design (Dassault Systèmes, 2014)

Altair Inspire

Altair Inspire is a tool used for the design of mechanical products that focuses on performance and manufacturability. It is an integrated-solution tool that allows the user to go from conceptual

design to manufacturing. Inspire is described as a "simulation-driven design platform" and provides simulations that test kinematic and dynamic motion, manufacturability constraints, and structural behaviour (using FEM analysis). Inspire also provides generative design and topology optimisation. Inspire can be used not only on a personal computer but also on Altair's cloud-based platform. Previously-made designs are stored in a local library and can be reused by designers (Altair, 2019a; Bangal, 2018).

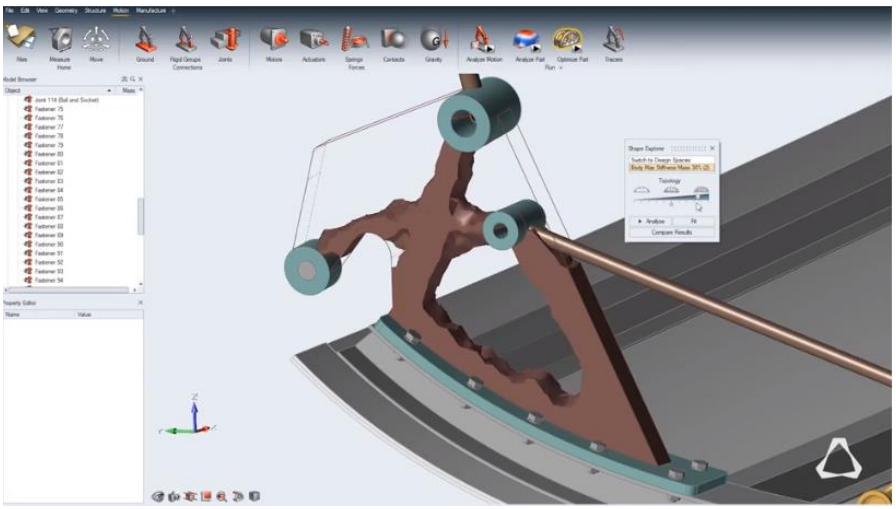


Figure 3.6 Altair Inspire (Altair, 2019b)

Some patterns can be noticed among these different conceptual design tools, as follows:

- **Reuse of knowledge:** Most of the tools take advantage of pre-existing knowledge and use this to enhance/speed up the design process. For example, PTC Creo 5.0 takes information from connected devices to help inform the design process, and most of the tools contain libraries of pre-made designs and parts that the designer can reuse.
- **Real-time simulation/validation:** All of the tools contain real-time simulation of the designs. This is an important feature for conceptual design because it easily allows the designer to quickly determine if their design is feasible or if something needs to be changed, before the design heads to the detailed stage.
- **Parametric design:** All of the tools are parametric and thus allow the user to change individual aspects of the design without affecting the overall design logic.
- **Cloud-based:** SWMC and Altair can both be connected to the internet. This is useful because it provides a means of remote collaboration.

3.3 Framework of a conceptual design tool

Based on the previous exploration of conceptual design, an illustration is created that describes the conceptual design process at a high level, shown in Figure 3.7 below.

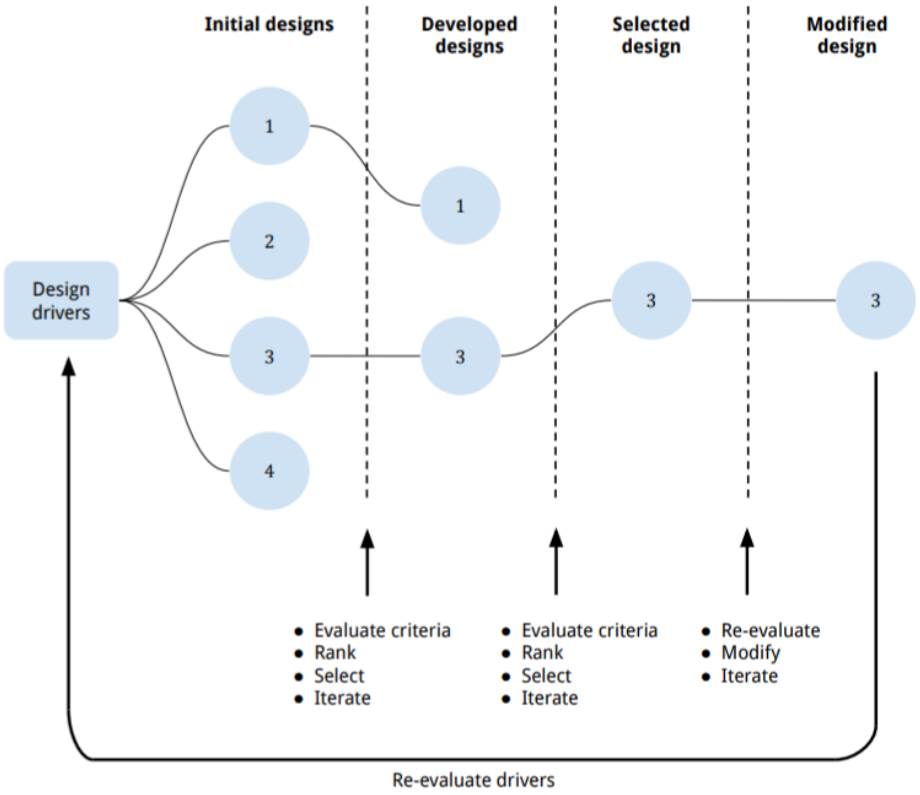


Figure 3.7 Conceptual design process illustration

Figure 3.7 does not represent the whole process of conceptual design, because this process is very complex and frequently non-linear, however it gives a broad overview of the events that occur in the conceptual design process. Certain "design drivers" lead to the creation of several initial designs. These designs are evaluated based on criteria specified by the designer; some designs are developed in more detail than others and re-evaluated based on the design criteria until a certain design is selected. The process is highly iterative and various designs are investigated in parallel. Even after a single design is selected, it may change again if the design drivers and/or criteria change.

Based on this overview of the conceptual design process, a number of qualities are specified as desirable in a conceptual design tool:

1. Quick and easy generation of design alternatives
2. Easy changes to the design
3. Parallel investigation/comparison of alternatives
4. Constant feedback on the design criteria
5. Visualisation of design and analysis results

These four qualities will thus be used as criteria for the development and evaluation of StructuralComponents 6.

4 Structural Mechanics

4.1 Structural analysis in StructuralComponents 5

The goal of StructuralComponents 5 was to create a conceptual design engine for mid-rise concrete buildings composed of cores, shear walls and floors. At the basis of StructuralComponents 5 are three "building blocks" composed of pre-defined configurations of shear walls and/or cores on a floorplan. These configurations are based on the super elements developed by Steenbergen (2007) and are shown in Figure 4.1 below. These building blocks can be stacked to create more complex building types. The building blocks are parametric; properties such as the height of the super element, distance between stability elements, and dimensions of the stability elements can be modified by the user (Hohrath, 2018).

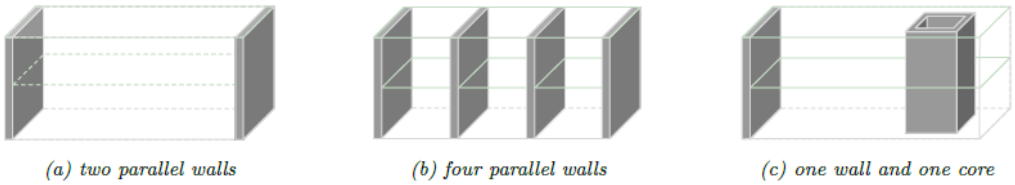


Figure 4.1 "Building blocks" from StructuralComponents 5 (Hohrath, 2018)

After defining a structure, the user can apply a specified wind pressure in the transverse and/or longitudinal direction of their structure, and the tool performs structural analysis. The tool calculates the distribution of the moment, shear force, rotation and displacement along the height of each stability element in the structure. The structural analysis in StructuralComponents 5 is performed using Steenbergen's super element method (2007); please refer to Appendix A for a detailed description of this analysis method (Hohrath, 2018).

A feasibility analysis was conducted to check the accuracy of the structural analysis in Structural Components 5. Each individual building block was analysed, and two additional structures composed of combinations of building blocks were analysed. Validation was performed by comparing the results of StructuralComponents 5 to the results of finite element analysis software MatrixFrame (Matrix Software, 2019). The finite element results were very similar to the results from StructuralComponents 5, which validates Steenbergen's Super Element Method (2007) as an accurate way to determine the building behaviour (Hohrath, 2018).

4.2 Structural analysis in StructuralComponents 6

The goal of StructuralComponents 6 is to extend the flexibility of StructuralComponents 5. Instead of providing the user with predefined building blocks which can be combined to make more complex building types, the aim of StructuralComponents 6 is to allow the user to have more freedom in where they place shear walls and/or cores. In StructuralComponents 6, the stability elements and floors themselves are viewed as the "building blocks" and the user has the freedom to place these elements in different arrangements as they like.

Because StructuralComponents 6 is a conceptual design tool, it must be easy for the user to develop, test and compare different design alternatives. Therefore, speed of analysis and easy generation of alternatives are goals in the development of the tool.

As in StructuralComponents 5, the analysis for StructuralComponents 6 is performed using differential equations rather than finite element analysis. The reason for this is threefold:

- Differential equations provide more insight into the structural behaviour than finite element analysis, because a clear relationship can be identified between the design parameters and analysis results. Finite element software, conversely, can be like a "black box" where the user applies input and receives output, but the relationship between the two is not always clear.
- Differential equations allow for a simpler form of model construction. Finite element models are often time-consuming to assemble, and this makes the comparison of different design alternatives difficult, especially if the design alternatives are very different from each other. This is not very suitable for conceptual design, in which the quick generation and comparison of alternatives should be possible. Using differential equations provides a much simpler method of analysis, which can be easily automated to apply to various designs (given certain limitations) using a programming language such as Python.
- Solving of finite element models is often slow as a result of the large number of individual elements that must be processed. By using differential equations, analysis is simpler and therefore more efficient.

To perform the structural analysis of a customised floorplan topology, an application needs to be written that can analyse multiple combinations and numbers of shear walls and/or cores. Two different approaches are considered for the initial development of the program:

1. Extend and automate Steenbergen's Super Element Method (2007), which was used in StructuralComponents 5. Derive new super elements, determine the pattern between super elements for different topologies and program this pattern.
2. Assume the floors are infinitely rigid. Create a derivation program for this simplified situation.

As demonstrated by the validation performed in StructuralComponents 5, Steenbergen's Super Element Method (2007) produces quite accurate results for the behaviour of the stability elements. Thus Approach 1 will also produce quite accurate results. The disadvantage of this approach, however, is that the derivation of super elements using Steenbergen's method is quite complex and time-consuming, and the pattern between different super elements may be difficult to program.

Assuming the floors are infinitely rigid as in Approach 2 greatly simplifies the problem and is much more easily programmable. However, the results of this method may not be accurate. The range of situations that can be modelled by this program may be restricted to specific situations where the floor stiffness does not excessively influence the reaction of the stability elements.

Based on an initial investigation of the two approaches, it was decided that the approach that is followed is Approach 2, to assume the floors are infinitely rigid. The reason for this decision is based on the complexity of the Super Element Method. Although the Super Element Method

produces quite accurate results for the force, bending moment and deflection of a building, the derivation of results for a given configuration of shear walls, cores and floors is complex. This means it cannot be easily implemented for customised configurations of shear walls and cores, and this is a topic that requires further research. It is therefore decided that in this first implementation of a tool for customised configurations of stability elements, the floors are considered as infinitely rigid.

To test the limitations of this approach, a few different tests are performed with different configurations of shear walls and cores. For each configuration, the building is considered to be prismatic, meaning all the stability elements are the same height and there are no changes in the floorplan along the height of the building. In the initial tests, only unfactored wind load is applied on the building configurations. The stability elements are viewed as flexural beams that are fixed at the foundation and free at the top. The floors restrain the stability elements in-plane, but have no contribution in the out-of-plane direction. The floors and stability elements are assumed to be solid concrete; reinforcing steel is ignored. The three initial tests are described in Sections 4.3, 4.4 and 4.5.

After the three initial tests are analysed, the best test case is chosen and further developed to include foundation stiffness and second-order effect under a vertical load, described in Sections 4.6 and 4.7. The case is then automated for custom configurations of stability elements, described in Section 4.8.

4.3 Test 1: Three shear walls connected by rigid floors

In the first test, a floor supported by three shear walls is modelled with infinitely rigid floors. The system is composed of three equidistant shear walls of equal height, with displacements u_1, u_2 and u_3, respectively. A uniform wind pressure is applied along the floor, in the long direction of the shear walls. This configuration is shown in Figure 4.2 below.

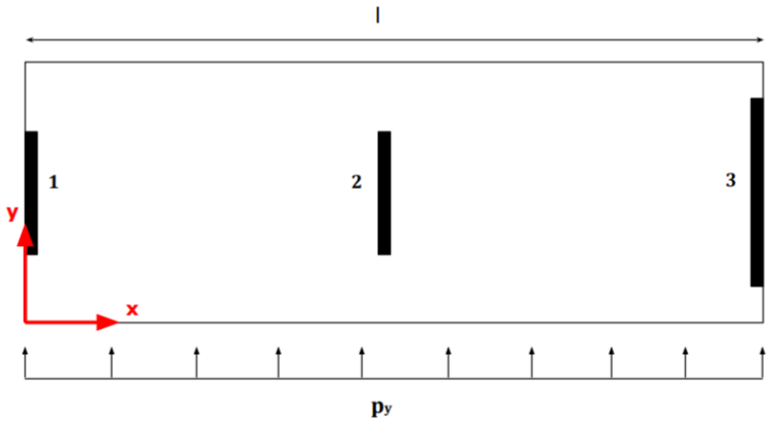


Figure 4.2 Three parallel shear walls connected by an infinitely rigid floor

The deflections of the walls are represented by the following system of equations. These are derived from force and moment equilibrium of the system and the relationships between different shear wall displacements based on rigid rotation of the floor. u_1, u_2, and u_3 represent the deflection of shear walls 1, 2 and 3, respectively, in the y-direction. EI_1, EI_2, EI_3 represent the

bending stiffness of the shear walls in the direction of the wind load. The load p_y is the wind load per 1 metre of the building height. In reality, the wind load would be divided into discrete point loads located at each of the floors, but as an approximation, the wind load is estimated as a uniformly-distributed load along the height of the building. The z-axis points out of the floorplan; the z-value is zero at the base of the building and maximum at the top.

$$EI_1 \frac{d^4u_1}{dz^4} + EI_2 \frac{d^4u_2}{dz^4} + EI_3 \frac{d^4u_3}{dz^4} = p_y \cdot l$$

$$\left(x_1 \cdot EI_1 \frac{d^4u_1}{dz^4}\right) + \left(x_2 \cdot EI_2 \frac{d^4u_2}{dz^4}\right) + \left(x_3 \cdot EI_3 \frac{d^4u_3}{dz^4}\right) = \frac{p_y \cdot l^2}{2}$$

$$u_2 = \frac{u_1 + u_3}{2}$$

The rotation, bending moment and shear force are derived as follows for a given wall "i":

$$\varphi_i = -\frac{du_i}{dz}$$

$$M_i = EI_i \cdot \frac{d\varphi_i}{dz}$$

$$V_i = \frac{dM_i}{dz}$$

The shear walls are viewed as beams fixed at the foundation and given the following boundary conditions: all displacements/rotations are zero at the base of the structure, and all shear forces/bending moments are zero at the top of the structure. The following boundary conditions are used for the system:

$$u_1(0) = 0$$

$$u_2(0) = 0$$

$$\varphi_1(0) = 0$$

$$\varphi_2(0) = 0$$

$$M_1(height) = 0$$

$$M_2(height) = 0$$

$$V_1(height) = 0$$

$$V_2(height) = 0$$

The distribution of displacement, rotation, moment and shear force of each shear wall along its height can then be calculated. A script is written in the mathematical software Maple (Maplesoft, 2019) to derive the displacement/force distributions; this script is provided in Appendix B.

To check the accuracy of this method, the same structure is constructed in the finite element software Oasys GSA (Oasys, 2019) and analysed. The results from Maple and finite element analyses are then compared. The following properties are used for the analysis are shown in Table 4.1.

The height of the building is chosen as 48m to represent a mid-rise building of sixteen storeys with 3-metre-tall floor heights. The floor length is chosen as 40 m to ensure the building is not too slender which would lead to dynamic effects. A wind load of 1.45 kN/m^2 is applied; this is the peak velocity wind pressure on a 50-metre-tall building located in an Urban region of Area I in the Netherlands according to NEN-EN 1991-1-4 (2011). A Young's modulus of 30 GPa is used to represent concrete. A floor thickness of 0.26 metres is chosen; this is based off technical specifications for the maximum thickness of a massive pre-cast concrete floor (Dycore, 2019). The dimensions of the shear walls vary to represent a non-symmetric case. In this test, unfactored wind load is applied on the building.

Table 4.1 Properties used from Test 1 analysis

Property	**Value**
Wind load	1.45 kN/m^2
Young's modulus	30 x 10^6 kN/m^2
Building height	48 m
Floor height	3 m
Floor length	40 m
Floor width	15 m
Floor thickness	0.26 m
Shear wall 1 dimensions	6m x 0.4m
Shear wall 2 dimensions	6m x 0.4m
Shear wall 3 dimensions	12m x 0.4m

In the finite element model constructed in Oasys GSA, the shear walls and floors are modelled as beam elements. The material of all elements is concrete, with the same Young's modulus as specified in Table 4.1 and a Poisson's ratio of 0.2. The concrete is grade C25/30. At the base of each shear wall, all translations and rotations are restrained. No other support conditions are applied on the building. The wind load is applied as a uniformly-distributed load on the floors. A wind load of 1.45 kN/m^2 x 3m = 4.35 kN/m is applied on all floors except the top floor and the ground floor. On the top floor and ground floor, a uniformly-distributed load of 2.175 kN/m (half of 4.35 kN/m) is applied. A static analysis is run on the finite element model. Figure 4.3 shows the finite element model in GSA.

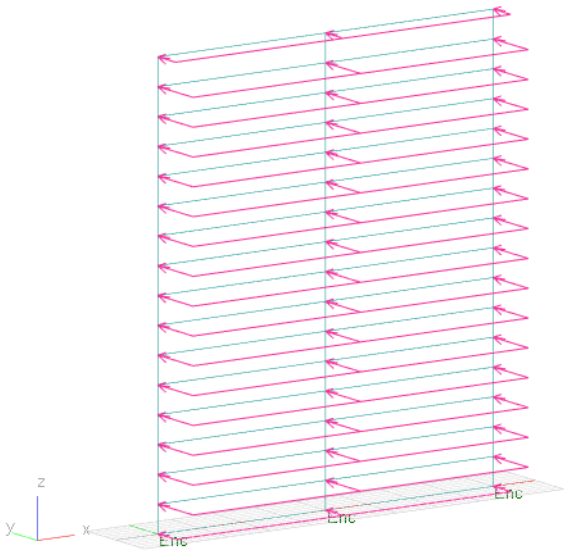


Figure 4.3 Finite element model for Test 1

The comparison of results in Maple and GSA is provided in Table 4.2. All the results are expressed in the y-direction on the global axis specified in Figure 4.3 above.

Table 4.2 Comparison of GSA and Maple results for Test 1

		Maple - rigid	GSA	% difference
Deflection at top (mm)	Wall 1	69.5	67.9	2.4%
	Wall 2	39.1	38.8	0.8%
	Wall 3	8.7	9.7	-10.3%
Shear force at base (kN)	Wall 1	1086.4	1014	7.1%
	Wall 2	611.1	754.1	-19.0%
	Wall 3	1086.4	1016	6.9%
Moment at base (kNm)	Wall 1	-26074.5	-25140	3.7%
	Wall 2	-14666.9	-15000	-2.2%
	Wall 3	-26074.5	-26680	-2.3%

For this method, the results of the Maple script and the finite element analysis are in fairly good agreement with each other. Most results are within a 10% difference. The shear force at the base of Wall 2 is less accurate, at almost a 20% difference. This discrepancy can be explained by the rigid behaviour of the floors. If the floors are considered to be flexible, they deform under the wind load and distribute the wind load to the walls based on the amount of deformation. However, when the floors are considered to be infinitely rigid, the deflection of Wall 2 is the average of the deflection of Walls 1 and 3. This causes more force to be applied on Walls 1 and 3, and less force to be applied on Wall 2. The increase in force on the outer walls is distributed approximately evenly among the two walls. However, the reduction in force on Wall 2 must compensate for the increase in force on both Wall 1 and Wall 3; thus, this reduction in force is twice as large on Wall 2. This leads to a larger relative difference in the shear force on Wall 2.

To verify the analyses, equilibrium conditions are checked for both the Maple and GSA model.

The total force of the system in the y-direction is calculated as follows:

$$F_{y,total} = (1.45\ kPa)(40m)(48m) = 2784\ kN$$

The total bending moment of the system in the y-direction is calculated as follows:

$$M_{y,total} = \frac{(1.45\ kPa)(40m)(48m)^2}{2} = 66816\ kNm$$

The sum of forces and bending moments in the y-direction for both Maple and GSA-analyses is shown in Table 4.3.

Table 4.3 Equilibrium check for Test 1

	Maple – rigid	**GSA**
Sum of forces (kN)	2783.9	2784.1
Sum of bending moments (kNm)	-66815.9	-66820

Table 4.3 proves that both analyses meet equilibrium conditions. Differences against the calculated results are caused by rounding error.

4.4 Test 2: Shear wall and core connected by rigid floors

A second test was created to check the behaviour of a shear wall and core together, for the case of infinitely rigid floors. A model is based on Super Element 3 from Steenbergen (2007). This configuration is shown in Figure 4.4 below:

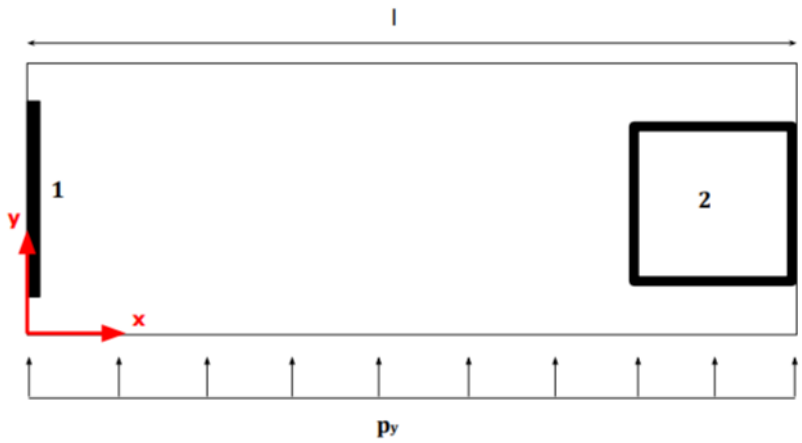


Figure 4.4 Shear wall and core connected by an infinitely rigid floor

The model is represented by the systems of equations as shown below. u_1 and u_2 represent the deflection of the shear wall and core, respectively, in the y-direction, and ϕ represents the rotation of the core. EI_1 and EI_2 are the bending stiffnesses of the shear wall and core, respectively, in the y-direction, and GI_t is the torsional stiffness of the core. p_y is the wind load per metre of building height. Again, the z-axis points out of the page and is zero at the base of the building, and the wind load is approximated as a uniformly-distributed load along the height of the building.

$$EI_1 \frac{d^4 u_1}{dz^4} + EI_2 \frac{d^4 u_2}{dz^4} = p_y \cdot l$$

$$\left(x_1 \cdot EI_1 \frac{d^4 u_2}{dz^4}\right) + \left(x_2 \cdot EI_2 \frac{d^4 u_2}{dz^4}\right) + GI_t \cdot \frac{d^2 \phi}{dz^2} = \frac{p_y \cdot l^2}{2}$$

$$\frac{(u_2 - u_1)}{(x_2 - x_1)} = \phi$$

The rotation, bending moment and shear force for the shear wall and core, individually, are derived as follows:

$$\varphi_i = -\frac{du_i}{dz}$$

$$M_i = EI_i \cdot \frac{d\varphi_i}{dz}$$

$$V_i = \frac{dM_i}{dz}$$

The torsion around the core is derived as follows:

$$T = GI_t \cdot \frac{d\phi}{dz}$$

The shear wall and core are viewed as beams fixed at the foundation and given the following boundary conditions: all displacements/rotations are zero at the base of the structure, and all shear forces/bending moments are zero at the top of the structure. The exact boundary conditions used for the system of equations are shown as follows:

$$u_1(0) = 0$$

$$u_2(0) = 0$$

$$\varphi_1(0) = 0$$

$$\varphi_2(0) = 0$$

$$M_1(height) = 0$$

$$M_2(height) = 0$$

$$V_1(height) = 0$$

$$V_2(height) = 0$$

A Maple script, provided in Appendix B, was written to derive the displacement, rotation, bending moment and shear force of the shear wall and core, and the torsion force in the core based on the equations shown above. To validate Maple results, they are compared against results of a finite element analysis, performed in Oasys GSA. The following properties were used for the analysis.

Table 4.4 Properties used for Test 2

Property	Value
Wind load	1.45 kN/m^2
Young's modulus	30 x 10^6 kN/m^2
Poisson's ratio	0.2
Building height	48 m
Floor height	3 m
Floor length	40 m
Floor width	15 m
Floor thickness	0.26 m
Shear wall dimensions	8m x 0.4m
Core	6m x 6m (0.2m thickness)
Distance between shear wall and core	36 m

In the finite element analysis, the shear wall, core and floors are modelled as beam elements. The material of all elements is concrete, with the same Young's modulus and Poisson's ratio as specified in Table 4.4. The concrete is grade C25/30. At the base of the shear wall and core, all translations and rotations are restrained. No other support conditions are applied on the building. The wind load is applied as a uniformly-distributed load on the floors. A wind load of 4.35 kN/m is applied on all floors except the top floor and the ground floor. On the top floor and ground floor, a uniformly-distributed load of 2.175 kN/m is applied. A static analysis is run on the finite element model. Figure 4.5 shows the model construction in GSA. In this figure, the vertical beam element on the far left is the shear wall, and the vertical beam element on the right is the core.

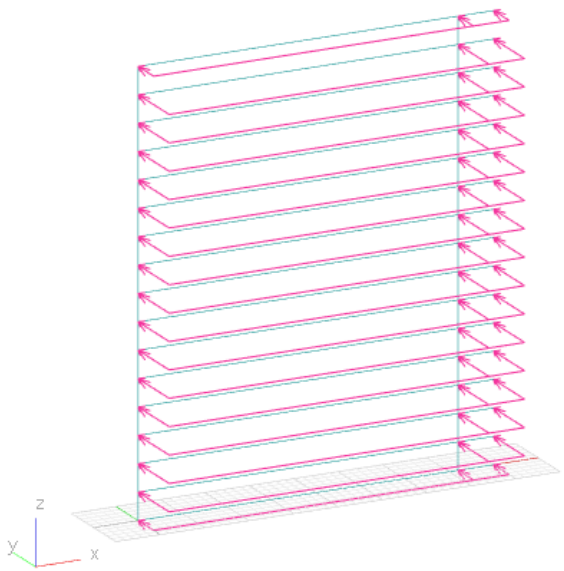


Figure 4.5 Finite element model for Test 2

The comparison of results in GSA and Maple is shown in Table 4.5.

Table 4.5 Comparison of GSA and Maple results for Test 2

		Maple - rigid	GSA	% difference
Deflection at top (mm)	Wall	34	32.6	4.3%
	Core	27	30.1	-10.3%
Shear force at base (kN)	Wall	1310	1230	6.5%
	Core	1474	1554	-5.2%
Moment at base (kNm)	Wall	-30528.9	-28710	6.3%
	Core	-36287.1	-38100	-4.8%
Torsion at base (kNm)	Core	0	-639	-100.0%

Briefly, the results from both Maple and GSA are checked to ensure they meet equilibrium requirements. Table 4.6 shows the equilibrium check for Table 4.5. As calculated in the previous section, the total force in the y-direction is 2784 kN, and the total bending moment in the y-direction at the base of the building is 66816 kNm.

Table 4.6 Equilibrium check for Test 2

	Maple – rigid	GSA
Sum of forces (kN)	2784	2784
Sum of bending moments (kNm)	-66816	-66810

Table 4.6 proves that the equilibrium conditions are met. Small differences are caused by rounding error.

Referring back to Table 4.5, it is seen that the results for deflection, shear force and bending moment from the Maple and finite element analyses are in good agreement, as they are mostly within a 10% difference. However, the torsion around the core as calculated in Maple is inaccurate compared to the finite element result. To look at this problem in more detail, the distribution of torsion along the height of the core is examined in both the Maple analysis and the finite element analysis. Figure 4.6 shows a graph of the torsion around the core taken from the results of the Maple analysis. Figure 4.7 shows a contour drawing of the torsion around the core from the finite element analysis.

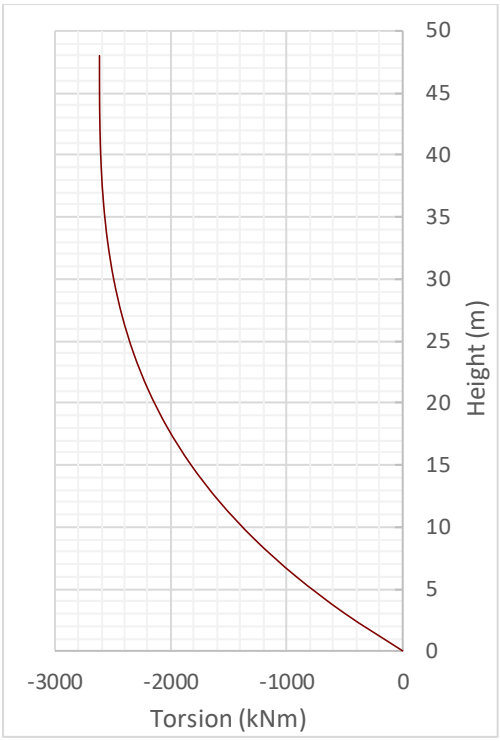


Figure 4.6 Torsion around core from Maple analysis for Test 2

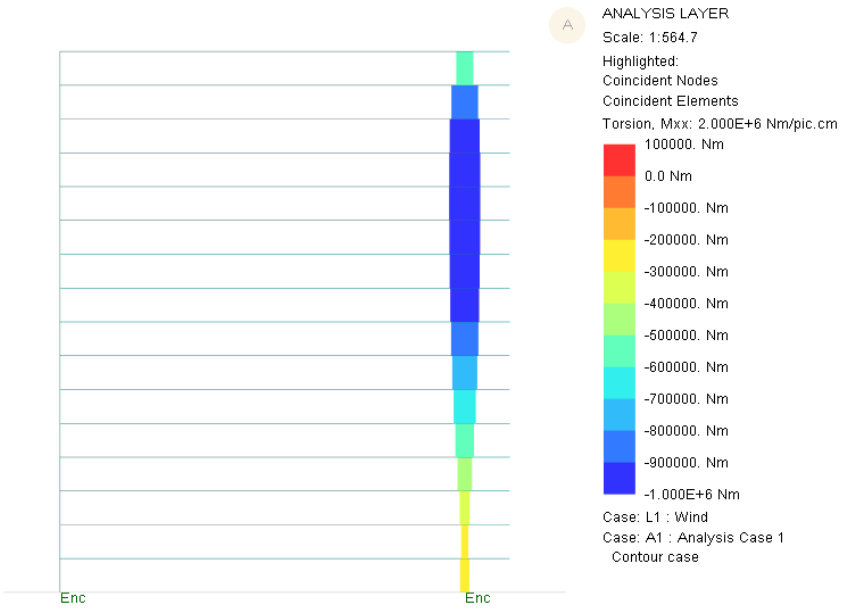


Figure 4.7 Torsion around core from finite element analysis for Test 2

In the Maple results, the torsion is zero at the base and reaches a maximum value of -2616 kNm at the top of the building. In the finite element results, the maximum torsion occurs at about two-thirds the height of the building and has a value of roughly -980 kNm, a bit more than a third of the result predicted in Maple.

The discrepancy between the Maple results and the finite element results indicates that the floor stiffness is an important factor when calculating the torsional force on a stability element in a

system with multiple stability elements. If the floors are infinitely rigid, they create restraint on the rotation of the stability elements. At each floor level, the stability elements and floor must rotate together as a single unit, and so the stability elements themselves cannot rotate independently. However, if the floors are flexible, the stability elements have some freedom to rotate independently. This has significant influence on the torsional force that occurs around the stability element.

The Maple results can be explained by the rigid behaviour at the floors. At the base of the structure, the core cannot rotate at all because it is rigidly connected to the shear wall, which is fixed at its base. Therefore, the torsional force at the base of the core is zero. As the building height increases, the deflection of the shear wall also increases. This deflection is coupled with an increase in the rotation of the core, and therefore an increase in the torsional force. At the top of the structure, the rotation of the core is at a maximum, so the torsional force is also maximum.

When the floors are flexible, the behaviour of the stability elements is more complex. To test the effect of height on the torsion around the core when the floors are flexible, two additional tests are created by varying the building height. In the first test, the building height is changed to 12 metres, and in the second test, the building height is changed to 150 metres. Each of these new models is constructed in Oasys GSA and analysed using a static analysis. Figure 4.8 shows the torsion around the core from GSA for the 12-metre-tall building, and Figure 4.9 shows the torsion around the core from GSA for the 150-metre-tall building.

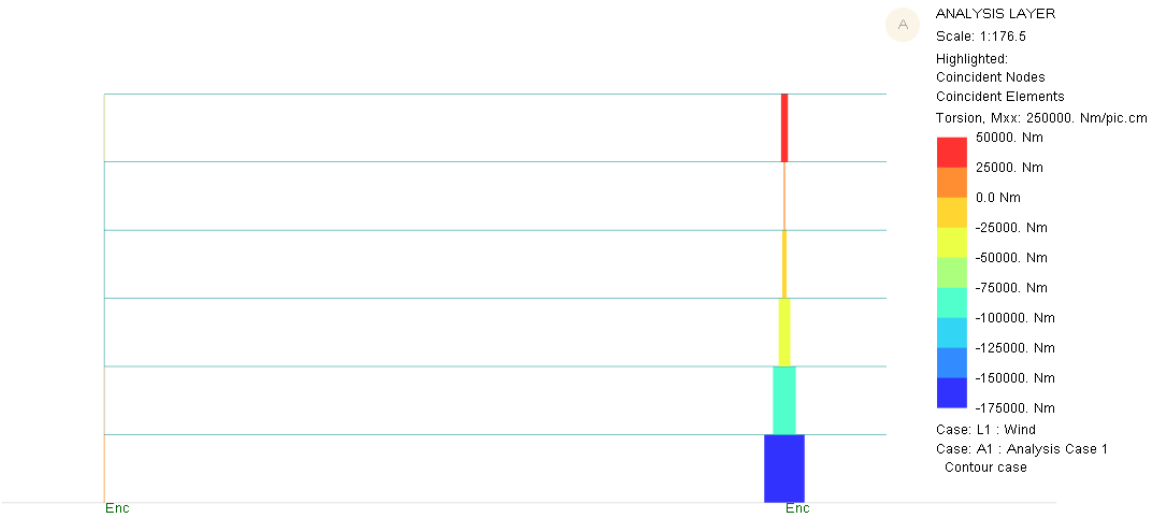


Figure 4.8 Comparison of torsion around the core for 12-m tall building

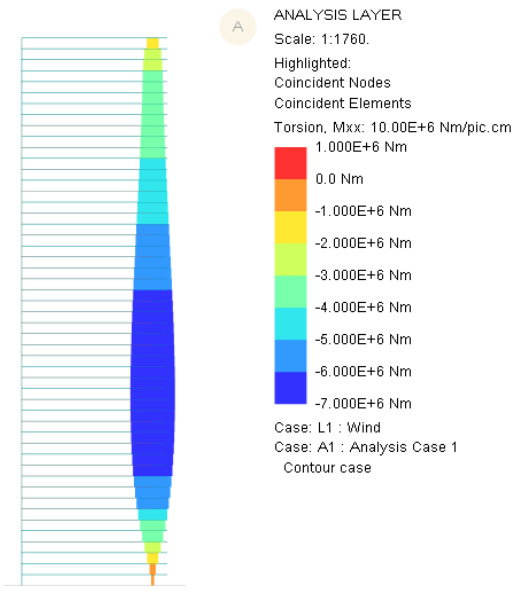


Figure 4.9 Comparison of torsion around the core for 150-m tall building

From these tests, it can be noted that the building height appears to have a significant effect on the location of the maximum torsional force. For a 12-metre-tall building, the maximum torsional force occurs at the base of the structure. However, for a 150-metre-tall building, the maximum torsion force occurs at around a third of the building height. To analyse this behaviour, the deflection of the structure at different heights is observed.

Figure 4.10 shows the deflection of the 12-metre-tall building with a scale factor of 25000 and Figure 4.11 shows the deflection of the 48-metre-tall building from the original test with a scale factor of 250, taken from the finite element analysis in Oasys GSA.

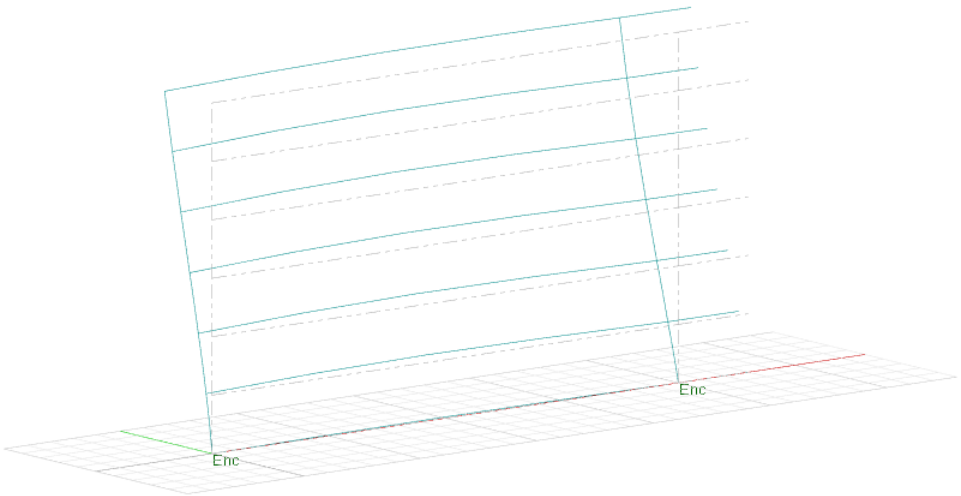


Figure 4.10 Deflection of 12-m tall building with core and shear wall, scale factor 25000

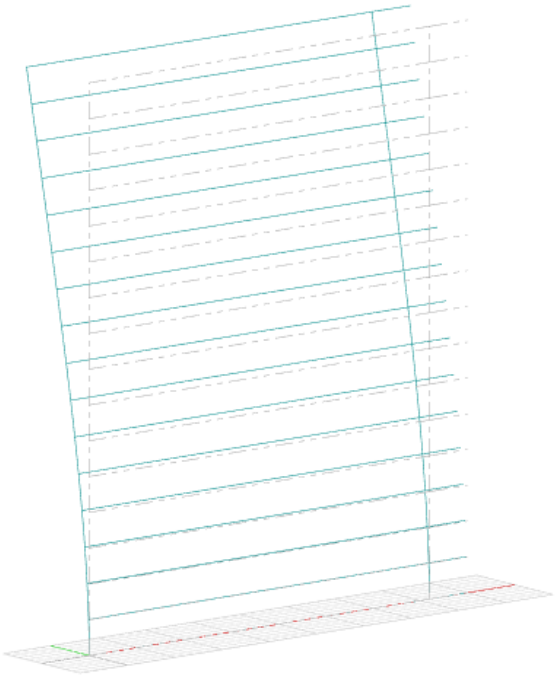


Figure 4.11 Deflection of 48-m tall building with core and shear wall, scale factor 250

For the 12-metre-tall building, the deflection is very small; it is approximately one hundredth of the deflection of the 48-metre-tall building. Additionally, there is significant bending in the floor in proportion to the bending in the shear wall and core. In this case, the building is so short that the shear wall cannot deflect very much, and this restricts the rotation of the core. However, since the floors are flexible, they can compensate for this restriction, and bend to allow the core to rotate slightly. This bending causes torsion around the core. Nearer the top of the building, the deflection of the shear wall is greater, which causes less restriction on the rotation of the core, thus causing less bending of the floor. Therefore, the torsion around the core decreases as the building height increases.

For the 48-metre-tall building, there is less bending in the first floor. In this case, the shear wall has more freedom to deflect, and does not place as large of a restriction on the rotation of the core as for the 12-metre-tall building. Therefore, the torsion around the core at the base is lower.

The location of maximum torsion on the taller buildings is interesting to note. For the 48-metre-tall building, the maximum torsion occurs at around two-thirds the height of the building, and for the 150-metre-tall building, the maximum torsion occurs at around one-third of the height. A likely explanation for this behaviour is the dependence of the rotation of the core on the deflection of the stability elements. Near the base of the building, the rotation of the core is restricted by the deflection of the shear wall. However, at the top of the building, the core is stiffer in deflection than the shear wall, so the deflection of the core itself limits its own rotation. The point of maximum torsion likely indicates the location of the "change-over" of the dependence of the rotation. This is a phenomenon that can only occur when the floors are flexible. When the floors are infinitely rigid, it appears that the rotation of the core is governed only by the deflection of the shear wall, as it increases steadily to a maximum value at the top.

To determine if it is necessary to include torsion in StructuralComponents 6, the torsional stress around the core is checked. Figure 4.12 below shows the torsional stress from the finite element analysis for the 48-metre-tall building.

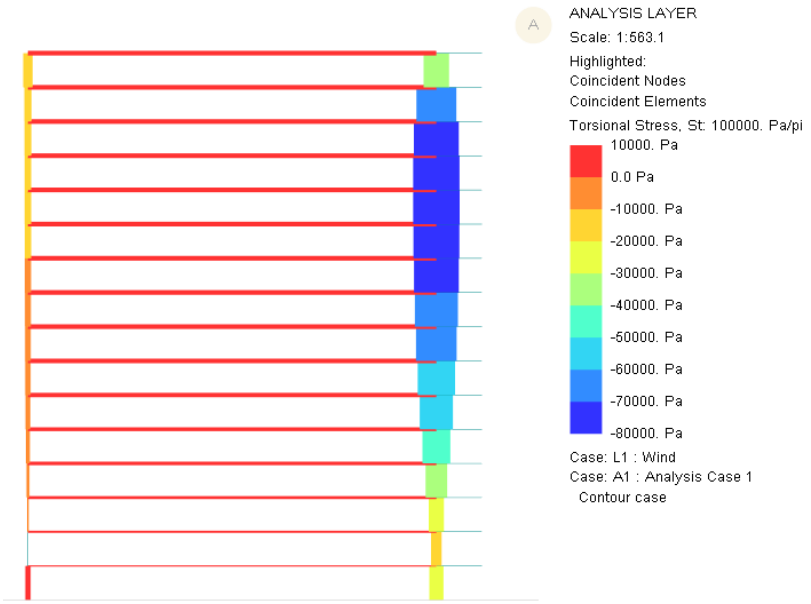


Figure 4.12 Torsional stress for 48-metre-tall building with shear wall and core

The maximum torsional stress on the core is approximately -80 kPa. The characteristic compressive cylinder strength of C12/15 concrete is 12000 kPa, and the characteristic tensile strength is 1600 kPa (NEN-EN 1992-1-1, 2005). These values are both much larger than the torsional stress on the core, indicating that a low-strength concrete is adequate to support this torsional force. As a further comparison, the torsional stress in the system is compared to the bending stress. Figure 4.13 shows the bending stress in the y-direction for the 48-metre-tall building from the finite element analysis.

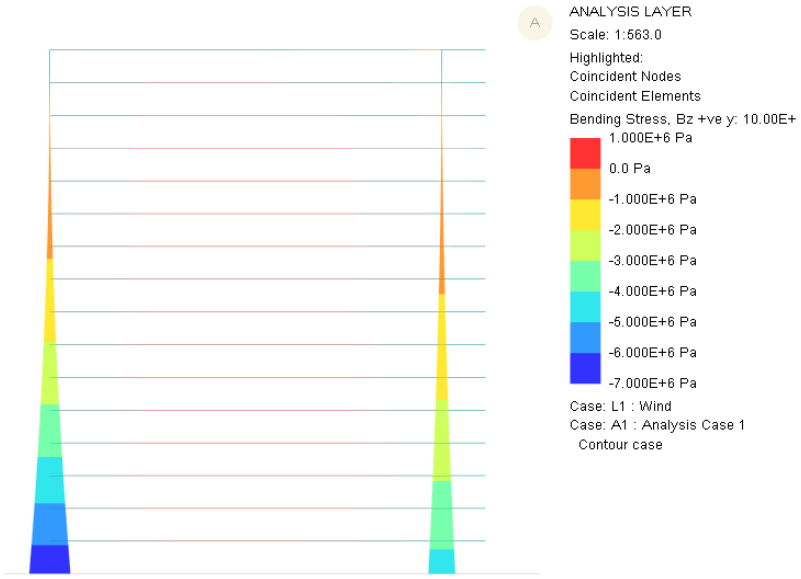


Figure 4.13 Bending stress in y-direction for 48-metre-tall building with shear wall and core

The maximum bending stress at the base of the core is approximately -4400 kPa, 55 times as much as the torsional stress. It can be concluded that the bending stress is governing over the torsional stress when determining the necessary strength of the concrete. On this basis, it is decided that torsion will be omitted in StructuralComponents 6, and only bending, shear force and deflection will be considered in the analysis.

It should be noted that for certain building configurations, the torsional stress is more significant. For example, the torsional stress becomes more significant if there is large asymmetry in the building plan. To show an example, the dimensions of the shear wall and the core from the example above are modified in an exaggerated way to create a highly asymmetric building plan. The dimensions of the shear wall are modified to be 4 metres deep and 0.05 metres wide, and the core is modified to be 8 metres by 8 metres, and 0.5 metres thick. Figure 4.14 and Figure 4.15 show the torsional stress and the bending stress in the y-direction on the structure, respectively.

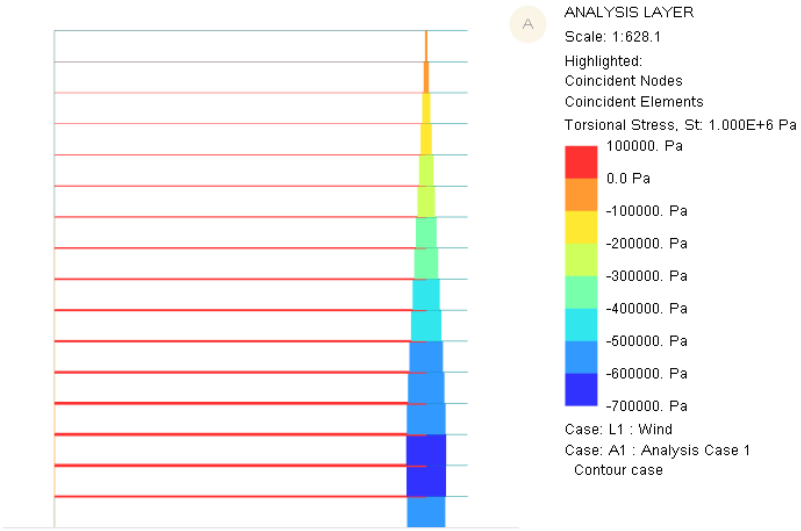


Figure 4.14 Torsional stress around the core for asymmetric building plan

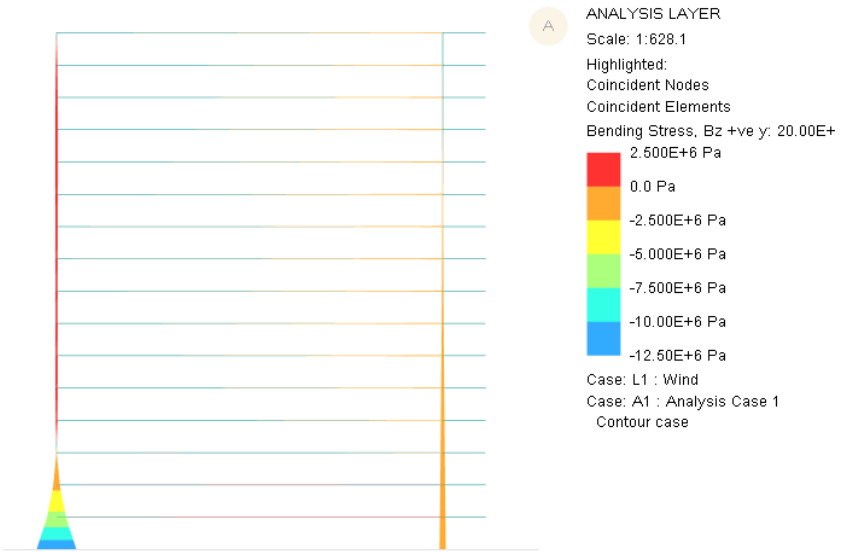


Figure 4.15 Bending stress around the core for asymmetric building plan

The maximum torsional stress on the core is approximately -600 kPa. The maximum bending stress is approximately -1850 kPa. There is a factor of roughly three times difference between these values. The bending stress is still governing, but the torsional stress is now significant in comparison.

In addition, the torsional stress around the core becomes a more significant factor if the building is shorter and broader. From the test on the 12-metre-tall building shown in Figure 4.8, the maximum bending stress at the base of the core is -162 kPa and the maximum torsional stress at the base of the core is -15 kPa. There is still a factor of 10 between the bending stress and the torsional stress, but it can be noted that the torsional stress becomes more significant as the building height decreases.

These analyses show that there is potential for the torsional stress to become significant for certain building sizes and configurations. Based on this finding, it is suggested that for future research into this tool, the effect of the flexibility of the floors on the torsion force should be analysed, and incorporated into the analysis method for the tool.

4.5 Test 3: Shear walls connected by rigid floors on two-dimensional plane

To extend the method to more arrangements of the stability elements, a new model is created with three shear walls on a two-dimensional plane. Wind load is applied in both the x and y directions. An image of the configuration for Test 3 is shown in Figure 4.16.

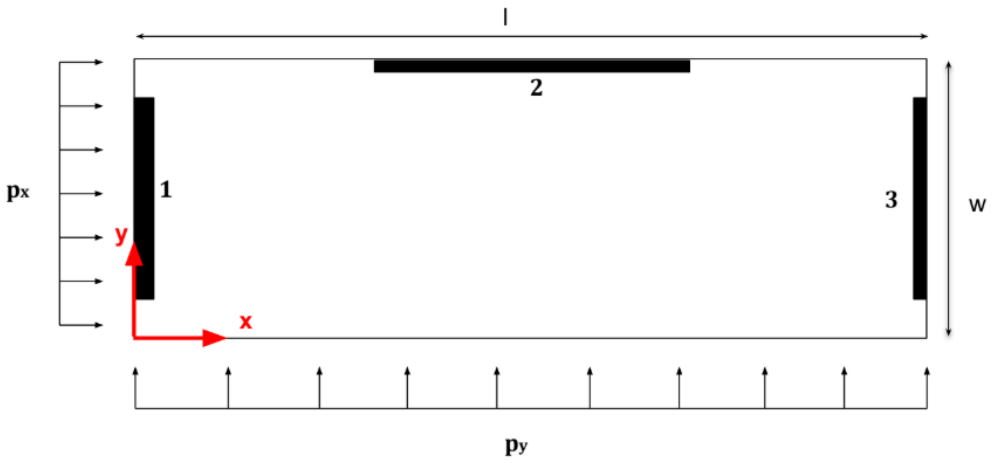


Figure 4.16 Three perpendicular shear walls connected by an infinitely rigid floor

The system can be represented by the following equations below, which are solved based on force and bending moment equilibrium. u_{x1}, u_{x2}, and u_{x3} represent the deflection of walls 1, 2 and 3 respectively in the x-direction, and u_{y1}, u_{y2}, and u_{y3} represent the deflection in the y-direction. These local displacements are stated in terms of global displacements u_x, u_y and ϕ, which are the deflection in x, deflection in y and rotation of the floor at the floor's rotational centre. EI_{x1}, EI_{x2} and EI_{x3} represent bending stiffness of each wall in the x-direction and EI_{y1}, EI_{y2} and EI_{y3} represent bending stiffness of each wall in the y-direction. x_1, x_2 and x_3 represent the distance of the centre of each shear wall away from the origin in the x-direction, and y_1, y_2 and y_3 represent the same in the y-direction. p_x and p_y represent the wind load per 1 m of building height in the x and y-directions, respectively. x_{centre} and y_{centre} represent the x and y coordinates, respectively, of the rotational centre of the floor.

Local system of equations:

$$EI_{x1}\frac{d^4u_{x1}}{dz^4} + EI_{x2}\frac{d^4u_{x2}}{dz^4} + EI_{x3}\frac{d^4u_{x3}}{dz^4} = p_x \cdot w$$

$$EI_{y1}\frac{d^4u_{y1}}{dz^4} + EI_{y2}\frac{d^4u_{y2}}{dz^4} + EI_{y3}\frac{d^4u_{y3}}{dz^4} = p_y \cdot l$$

$$\left(a_1 \cdot EI_{y1}\frac{d^4u_{y1}}{dz^4}\right) - \left(b_1 \cdot EI_{x1}\frac{d^4u_{x1}}{dz^4}\right) + \left(a_2 \cdot EI_{y2}\frac{d^4u_{y2}}{dz^4}\right) - \left(b_2 \cdot EI_{x2}\frac{d^4u_{x2}}{dz^4}\right) + \left(a_3 \cdot EI_{y3}\frac{d^4u_{y3}}{dz^4}\right) - \left(b_3 \cdot EI_{x3}\frac{d^4u_{x3}}{dz^4}\right) = p_y \cdot l \cdot \left(x_{centre} - \frac{l}{2}\right) - p_x \cdot l \cdot \left(y_{centre} - \frac{w}{2}\right)$$

Relationships between local and global displacements:

$$u_{x1} = u_x - \phi \cdot b_1$$

$$u_{x2} = u_x - \phi \cdot b_2$$

$$u_{x3} = u_x - \phi \cdot b_3$$

$$u_{y1} = u_y + \phi \cdot a_1$$

$$u_{y2} = u_y + \phi \cdot a_2$$

$$u_{y3} = u_y + \phi \cdot a_3$$

where

$$a_1 = x_{centre} - x_1$$

$$a_2 = x_{centre} - x_2$$

$$a_3 = x_{centre} - x_3$$

$$b_1 = y_{centre} - y_1$$

$$b_2 = y_{centre} - y_2$$

$$b_3 = y_{centre} - y_3$$

The rotation, bending moment and shear force are derived as follows for a given wall "i" for either the x or y-direction:

$$\varphi_{(x,y)i} = -\frac{du_{(x,y)i}}{dz}$$

$$M_{(x,y)i} = EI_i \cdot \frac{d\varphi_{(x,y)i}}{dz}$$

$$V_{(x,y)i} = \frac{dM_{(x,y)i}}{dz}$$

The shear walls are viewed as beams fixed at the foundation and given the following boundary conditions: all displacements/rotations are zero at the base of the structure, and all shear forces/bending moments are zero at the top of the structure. The boundary conditions used for the system are shown as follows:

$$u_{x1}(0) = 0$$

$$u_{y1}(0) = 0$$

$$u_{y2}(0) = 0$$

$$\varphi_{x1}(0) = 0$$

$$\varphi_{y1}(0) = 0$$

$$\varphi_{y2}(0) = 0$$

$$M_{x1}(height) = 0$$

$$M_{y1}(height) = 0$$

$$M_{y2}(height) = 0$$

$$V_{x1}(height) = 0$$

$$V_{y1}(height) = 0$$

$$V_{y2}(height) = 0$$

A Maple script, provided in Appendix B, was written to derive the displacement, rotation, bending moment and shear force of the shear walls in both the y and x directions based on the equations described above. The Maple analysis was performed twice, once with the wind load applied in the x-direction, and once with the wind load applied in the y-direction. To validate Maple results, they were compared against results of a finite element analysis using Oasys GSA. The following properties were used for analysis.

Table 4.7 Properties used for Test 3

Property	**Value**
Wind load (x or y)	1.45 kN/m^2
Young's modulus	30 x 10^6 kN/m^2
Building height	48 m
Floor height	3 m
Floor length	40 m
Floor width	15 m
Floor thickness	0.26 m
Shear wall 1 dimensions	6m x 0.4m
Shear wall 2 dimensions	0.2m x 8m
Shear wall 3 dimensions	6m x 0.2m

In the finite element analysis, the shear walls are modelled as beam elements. The floors are constructed from two-dimensional Quad-4 elements. The material of all elements is concrete,

with the same Young's modulus and Poisson's ratio as specified in Table 4.7. The concrete is grade C25/30. At the base of the shear walls, all translations and rotations are restrained. No other support conditions are applied on the building. The wind load is applied as a uniformly-distributed load on the floors. A wind load of 4.35 kN/m is applied on all floors except the top floor and the ground floor. On the top floor and ground floor, a uniformly-distributed load of 2.175 kN/m is applied. Two analyses are performed; one with wind load applied in the x-direction, and one with wind load applied in the y-direction. A static analysis is run on the finite element model for both cases. The finite element models for these two analyses are shown in Figure 4.17 and Figure 4.18.

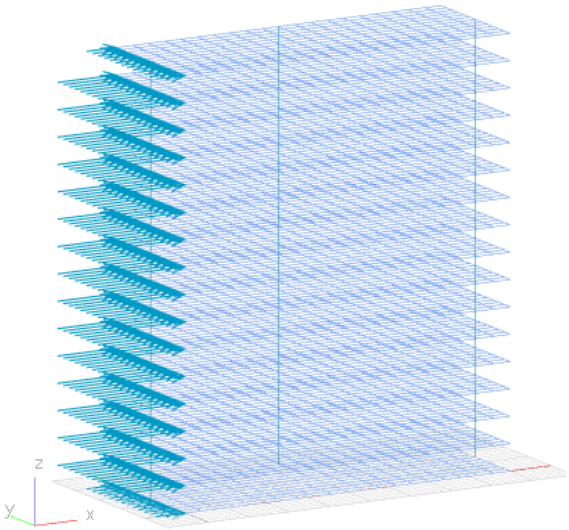


Figure 4.17 GSA model for Test 3, wind load in x-direction

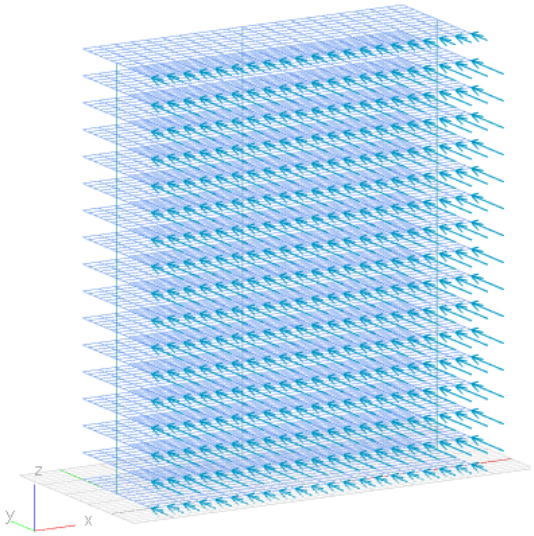


Figure 4.18 GSA model for Test 3, wind load in y-direction

A comparison of the results from Maple and the GSA analysis for both tests are summarised in the tables below.

Table 4.8 Comparison of Maple and GSA for Test 3, px = 1.45 kPa and py = 0 kPa

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	63.1	53.7	17.5%
		Wall 2	56.1	47.7	17.6%
		Wall 3	63.1	53.6	17.7%
	y	Wall 1	-12.5	-11	13.6%
		Wall 2	6.2	5	24.0%
		Wall 3	24.9	20.9	19.1%
Shear force at base (kN)	x	Wall 1	4.4	56.1	-92.2%
		Wall 2	1039.1	975.2	6.6%
		Wall 3	0.5	12.8	-96.1%
	y	Wall 1	-194.9	-183	6.5%
		Wall 2	0.1	0.7	-85.7%
		Wall 3	194.8	182.3	6.9%
Moment at base (kNm)	x	Wall 1	-105.2	-166.2	-36.7%
		Wall 2	-24937.6	-22030	13.2%
		Wall 3	-13.2	-23.7	-44.3%
	y	Wall 1	4676.7	4250	10.0%
		Wall 2	-1.7	-2.4	-29.2%
		Wall 3	-4674.9	-4128	13.3%

Table 4.9 Comparison of Maple and GSA for Test 3, px = 0 kPa and py = 1.45 kPa

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	16.6	13.3	24.8%
		Wall 2	-0.07	-1	-93.0%
		Wall 3	16.6	13.3	24.8%
	y	Wall 1	89.1	88.8	0.3%
		Wall 2	133.6	128.1	4.3%
		Wall 3	178	167.4	6.3%
Shear force at base (kN)	x	Wall 1	1.1	6	-81.7%
		Wall 2	-1.3	-7	-81.4%
		Wall 3	0.1	0.9	-88.9%
	y	Wall 1	1391.5	1365	1.9%
		Wall 2	1.5	58.3	-97.4%
		Wall 3	1391	1361	2.2%
Moment at base (kNm)	x	Wall 1	-27.7	-36.6	-24.3%
		Wall 2	31.2	436.6	-92.9%
		Wall 3	-3.5	-2.3	52.2%
	y	Wall 1	-33395.3	-32990	1.2%
		Wall 2	-37.1	-60.9	-39.1%
		Wall 3	-33383.6	-31750	5.2%

Before the results are analysed, an equilibrium check is performed on both analyses.

To check force equilibrium for the Maple and GSA analyses, the sum of the forces on each shear wall for each analysis should be equal to the total external force on the building. To check the moment equilibrium in Maple, the sum of the bending moments should be equal to the external bending moment on the building.

In the GSA analysis, the floors exhibit out-of-plane behaviour, which was not accounted for in the Maple analysis. This means that the floors exert a normal force on the shear walls. To check the bending moment equilibrium for the GSA analysis, the sum of the bending moments on all the shear walls plus the additional bending moment caused by this normal force must be equal to the total external bending moment.

In the first analysis, wind load is applied in the x-direction. The total external force in the x-direction can be calculated as follows:

$$F_{x,total} = (1.45\ kPa)(15m)(48m) = 1044\ kN$$

The total external bending moment in the x-direction at the base of the building can be calculated as follows:

$$M_{x,total} = \frac{(1.45\ kPa)(15m)(48m)^2}{2} = 25056\ kNm$$

The total external force and bending moment in the y-direction are equal to zero.

Table 4.10 shows the normal force on each wall taken from the GSA analysis, when wind is applied in the x-direction. A positive value indicates tension, and a negative value indicates compression.

Table 4.10 Normal force on shear walls from GSA for Test 3, wind applied in x-direction

Wall	Normal force (kN)
Wall 1	79.0
Wall 2	-15.9
Wall 3	-63.1

The total bending moment in the x-direction from the GSA analysis can be calculated by summing moments about the centre of the building:

$$\Sigma M_x = -166.2\ kNm - 22030\ kNm - 23.7\ kNm - 79.0\ kN(20m) - 15.9\ kN(0m) - 63.1\ kN(20m)$$

$$\Sigma M_x = -25061.9$$

Table 4.11 shows all equilibrium checks for Test 3, when wind is applied in the x-direction.

Table 4.11 Equilibrium check for Test 3, wind applied in x-direction

	Maple – rigid	**GSA**
Sum of forces in x (kN)	1044	1044.1
Sum of forces in y (kN)	0	0
Sum of bending moments in x (kNm)	-25056	-25061.9
Sum of bending moments in y (kNm)	0.1	0.4

Table 4.11 shows that both the Maple and GSA analyses for wind in the x-direction meet equilibrium conditions. Small differences in value are caused by rounding error.

Another equilibrium check is performed for the second analysis, where wind is applied in the y-direction. As calculated previously, the total force in the y-direction in this case is 2784 kN, and the total bending moment in the y-direction at the base of the building is 66816 kNm. The total force and bending moment in the x-direction are both 0.

Table 4.12 shows the normal force at the base of each shear wall when wind is applied in the y-direction.

Table 4.12 Normal force on shear walls from GSA for test 3, wind applied in y-direction

Wall	**Normal force (kN)**
Wall 1	144.3
Wall 2	-268.7
Wall 3	124.4

The total bending moment in the y-direction from the GSA analysis can be calculated by summing moments about the centre of the building:

$$\Sigma M_y = -32990\ kNm - 60.9\ kNm - 31750\ kNm + 144.3\ kN(0m) - 268.7\ kN(7.5m) + 124.4\ kN(0m)$$

$$\Sigma M_y = -66816.2\ kNm$$

Table 4.13 shows the rest of the equilibrium checks for Test 3, when wind is applied in the y-direction.

Table 4.13 Equilibrium check for Test 3, wind applied in y-direction

	Maple – rigid	**GSA**
Sum of forces in x (kN)	-0.1	-0.1
Sum of forces in y (kN)	2784	2784.3
Sum of bending moments in x (kNm)	0	-0.3
Sum of bending moments in y (kNm)	-66816	-66816.2

Table 4.13 shows that equilibrium is retained for both analyses. Again, small differences in value are caused by rounding error.

Now that the equilibrium is checked, the results are analysed.

As can be observed in Table 4.8 and Table 4.9, the results for this test are considerably less accurate than the results for the previous tests. Almost all results are above a 10% difference. It can be noted however that for some values, a large difference in percentage does not cause a great difference in value, since the values predicted in the finite element analysis are already small. For example, in Table 4.9, the percentage difference on the deflection of Wall 2 in the x-direction is 93%, however the absolute difference in deflection is 0.93 mm, which is a very small difference.

Another observation to note is that the Maple results tend to overestimate the force on stiffer stability elements and underestimate the force on weaker stability elements in a given direction. Table 4.14 shows an example of this for the shear force in the y-direction in the case where the wind load is applied in the y-direction (these values are taken from Table 4.9).

Table 4.14 Shear force in y-direction for Test 3, wind load applied in y

Wall	Maple – rigid	GSA	% difference
Wall 1	1391.5	1365	1.9%
Wall 2	1.5	58.3	-97.4%
Wall 3	1391	1361	2.2%

Walls 1 and 3 are strong in the y-direction, and the shear force on these walls is overestimated by the Maple calculation. Conversely, Wall 2 is strong in the x-direction and weak in the y-direction. The shear force in the y-direction is underestimated by the Maple calculation.

There is one notable exception to this trend: the bending moment in the x-direction in Table 4.9 (where the wind load is applied in the y-direction only). The results for the bending moment in the x-direction when wind is applied in y are shown in Table 4.15 (values taken from Table 4.9).

Table 4.15 Bending moment in x-direction for Test 3, wind load applied in y

Wall	Maple – rigid	GSA	% difference
Wall 1	-27.7	-36.6	-24.3%
Wall 2	31.2	436.6	-92.9%
Wall 3	-3.5	-2.3	52.2%

In this case, the Maple results greatly underestimate the force on Wall 2, which is unexpected because Wall 2 is much stiffer in the x-direction than Walls 1 and 3 and should thus carry a much larger bending moment.

This discrepancy can be explained by the equilibrium of the system. In the GSA analysis, the floors exert a significant normal force on the shear walls. To maintain equilibrium with these normal forces, the bending moment at the base of Wall 2 must have a large positive value. In the Maple analysis, there is no normal force exerted on the shear walls, so to maintain equilibrium in this case, the bending moment at the base of Wall 2 is much smaller.

It is important to note that the bending moment in the x-direction at the base of Wall 2 in Table 4.9 (where wind is applied in the y-direction) is not governing for the design of Wall 2. The bending moment is significantly larger in Table 4.8 (where wind is applied in the x-direction), therefore this result is governing in the design of Wall 2. It is therefore decided that the result from Table 4.9 can be disregarded for purposes of building design. However, it is clear that the

out-of-plane effects of the floors have a significant effect on the bending moments perpendicular to the direction of the force application, and this would be interesting topic for further research.

To test the effect of floor thickness on the accuracy of the results, another test is performed by reducing the floor thickness to 0.1 metres but keeping all other properties the same. The results for the two tests (with wind in the x and y-directions respectively) are shown below.

Table 4.16 Comparison of Maple and GSA for Test 3, px = 1.45 kPa and py = 0 kPa with 0.1m floor thickness

			Maple - rigid	**GSA**	**% difference**
Deflection at top (mm)	x	Wall 1	63.1	63.9	-1.3%
		Wall 2	56.1	56.9	-1.4%
		Wall 3	63.1	63.9	-1.3%
	y	Wall 1	-12.5	-12.4	0.8%
		Wall 2	6.2	6.1	1.6%
		Wall 3	24.9	24.7	0.8%
Shear force at base (kN)	x	Wall 1	4.4	52.5	-91.6%
		Wall 2	1039.1	981.5	5.9%
		Wall 3	0.5	10	-95.0%
	y	Wall 1	-194.9	-184	5.9%
		Wall 2	0.1	0.5	-80.0%
		Wall 3	194.8	183.5	6.2%
Moment at base (kNm)	x	Wall 1	-105.2	-176.4	-40.4%
		Wall 2	-24937.6	-24650	1.2%
		Wall 3	-13.2	-22.5	-41.3%
	y	Wall 1	4676.7	4590	1.9%
		Wall 2	-1.7	-2.4	-29.2%
		Wall 3	-4674.9	-4582	2.0%

Table 4.17 Comparison of Maple and GSA for Test 3, px = 0 kPa and py = 1.45 kPa with 0.1m floor thickness

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	16.6	16.4	1.2%
		Wall 2	-0.07	-0.2	-65.0%
		Wall 3	16.6	16.4	1.2%
	y	Wall 1	89.1	90.8	-1.9%
		Wall 2	133.6	135.1	-1.1%
		Wall 3	178	179.3	-0.7%
Shear force at base (kN)	x	Wall 1	1.1	1.5	-26.7%
		Wall 2	-1.3	-4.3	-69.8%
		Wall 3	0.1	2.8	-96.4%
	y	Wall 1	1391.5	1365	1.9%
		Wall 2	1.5	58.3	-97.4%
		Wall 3	1391	1361	2.2%
Moment at base (kNm)	x	Wall 1	-27.7	-34.6	-19.9%
		Wall 2	31.2	66.7	-53.2%
		Wall 3	-3.5	-2.9	20.7%
	y	Wall 1	-33395.3	-33450	-0.2%
		Wall 2	-37.1	-66.1	-43.9%
		Wall 3	-33383.6	-33160	0.7%

When the floor thickness is reduced, the deflection results become considerably more accurate. Additionally, the results for shear force and bending moment become more accurate along the strong axes of the shear walls. However, many results along the weak axes of the walls become less accurate. For example, the shear force in the x-direction when the wind load is applied in the x-direction shows this behaviour (see Table 4.16). When the floor thickness is reduced, the calculated shear force on Wall 2, which is stiffest in the x-direction, becomes more accurate. However, the calculated shear force on Walls 1 and 3, which are weak in the x-direction, become considerably less accurate. However, the forces in the weak direction of the walls are not governing in design, so this inaccuracy is not considered to be important.

To analyse the accuracy of the results in terms of their applicability to building design, only the governing (maximum) values shall be considered. The governing results for design for shear force and bending moment all share the following characteristics:

1. The result occurs along the strong axis of the stability element
2. The result is in the same direction as the wind application

Additionally, the governing results for deflection occur in the direction of the wind application. Table 4.18 and Table 4.19 show the governing values for design, taken from Table 4.16 and Table 4.17.

Table 4.18 Comparison of Maple and GSA for Test 3, px = 1.45 kPa and py = 0 kPa with 0.1m floor thickness, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	63.1	63.9	-1.3%
		Wall 2	56.1	56.9	-1.4%
		Wall 3	63.1	63.9	-1.3%
Shear force at base (kN)	x	Wall 2	1039.1	981.5	5.9%
Moment at base (kNm)	x	Wall 2	-24937.6	-24650	1.2%

Table 4.19 Comparison of Maple and GSA for Test 3, px = 0 kPa and py = 1.45 kPa with 0.1m floor thickness, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	y	Wall 1	89.1	90.8	-1.9%
		Wall 2	133.6	135.1	-1.1%
		Wall 3	178	179.3	-0.7%
Shear force at base (kN)	y	Wall 1	1391.5	1365	1.9%
		Wall 3	1391	1361	2.2%
Moment at base (kNm)	y	Wall 1	-33395.3	-33450	-0.2%
		Wall 3	-33383.6	-33160	0.7%

As seen in Table 4.18 and Table 4.19, the governing values are quite accurate. The maximum percentage difference between the Maple results and the finite element results is 5.9%.

If the maximum allowable percentage difference between the Maple and finite element results is limited to 10% and only the governing values are judged, then the maximum allowable floor thickness can be determined for Test 3. Based on this criterium, the maximum allowable floor thickness to the nearest centimetre is determined to be 0.21 metres, for the particular floor size and configuration of shear walls in Test 3 and properties as presented in Table 4.7. Table 4.20 and Table 4.21 show the comparison between Maple and finite element results when floor is 0.21 metres thick, for the governing results only.

Table 4.20 Comparison of Maple and GSA for Test 3, px = 1.45 kPa and py = 0 kPa with 0.21m floor thickness, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	63.1	58.1	8.6%
		Wall 2	56.1	51.7	8.5%
		Wall 3	63.1	58.1	8.6%
Shear force at base (kN)	x	Wall 2	1039.1	978.3	6.2%
Moment at base (kNm)	x	Wall 2	-24937.6	-23180	7.6%

Table 4.21 Comparison of Maple and GSA for Test 3, px = 0 kPa and py = 1.45 kPa with 0.21m floor thickness, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	y	Wall 1	89.1	89.7	-0.7%
		Wall 2	133.6	131.4	1.7%
		Wall 3	178	173	2.9%
Shear force at base (kN)	y	Wall 1	1391.5	1365	1.9%
		Wall 3	1391	1362	2.1%
Moment at base (kNm)	y	Wall 1	-33395.3	-33210	0.6%
		Wall 3	-33383.6	-32420	3.0%

This floor thickness is not universal for all floor sizes and stability element dimensions. When the floor depth increases from 15 metres to 30 metres, this floor thickness is no longer acceptable. To maintain a percentage difference of 10%, the floor thickness must be reduced to 0.19 metres. Table 4.22 and Table 4.23 show the governing results for deflection, shear force and bending moment when the floor depth is increased to 30 metres.

Table 4.22 Comparison of Maple and GSA for Test 3, px = 1.45 kPa and py = 0 kPa with 0.19m floor thickness and 30 metre depth, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	168.1	154.8	8.6%
		Wall 2	112	104.3	7.4%
		Wall 3	168.1	154.8	8.6%
Shear force at base (kN)	x	Wall 2	2074.9	1947	6.6%
Moment at base (kNm)	x	Wall 2	-49796.9	-46550	7.0%

Table 4.23 Comparison of Maple and GSA for Test 3, px = 0 kPa and py = 1.45 kPa with 0.19m floor thickness and 30 metre depth, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	y	Wall 1	89.1	88.3	0.9%
		Wall 2	133.6	128.4	4.1%
		Wall 3	178	168.6	5.6%
Shear force at base (kN)	y	Wall 1	1392.2	1376	1.2%
		Wall 3	1390.3	1366	1.8%
Moment at base (kNm)	y	Wall 1	-33412.8	-32870	1.7%
		Wall 3	-33366.1	-31890	4.6%

From these results, it can be concluded that as the floors become thinner, the Maple results generally become more accurate. Additionally, as the floor becomes deeper, it must also become thinner to retain the accuracy of the results.

This is caused by the out-of-plane behaviour of the floors. In the Maple calculation, out-of-plane behaviour of the floors is ignored. However, in GSA, out-of-plane effects of the floor are included. As the floor becomes thinner, these out-of-plane effects become less and less significant, which matches better with the Maple calculation. Additionally, as the floors become deeper, the out-of-plane behaviour becomes more significant. This is simply because the amount of wind load applied on the floors in the x-direction increases as the building grows in this direction.

The system of equations as presented in Test 3 is chosen for further development of the tool. The reason for this decision is that Test 3 is better than Test 1 and Test 2 on the basis of flexibility and accuracy, respectively. Test 3 is better than Test 1 in terms of flexibility, because wind can be applied in both the x and y-directions, instead of just the y-direction. Test 3 is better than Test 2 in terms of accuracy, because Test 2 includes torsion, which as demonstrated cannot be accurately calculated when the floors are assumed to be infinitely rigid. Additionally, the torsion stresses are generally low, so it is decided that torsion can be ignored in the tool. Furthermore, the governing results from Test 3 are within a reasonable range of accuracy when compared to finite element results. Thus, Test 3 is chosen as the best case. Further development of this case is described in the following sections.

4.6 Foundation stiffness

In the previous analyses, the stability elements were considered to be rigidly fixed to the foundation. However, this is not a realistic situation for a building. Therefore, the addition of rotational springs at the base of the stability elements is explored to represent the stiffness of the foundation. The system of equations for Test 3, with three shear walls on a two-dimensional plane, is expanded to include rotational springs at the base of each shear wall.

The spring stiffness at the foundation is generally calculated by determining the average stiffness of the foundation piles in the foundation. However, in the conceptual design, it is assumed that the number and stiffness of the foundation piles is not yet known. Instead, a recommended stiffness for the base of each stability element is calculated using a deflection requirement of $l/1000$, which is half of the maximum allowable deflection $l/500$. The recommended stiffness is calculated as follows:

$$C = M_{base} \cdot 1000$$

For each wall, a recommended stiffness is calculated for both x and y directions.

The spring stiffness is added as post-processing to the system of equations as described in Section 4.5. First, the deflection, shear force and bending moments are calculated assuming that the shear walls are rigidly fixed at the foundation. Then the recommended stiffnesses at the base of the stability elements are calculated. Finally, the total deflection of each stability element is modified as follows:

$$u_{total} = u_0 + \frac{M_{base}}{C} \cdot height$$

To get appropriate stiffnesses at the bases of the walls, a wind load needs to be applied in both the x and y direction, in two separate analyses. The maximum spring stiffnesses from these two

analyses are used as recommended spring stiffnesses. Recommended spring stiffnesses in the y-direction are taken from the analysis where wind is applied in the y-direction, and recommended spring stiffnesses in the x-direction are taken from the analysis where wind is applied in the x-direction. This is illustrated in Figure 4.19 below.

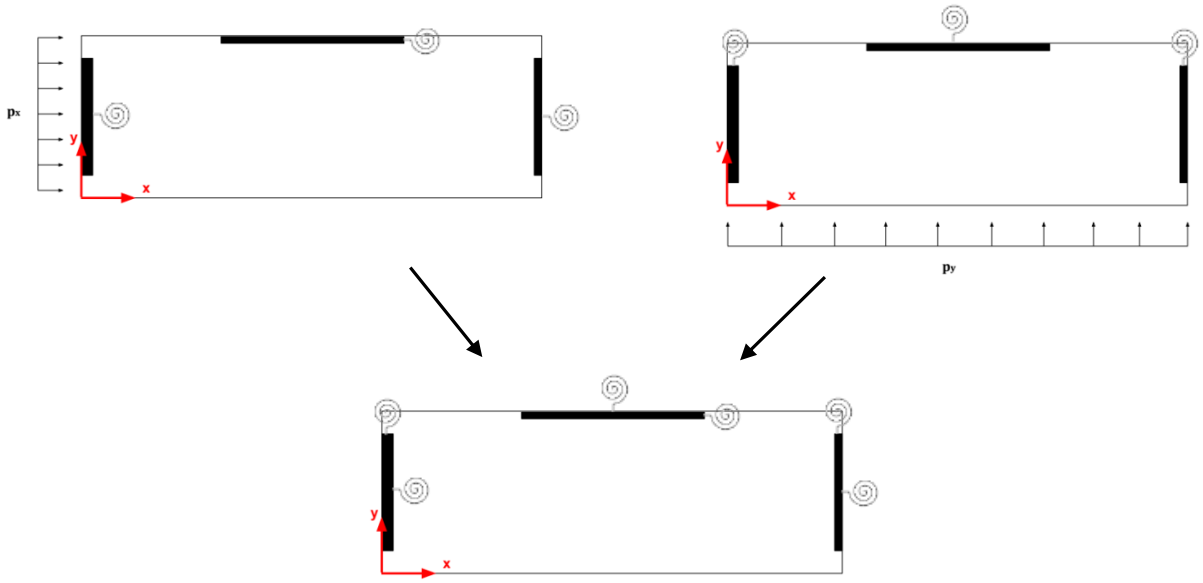


Figure 4.19 Rotational springs at the base of the shear walls

This double-analysis to calculate the foundation stiffnesses is done to prevent an excessively small foundation stiffness in a certain direction, which would be unrealistic to an actual building design. For example, if the foundation stiffnesses are calculated with the wind load only applied in the x-direction, the rotational springs in the x-direction will be very strong, but the rotational springs in the y-direction will be very weak; in some cases they will be so weak as to resemble a pin-support in the y-direction. This would invalidate the assumption that the bending moment is maximum at the base of the building. To maintain the integrity of this analysis method, it is necessary to have a sufficiently strong springs at the foundation.

This post-processing calculation is added the Maple script for Test 3 (see Appendix B). Two analyses are performed with the Maple script: one with wind load applied in the x-direction and one with wind load applied in the y-direction. Two iterations are performed for each analysis. In the first iteration, the recommended spring stiffnesses for the application of wind load in both the x and y-directions are determined based on the deflection requirement of l/1000. The maximum spring stiffness values from the two analyses are combined and are added as static inputs into the analyses. The wall deflections are then solved again using these combined spring stiffness inputs. This is done for the purpose of initial comparison; in the final tool, this process is automated. To validate Maple results, they are compared against results of a finite element analysis using GSA.

Table 4.24 shows the properties used for the calculation. The final three rows show the recommended spring stiffnesses at the base of the walls as determined in the Maple calculation. Note that all stiffness values refer to the rotation *in the direction* of the given axis, not about the axis.

Table 4.24 Properties used for test with foundation stiffness

Property	**Value**
Wind load	1.45 kN/m^2
Young's modulus	30 GPa
Building height	48 m
Floor height	3 m
Floor length	40 m
Floor width	15 m
Floor thickness	0.1 m
Shear wall 1 dimensions	6m x 0.4m
Shear wall 2 dimensions	0.2m x 8m
Shear wall 3 dimensions	6m x 0.2m
Shear wall 1 spring stiffness	x: 105204 kNm/rad y: 33395292 kNm/rad
Shear wall 2 spring stiffness	x: 24937644 kNm/rad y: 37097 kNm/rad
Shear wall 3 spring stiffness	x: 13150 kNm/rad y: 33383609 kNm/rad

The finite element model used for the GSA analysis is nearly identical to the model described in Section 4.5. As before, the shear walls are constructed using beam elements, and the floors are constructed from two-dimension Quad-4 elements. The only differences in the model are the floor thickness (0.1 metres instead of 0.26 metres) and the release conditions on the shear walls. At the base of each shear wall, all translations are restrained. The rotations around the x and y-axes at the base of the shear walls are the same as specified in Table 4.24; however, in GSA, spring stiffness are specified around the axis instead of in the direction of the axis, so the spring stiffness specified for "x" in Table 4.24 refers to the spring stiffness around the y-axis, and vice-versa for "y". Rotations around the z-axis are not restrained. To limit torsion around the shear walls, all of the shear wall elements are released around their local x-axis (this corresponds to the global z-axis). A static analysis is run on the finite element model. Figure 4.20 and Figure 4.21 show the finite element models used for the foundation stiffness tests.

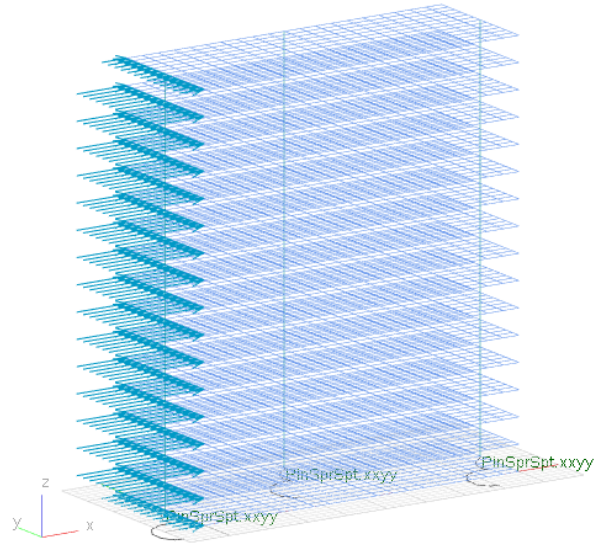


Figure 4.20 GSA model for foundation stiffness test, wind load in x-direction

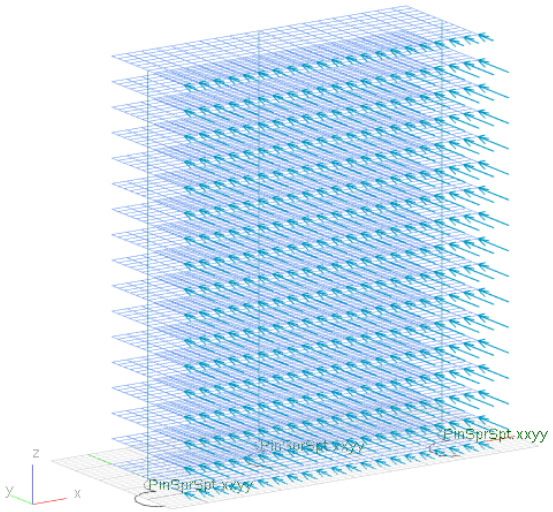


Figure 4.21 GSA model for foundation stiffness test, wind load in y-direction

Table 4.25 and Table 4.26 show the comparison of Maple and finite element analyses.

Table 4.25 Comparison of Maple and GSA for test with foundation stiffness, px = 1.45 kPa and py = 0 kPa

			Maple - rigid	**GSA**	**% difference**
Deflection at top (mm)	x	Wall 1	111.1	113.2	-1.9%
		Wall 2	104.1	103.7	0.4%
		Wall 3	111.1	113.2	-1.9%
	y	Wall 1	-19.2	-19.1	0.5%
		Wall 2	8.5	6.2	37.1%
		Wall 3	31.7	31.6	0.3%
Shear force at base (kN)	x	Wall 1	4.4	34	-87.1%
		Wall 2	1039.1	1001	3.8%
		Wall 3	0.5	8.9	-94.4%
	y	Wall 1	-194.9	-187.6	3.9%
		Wall 2	0.1	-0.3	-133.3%
		Wall 3	194.8	187.9	3.7%
Moment at base (kNm)	x	Wall 1	-105.2	-120.2	-12.5%
		Wall 2	-24937.6	-24550	1.6%
		Wall 3	-13.2	-17.9	-26.3%
	y	Wall 1	4676.7	4625	1.1%
		Wall 2	-1.7	-0.5	240.0%
		Wall 3	-4674.9	-4618	1.2%

Table 4.26 Comparison of Maple and GSA for test with foundation stiffness, px = 0 kPa and py = 1.45 kPa

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	29.3	16.6	76.5%
		Wall 2	-0.1	-0.2	-50.0%
		Wall 3	29.3	16.6	76.5%
	y	Wall 1	137.1	138.2	-0.8%
		Wall 2	181.6	183.1	-0.8%
		Wall 3	226	227.8	-0.8%
Shear force at base (kN)	x	Wall 1	1.2	-8.3	-114.5%
		Wall 2	-1.3	7.8	-116.7%
		Wall 3	0.1	0.5	-80.0%
	y	Wall 1	1391.5	1365	1.9%
		Wall 2	1.5	51.2	-97.1%
		Wall 3	1391	1368	1.7%
Moment at base (kNm)	x	Wall 1	-27.7	-5.4	413.0%
		Wall 2	31.2	32.3	-3.4%
		Wall 3	-3.5	0.8	-537.5%
	y	Wall 1	-33395.3	-33330	0.2%
		Wall 2	-37.1	-44.1	-15.9%
		Wall 3	-33383.6	-33260	0.4%

Before the results are analysed, equilibrium is checked for the system.

Table 4.27 shows the normal force calculated in GSA for each wall when wind is applied in both x and y-directions. Positive indicates tension.

Table 4.27 Normal force on shear walls from GSA for Test 3 with foundation stiffness

	GSA – wind in x-direction	GSA – wind in y-direction
Normal force Wall 1 (kN)	10.0	12.7
Normal force Wall 2 (kN)	-0.9	-24.0
Normal force Wall 3 (kN)	-9.1	11.3

When the wind is applied in the x-direction, the total external forces and bending moments are:

Table 4.28 Total external forces and bending moments, wind applied in x

Total external force in x (kN)	1044
Total external force in y (kN)	0
Total external bending moment in x (kNm)	25056
Total external bending moment in y (kNm)	0

Table 4.29 shows the equilibrium check for wind applied in the x-direction.

Table 4.29 Equilibrium check for Test 3 with foundation stiffness, wind applied in x

	Maple – rigid	**GSA**
Sum of forces in x (kN)	1044	1043.9
Sum of forces in y (kN)	0	0
Sum of bending moments in x (kNm)	-25056	-25070.1
Sum of bending moments in y (kNm)	0.1	-0.3

When the wind is applied in the y-direction, the total external forces and bending moments are:

Table 4.30 Total external forces and bending moments, wind applied in y

Total external force in x (kN)	0
Total external force in y (kN)	2784
Total external bending moment in x (kNm)	0
Total external bending moment in y (kNm)	66816

Table 4.31 shows the equilibrium check for wind applied in the y-direction.

Table 4.31 Equilibrium check for Test 3 with foundation stiffness, wind applied in y

	Maple – rigid	**GSA**
Sum of forces in x (kN)	0	0
Sum of forces in y (kN)	2784	2784.2
Sum of bending moments in x (kNm)	0	-0.3
Sum of bending moments in y (kNm)	-66816	-66814.1

Table 4.29 and Table 4.31 show that equilibrium conditions are met.

The results of the analysis are analysed. Similarly to analyses from Section 4.5, results calculated along the strong axis of each wall are considerably more accurate than results calculated along the weak axis of the walls. However, results calculated along the weak axis are generally small and not governing in design, so the results are considered acceptable.

4.7 Vertical load and second-order effect

As a final addition to the analysis, vertical load is included to the analysis. The vertical load causes a second-order effect on the deflection and bending moment of the stability elements.

A factor for the second-order effect is calculated for the entire system of stability elements in the x and y directions. The overall foundation stiffness of the building in the x and y-directions is estimated as a weighted average of the foundation stiffnesses of the individual walls, based on their bending stiffness. The calculation for the second-order factor in the x-direction for the building configuration shown in Figure 4.16 is shown as follows. $Q_{cr,x}$ refers to the critical downward load calculated for the x-direction. $Q_{cr,fx}$ is the contribution from the foundation stiffness in the x-direction, and $Q_{cr,bx}$ is the contribution from the bending of the building in the x-direction. N_{total} refers to the total vertical load on the stability elements combined. The calculation method is taken from Ham et. al (2017).

$$C_x = \frac{C_{x1}.EI_{x1} + C_{x2}.EI_{x2} + C_{x3}.EI_{x3}}{EI_{x1} + EI_{x2} + EI_{x3}}$$

$$Q_{cr,fx} = \frac{2C_x}{height}$$

$$Q_{cr,bx} = \frac{8(EI_{x1} + EI_{x2} + EI_{x3})}{height^2}$$

$$\frac{1}{Q_{cr,x}} = \frac{1}{Q_{cr,fx}} + \frac{1}{Q_{cr,bx}}$$

$$n_x = \frac{Q_{cr,x}}{N_{total}}$$

$$second\ order\ factor\ in\ x = \frac{n_x}{n_x - 1}$$

As a final post-processing calculation, both the deflection and the bending moment in the x and y-directions are multiplied by their respective second order factor for the x or y-direction.

To verify the accuracy of this method, the Maple results are compared against the results of a finite element analysis from Oasys GSA. For this analysis, the same properties are used for the analysis as specified in Table 4.24, but additionally a dead load is added to each wall. No live load is added to the analysis. All loads are unfactored.

For each wall, the dead load on the wall consists of the self-weight of the wall itself, plus the self-weight of a certain floor area prescribed to each wall. For this test analysis, the following floor areas are prescribed to each shear wall.

Table 4.32 Floor area supported by shear walls

Wall	**Supported floor area (m^2)**
Wall 1	30
Wall 2	40
Wall 3	30

These floor areas are chosen simply to represent a hypothetical situation. The actual floor area that each shear wall supports depends on the arrangement of columns in the building and should be determined by the structural engineer. Since columns are not represented in this tool, a simple estimate of the possible floor area to be supported by each shear wall is chosen for the test analysis. The supported floor areas in Table 4.32 Floor area supported by shear walls lead to the following distributed dead loads on each of the shear walls (including the self-weight of the wall itself).

Table 4.33 Dead loads on the stability elements

Wall	Distributed dead load (kN/m)
Wall 1	83.3
Wall 2	71.9
Wall 3	53.9

The finite element model constructed in Oasys GSA is identical the model described in the previous section, with an additional gravity load added on the shear walls as specified in Table 4.33.

A static P-delta analysis is run on the finite element model. Figure 4.22 and Figure 4.23 show the finite element models for this analysis.

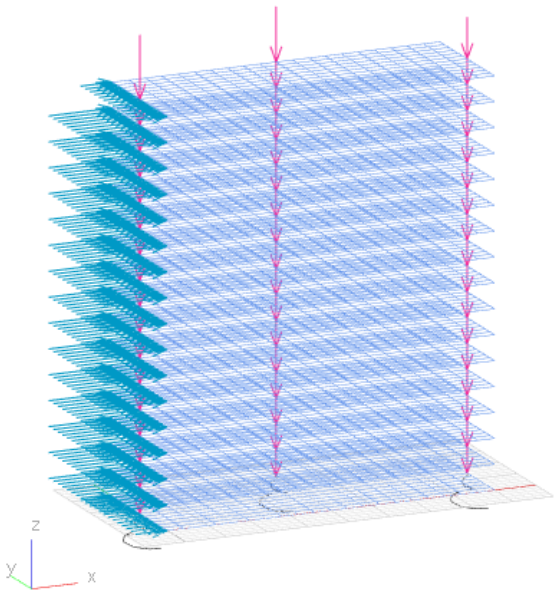


Figure 4.22 GSA model for second order effect test, wind load in x-direction

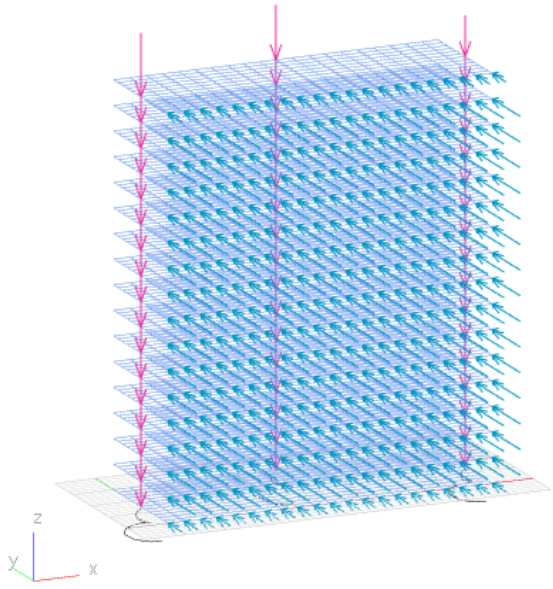


Figure 4.23 GSA model for second order effect test, wind load in y-direction

Table 4.34 and Table 4.35 show the comparison of the governing results only from the Maple and GSA analyses.

Table 4.34 Comparison of Maple and GSA for second order effect test, p_x = 1.45 kPa and p_y = 0 kPa, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	x	Wall 1	113.5	115.7	-1.9%
		Wall 2	106.3	105.9	0.4%
		Wall 3	113.5	115.7	-1.9%
Shear force at base (kN)	x	Wall 2	1039.1	1009	3.0%
Moment at base (kNm)	x	Wall 2	-25470.9	-25050	1.7%

Table 4.35 Comparison of Maple and GSA for second order effect test, p_x = 0 kPa and p_y = 1.45 kPa, governing values only

			Maple - rigid	GSA	% difference
Deflection at top (mm)	y	Wall 1	139.3	140.1	-0.6%
		Wall 2	184.5	186.1	-0.9%
		Wall 3	229.8	231.8	-0.9%
Shear force at base (kN)	y	Wall 1	1391.5	1366	1.9%
		Wall 3	1391	1372	1.4%
Moment at base (kNm)	y	Wall 1	-33942.9	-33730	0.6%
		Wall 3	-33931	-33760	0.5%

All of the results from the Maple analysis are within a 3% difference from the finite element results, which is a very small difference. On this basis, this method of applying the second-order effect is determined to be suitable for the analysis.

4.8 Automation of calculation method

The goal of StructuralComponents 6 is to create a tool that allows a user to create customised configurations of shear walls and cores, and analyse these configurations. Therefore, the analysis method developed in the previous sections is automated for various numbers and locations of stability elements.

To review, the deflection of each shear wall is calculated by solving a system of three differential equations. These three differential equations represent the force equilibrium in the x-direction, the force equilibrium in the y-direction, and the bending moment equilibrium of the rigid floor system on the x-y plane. The deflection of each shear wall is expressed along the z-axis, perpendicular to the plane of the equilibrium equations. For a building with "i" stability elements, this system of differential equations can be expressed as below. Note that w and l refer to the width and length of the floor, respectively. u_x, u_y and ϕ refer to the deflection and the rotation of the entire floor system at its rotational centre. x_{centre} and y_{centre} refer to the x and y coordinates, respectively, of the rotational centre of the floor system.

Local system of equations:

$$EI_{x1}\frac{d^4u_{x1}}{dz^4} + EI_{x2}\frac{d^4u_{x2}}{dz^4} + \cdots + EI_{xi}\frac{d^4u_{xi}}{dz^4} = p_x \cdot w$$

$$EI_{y1}\frac{d^4u_{y1}}{dz^4} + EI_{y2}\frac{d^4u_{y2}}{dz^4} + \cdots + EI_{yi}\frac{d^4u_{yi}}{dz^4} = p_y \cdot l$$

$$\left(a_1 \cdot EI_{y1}\frac{d^4u_{y1}}{dz^4}\right) - \left(b_1 \cdot EI_{x1}\frac{d^4u_{x1}}{dz^4}\right) + \left(a_2 \cdot EI_{y2}\frac{d^4u_{y2}}{dz^4}\right) - \left(b_2 \cdot EI_{x2}\frac{d^4u_{x2}}{dz^4}\right) + \cdots + \left(a_i \cdot EI_{yi}\frac{d^4u_{yi}}{dz^4}\right) - \left(b_i \cdot EI_{xi}\frac{d^4u_{xi}}{dz^4}\right) = p_y \cdot l \cdot \left(x_{centre} - \frac{l}{2}\right) - p_x \cdot l \cdot \left(y_{centre} - \frac{w}{2}\right)$$

Relationships between local and global displacements for wall "i":

$$u_{xi} = u_x - \phi \cdot b_i$$

$$u_{yi} = u_y + \phi \cdot a_i$$

where

$$a_i = x_{centre} - x_i$$

$$b_i = y_{centre} - y_i$$

The deflections are initially calculated assuming the stability elements are rigidly connected at their base. The rotation, bending moment, and shear force for each stability element is derived from this initial deflection. The boundary conditions for the system of equations can be expressed in terms of the global deflections and rotation u_x, u_y and ϕ as follows:

z = 0:	$u_x = 0$	$\frac{du_x}{dz} = 0$
	$u_y = 0$	$\frac{du_y}{dz} = 0$
	$\phi = 0$	$\frac{d\phi}{dz} = 0$
z = height:	$\frac{d^2u_x}{dz^2} = 0$	$\frac{d^3u_x}{dz^3} = 0$
	$\frac{d^2u_y}{dz^2} = 0$	$\frac{d^3u_y}{dz^3} = 0$
	$\frac{d^2\phi}{dz^2} = 0$	$\frac{d^3\phi}{dz^3} = 0$

After the initial deflection, shear force and bending moment is calculated using the method above, the recommended spring stiffness at the base of each wall is determined. The additional deflection caused by the foundation springs is added to previously-calculated deflection for each wall. Then, vertical load is applied. The normal force on each wall is calculated. Lastly, the second order effect factor is calculated, and the deflection and bending moment are multiplied by this factor.

The automation of the analysis is performed in Python, using SymPy (SymPy Development Team, 2018). SymPy is a Python library that allows symbolic mathematics to be performed in Python.

The calculation described above is performed in Python using a loop. For each wall that is added to the analysis, new terms are added to the left-hand-side of each equilibrium equation. After the system of equations is constructed for a given number of walls, it is solved using SymPy's dsolve function to determine the initial deflections, shear forces and bending moments on each of the walls, with the global boundary conditions as specified above.

The operation described above is performed twice; once with wind load applied in the x-direction and once with wind load applied in the y-direction. The rotational spring stiffness at the base of each stability element is determined for both analyses, using the calculated bending moment at the base and a deflection requirement of height/1000. The maximum spring stiffness values from these two analyses are combined together and are used as the "recommended foundation stiffness" values for the rest of the analysis. As previously described, this double-analysis is necessary to calculate realistically large foundation stiffnesses for both x and y directions.

The user defines whether they would like the wind load to be applied in the x or y direction (this is further described in Chapter 5). If the user chooses to apply the wind load in the x-direction, then the deflection, shear force and bending moment previously calculated for wind in the x-direction is used for the remainder of the analysis, and the values calculated for wind in the y-direction are ignored.

The additional deflection caused by the foundation stiffness is calculated and added to the original deflection for each wall, for the given direction of wind application. Finally, the second order factor is determined, and the deflection and bending moments are multiplied by this factor.

The Python code is implemented using the "Python Script" component in Grasshopper. This is discussed further in Chapter 5. The Python code used for automation is provided in Appendix C.

5 System architecture

5.1 Objectives

The goal of StructuralComponents 6 is to develop a conceptual design tool that allows a user to build and analyse custom configurations of floors and stability elements. StructuralComponents 6 is based on StructuralComponents 5 and is thus implemented in a similar way: the tool comprises of a collection of components in Grasshopper, and the structural analysis is developed in Python via Grasshopper's "Python Script" component. This section describes the main objectives for the development of the tool.

In Chapter 3, a high-level framework of the conceptual design process was developed. Through a study of this framework, some important qualities for conceptual design tools were identified. These qualities are listed below:

1. Quick and easy generation of design alternatives
2. Easy changes to the design
3. Parallel investigation/comparison of alternatives
4. Constant feedback on the design criteria
5. Visualisation of design and analysis results

These five qualities have been used as drivers in the design of the framework for StructuralComponents 6. The objectives in the development of StructuralComponents 6 in relation to these design drivers are described below.

1. Quick and easy generation of design alternatives

To make design alternatives quick and easy to generate, the process of model construction is made simple and intuitive. Additionally, the amount of user inputs is limited to avoid clutter in the tool; only inputs that are necessary for design are included.

2. Easy changes to the design

The tool consists of a pre-defined logic that can be modified by a limited number of changeable parameters. In this way, the user can easily change a single parameter of their existing design, without needing to entirely reconstruct the design. Changeable parameters in the tool include the material properties, loads applied on the building and the building geometry.

3. Parallel investigation/comparison of alternatives

It is possible to compare different design alternatives side-by-side in the tool. Additionally, different design alternatives can be saved, reopened and modified at a later time.

4. Constant feedback on the design criteria

StructuralComponents 6 provides the user with feedback on the structural integrity of their design. The tool performs the following checks:

1. Stiffness: The deflection of the building should not exceed height/500
2. Strength: The stress in the material should not exceed the material's strength
3. Stability: There should be no tension in the foundation

With every new change to the design, these checks are re-calculated and returned to the user.

5. Visualisation of design and analysis results

The stability elements and floors are visualised in the Rhinoceros 3D environment so the designer can see what their building looks like. The deflection, shear force, bending moment and normal force is displayed along each stability element. Additionally, the checks described in the point above are visualised on the stability elements; if one of these checks fails, the offending stability element turns red to inform the user that their design is not adequate.

The following sections describe the details of the tool: first, how the analysis method described in Chapter 4 is transferred into Grasshopper; secondly, the user interface is described.

5.2 Structural analysis in Grasshopper

Transference into Grasshopper

The first step in developing the tool is to transfer the calculation method described in Chapter 4 into Grasshopper. An overview of the calculation method in Python has been described in Section 4.8. As stated in this section, the structural analysis is implemented in Python using SymPy, a Python library for symbolic mathematics.

Grasshopper contains its own "Python Script" component that allows users to write script in Python. However, this component does not support SymPy. In order to use SymPy in the Python Script component, SymPy must be imported remotely into Grasshopper. This is done using the Grasshopper plug-in component "GH Python Remote" (Digital Structures, 2018).

GH Python Remote is a component developed by Cuvilliers at the Massachusetts Institute of Technology. This component connects Grasshopper's Python Script component to an external version of Python, which allows the user to import external Python libraries, such as SymPy or NumPy, into the Python Script component. In order to use SymPy in the Python Script component, the GH Python Remote component must be placed on the Grasshopper canvas and set to run. An image of the GH Python Remote component importing SymPy is shown in Figure 5.1 (Digital Structures, 2018).

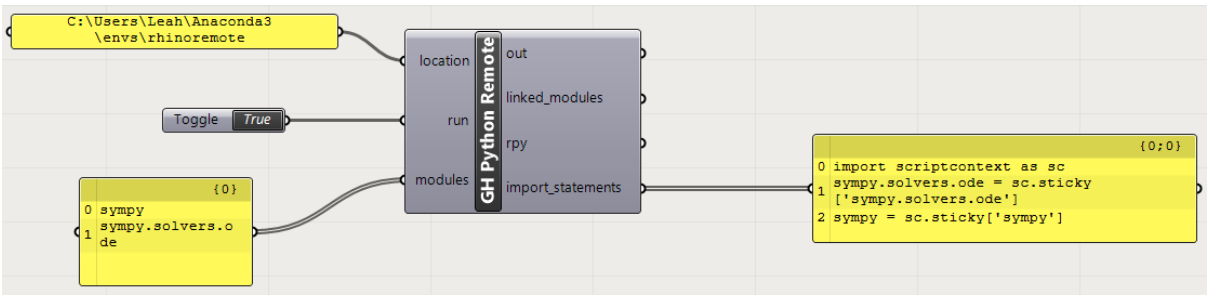


Figure 5.1 GH Python Remote component

The method of structural analysis with Python and SymPy has been described in Section 4.8. Using this method, the Python Script component is used to calculate the deflection, shear force and bending moment distribution along the height of each shear wall in the user's custom configuration. The script also includes a calculation for the normal force at the base of each shear wall. The script also calculates the recommended foundation stiffnesses in the x and y direction at the base of each shear wall, and returns these values to the user (this is described in more detail in Section 5.3).

After calculating the distribution for deflection, shear force and bending moment, the Python component calculates nodal values of these properties at each floor level, for each wall. These nodal values are used to construct curves along each shear wall, which visually represent the graph of deflection, shear force and bending moment along each wall. The visualisation of results is further explained in Section 5.3.

Checks for stiffness, strength and stability

After calculating the deflections, forces and bending moments on all the stability elements, the tool must provide the user with feedback on whether or not their building configuration is adequate. To test the adequacy of the building, three checks are performed. These checks are performed for a concrete structure; f_{ck} refers to the characteristic cylinder stress of the concrete. The checks were implemented mathematically into the calculation as shown below:

1. Stiffness

$$abs(u_{x,y}) \leq \frac{height}{500}$$

2. Strength

a) Compressive strength

$$\frac{N}{A} + abs\left(\frac{M_{x,y}}{W_{x,y}}\right) \leq \frac{f_{ck}}{1.5}$$

b) Shear strength

$$abs(V_{x,y}) \leq \left(v_{min} + 0.15 \cdot \frac{N}{A}\right) \cdot A$$

where:

$$v_{min} = 0.035 \cdot k^{\frac{3}{2}} \sqrt{f_{ck}}$$

$$k = 1 + \sqrt{\frac{200}{d}} \leq 2 \quad \text{with d in mm [4]}$$

3. Stability

$$abs\left(\frac{M_{x,y}}{W_{x,y}}\right) - \frac{N}{A} \leq 0$$

[4] From Section 6.2.2 of NEN 1992-1-1 (2005) "Members not requiring design shear reinforcement"

For the stiffness check, the deflection in both x and y directions is checked at the top of each shear wall against the performance metric. For the strength check, the maximum compressive and shear stresses at the base of each shear wall are checked against the performance metrics. For the shear check, the property "d" is the wall dimension parallel to the force application, for x and y respectively.

Note that the shear check does not indicate failure of the building, but indicates that extra shear reinforcement is necessary in the walls.

For the stability check, tensile stress is checked for each shear wall. If there is any tension in the foundation (the check is greater than zero), the stability check fails.

If an individual wall check fails, the user is alerted with a message for that particular wall. Additionally, the wall colour is defined for each wall based on whether the wall passes or fails a check. If the wall passes a check, its colour is grey, and if it fails a check, its colour becomes red. The warnings and visualisation of checks is further described in Section 5.3.

The entire Python script including the derivation of the deflection, forces and bending moments and the structural integrity checks is provided in Appendix C.

5.3 User Interface

The user interface of StructuralComponents 6 consists of a collection of components in Grasshopper. The building geometry and analysis results are visualised in the Rhinoceros 3D environment. Four different Grasshopper components have been developed as part of the tool. These components are briefly described below.

Construct shear wall

The "Construct shear wall" component allows the user to define the cross-section and location of a shear wall on the xy-plane. Each time the user wants to add a new shear wall to their model, they must add another "Construct shear wall" component. The model may include as many "Construct shear wall" components as the user likes, but must include a minimum of three to ensure stability of the building.

Construct floor

The "Construct floor" component allows the user to define the dimensions of the floor and location of the floor on the xy-plane. In this tool, only a rectangular floor can be modelled. Only one "Construct floor" component is needed for a single model.

Calculator

The "Calculator" component calculates the deflection, bending moment, shear force and normal force along the shear walls given user inputs. The "Calculator" component also performs checks for stiffness, strength and stability.

The "Calculator" component contains a "Messages" output, which indicates the recommended foundation stiffness for each wall and provides a message if a check for stiffness, strength or stability fails.

Visualiser	The "Visualiser" component visualises the building geometry. It also visualises the deflection, bending moment, shear force and normal force along the shear walls. Additionally, it visualises the check for stiffness, strength and stability; if one of these checks fails for a certain shear wall, that wall turns red.

Due to time constraints in the project, a "Construct core" component was not created; however a core can be modelled in the tool using four shear walls.

The construction of a model with three shear walls is illustrated in Figure 5.2.

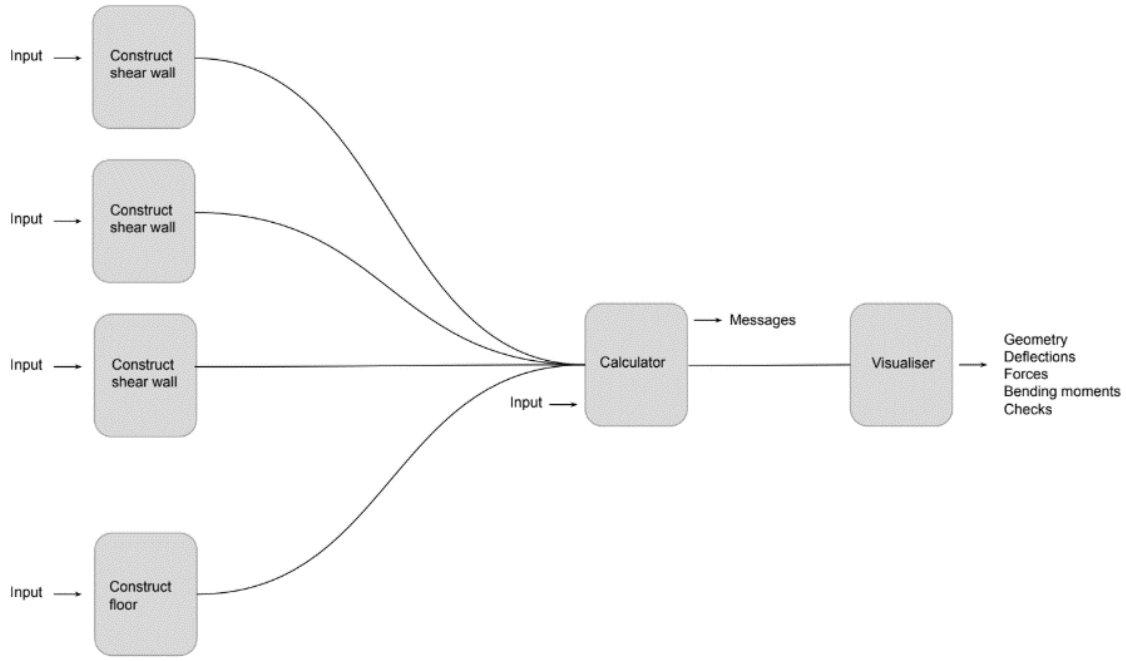


Figure 5.2 Model construction in StructuralComponents 6

The "Construct shear wall" and "Construct floor" components can be grouped together as the "Floorplan construction" components. The "Floorplan construction" components, "Calculator" component and "Visualiser" component are described in more detail in the following sections.

Floorplan construction components

The Floorplan construction components consist of the "Construct shear wall" component and the "Construct floor" component. These components are used to define the floorplan of the building on the xy-plane. These components also gather some additional information about the shear walls and floor; this information is then entered into the Calculator component for analysis.

The "Construct shear wall" component is shown in Figure 5.3.

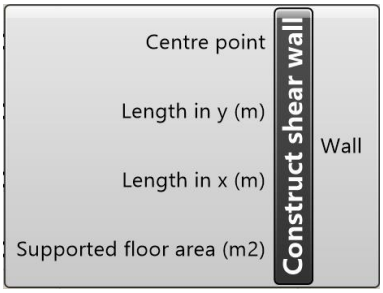


Figure 5.3 Construct shear wall component

The "Construct shear wall" component has four inputs: Centre point, Length in x, Length in y, and Supported Floor Area. The first three inputs are used to define the location and cross-section of the shear wall. Note that the centre point of the shear wall must lie on the xy-plane. The final input, Supported Floor Area, specifies the floor area that is supported by the shear wall. The supported floor area depends on the column configuration of the building and should be defined by the engineer. The Supported floor area input refers to the supported area of one single floor.

The "Construct floor" component is shown in Figure 5.4.

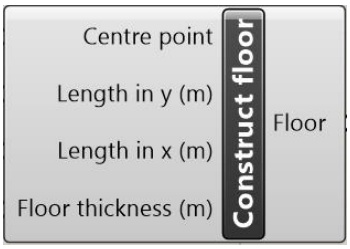


Figure 5.4 Construct floor component

The "Construct floor" component has four inputs: Centre point, Length in y, Length in x and Floor thickness. These inputs are used to define the location and dimensions of a single floor.

These two components can be viewed as the "start" components for the analysis. After the shear walls and floor are constructed, they are added as inputs into the next component, the Calculator.

Calculator component

The "Calculator" component is shown in Figure 5.5.

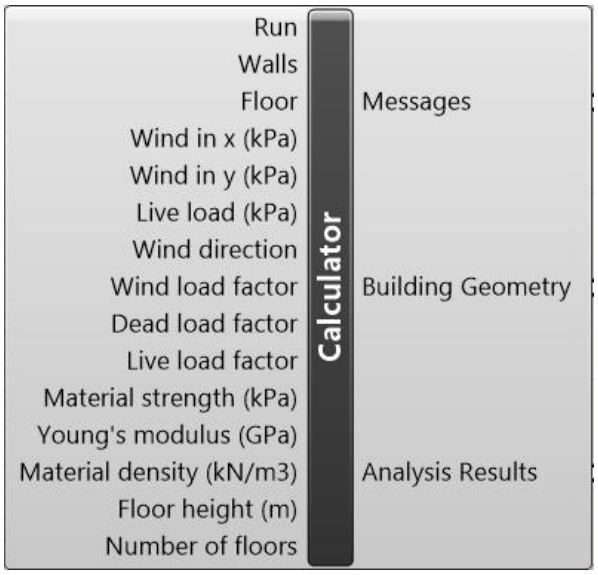


Figure 5.5 Calculator component

The inputs for the "Calculator" component are described below:

Run	Boolean toggle indicating "True" or "False". If True, the analysis runs, and if False, the calculator stops.
Walls	Collects the "Construct shear wall" components. To add more than one "Construct shear wall" component here, the user must press down the "Shift" key while entering components.
Floor	Collects the "Construct floor" component.
Wind load in x-direction & wind load in y-direction	Wind load in the x-direction and wind load in the y-direction in kPa. Both values must be specified to properly calculate the recommended spring stiffnesses at the foundation.
Live load	Live load on the floors in kPa.
Wind direction	Direction of wind application, either x or y.
Wind, Dead and Live load factor	Load factors to apply to the wind, dead and live load. If not specified, they are set by default to 1.
Material strength	Design compressive cylinder strength of concrete, f_{ck}, in kPa.
Young's modulus	Young's modulus of the building material in GPa.[5]

[5] This value is expressed in GPa rather than kPa because the maximum value of a Grasshopper number slider is limited to one million, which is not large enough to express the Young's modulus in kPa.

Material density	Density of the building material in kN/m^3.
Floor height	Floor height of the building in metres.
Number of floors	Number of floors in the building.

The Calculator component has three outputs: Messages, Building Geometry, and Analysis Results.

As previously explained, the Messages output shows the recommended foundation stiffness at the base of each wall. It also includes a warning if a check for stiffness, strength or stability fails. The user can connect a Grasshopper "Panel" component to the Messages output to see the messages. Figure 5.6 shows an example of the output message for a configuration with three walls.

```
                                                                    {1;0}
 0 Running...
 1
 2 Wall 1
 3 Foundation stiffness in x: 105204 kNm/rad
 4 Foundation stiffness in y: 33395292 kNm/rad
 5 Warning! Deflection of Wall 1 in y-direction exceeds height/500
 6 Warning! Shear force on Wall 1 in y-direction exceeds material shear strength;
   shear reinforcement required
 7
 8 Wall 2
 9 Foundation stiffness in x: 24937644 kNm/rad
10 Foundation stiffness in y: 37097 kNm/rad
11 Warning! Deflection of Wall 2 in y-direction exceeds height/500
12
13 Wall 3
14 Foundation stiffness in x: 13150 kNm/rad
15 Foundation stiffness in y: 33383609 kNm/rad
16 Warning! Deflection of Wall 3 in y-direction exceeds height/500
17 Warning! Compressive stress of Wall 3 in y-direction exceeds material compressive
   strength
18 Warning! Shear force on Wall 3 in y-direction exceeds material shear strength;
   shear reinforcement required
19
20 Forces and displacements solved.
21 Analysis time: 5.60776519775 seconds
```

Figure 5.6 "Messages" output for a configuration with three shear walls

The Building Geometry output contains information about the building geometry that is processed in the Calculator component. This output contains the geometry of the shear walls and floors, and the colour of the shear walls (which depends on the outcome of the checks).

The Analysis Results output contains the nodal results at all floor levels for deflection, bending moment, shear force and normal force on each shear wall, as calculated by the Calculator.

The Building Geometry and Analysis Results outputs are entered directly as inputs into the Visualiser component, where they can be visualised.

Visualiser component

The Visualiser component is shown in Figure 5.7.

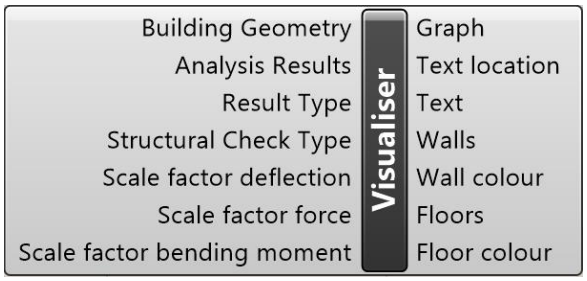


Figure 5.7 Visualiser component

The first two inputs, Building Geometry and Analysis Results, are taken directly from the identically-named outputs of the Calculator component.

The Result Type input indicates which result the user would like to see. There are seven options for result type, as follows:

1. Deflection in x
2. Deflection in y
3. Shear force in x
4. Shear force in y
5. Bending moment in x
6. Bending moment in y
7. Normal force

The Structural Check Type input indicates which structural check the user would like to see. There are five options for structural check type, as follows:

1. None
2. Stiffness
3. Strength (compressive)
4. Strength (shear)
5. Stability

The final three inputs indicate the scale factors for visualisation for the deflection, force (both shear and normal) and bending moment. By default, the scale factor for deflection is 1. The default visualisation equivalency for force is 5000 kN per metre on the Grasshopper grid. The default visualisation equivalency for bending moment is 50000 kNm per metre on the Grasshopper grid. The size of the graphs for deflection, force and bending moment can be increased using the scale factor inputs. If the user does not specify a scale factor input, it is set by default to 1.

The first three outputs, Graph, Text Location and Text collectively show the graphs for deflection, force or bending moment. The results which appear in this output depend on the user's selection for result type. The graph is visualised using two Grasshopper components: "Custom Preview" and "Text Tag". The Graph output is entered into the "Custom Preview" component, and the outputs "Text Location" and "Text" are entered into the "Text Tag" component.

The last four outputs show the geometry of the building. These can be visualised with two "Custom Preview" components, one for the shear walls and one for the floor. The visualisation of these outputs is shown in Figure 5.8. Note that a blue colour "swatch" has been used to define the colour for the first "Custom Preview" component; this is simply done for aesthetic purposes and

is not important to the analysis. If no colour is specified in the "Custom Preview", the colour is pink by default.

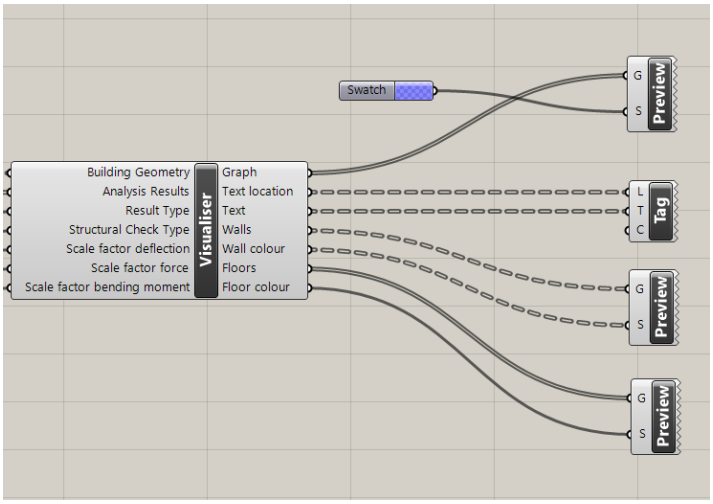


Figure 5.8 Visualisation of results

To "turn off" the view on any of the visualisation outputs, the user can either delete the "Custom Preview" or "Text Tag" component, or right-click the component and deselect the option "Preview".

The following images show some different visualisations for a building with three shear walls and wind load applied in the y-direction. Figure 5.9 shows the building geometry with results of a check for compressive strength. The red colour indicates that the compression in the edge of Wall 3 exceeds the compressive strength of the concrete. Figure 5.10 shows the deflection in the y-direction on the walls with a scale factor of 50, without floors shown.

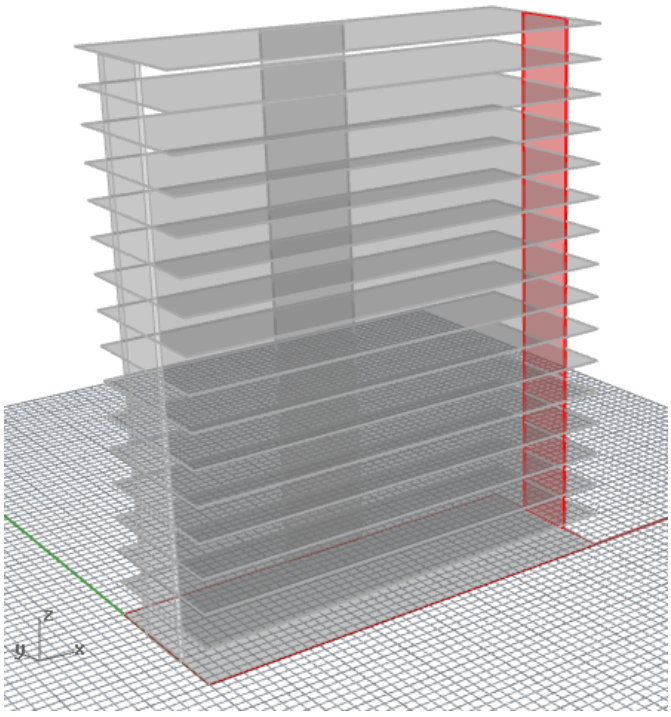


Figure 5.9 Visualisation of compressive strength check

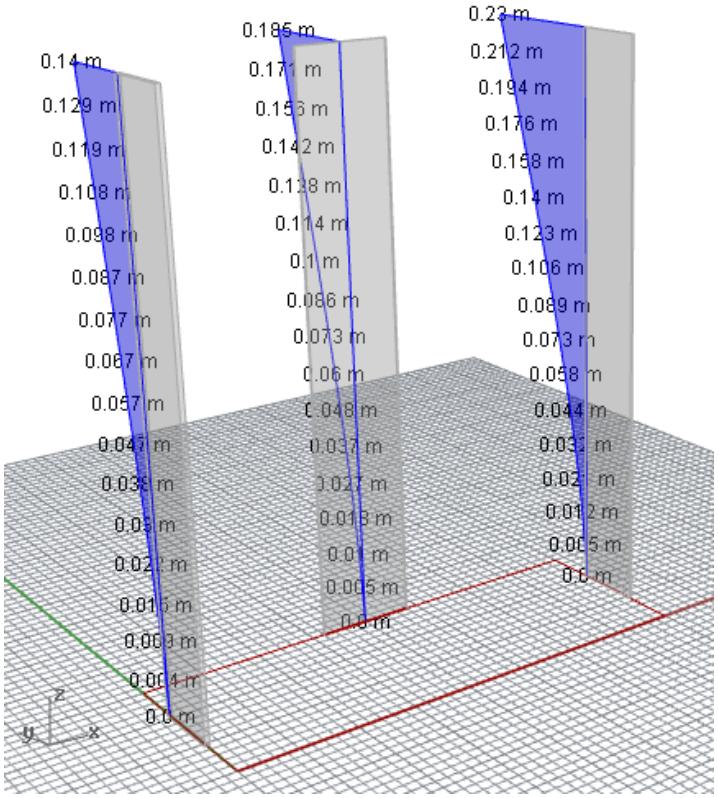


Figure 5.10 Visualisation of deflection in the y-direction, scale factor 50

6 Case Study: EWI Building

6.1 Model construction

To provide a demonstration of the use of the tool, a case study is performed. The case study is performed on an existing building, the building of the Electrical Engineering, Mathematics and Computer Science ("EWI" building) at the Delft University of Technology. The building is composed of a 90-metre-tall tower and several lower buildings; for the purpose of this case study only the tower will be modelled. An image of the EWI building is shown in Figure 6.1.

Figure 6.1 EWI Building, Delft University of Technology

The tower of the EWI building is principally supported in the lateral direction by nine shear walls and two core-like structures which run from the building foundation to the top of the tower. The following image shows a simplified illustration of the stability system.

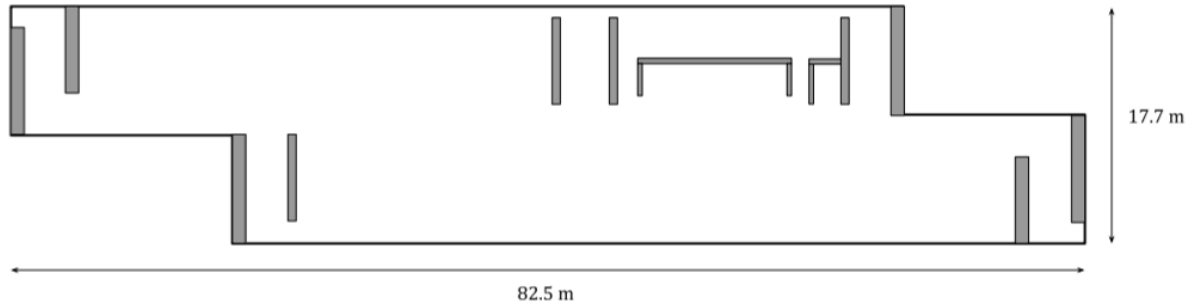


Figure 6.2 Layout of shear walls on EWI building

StructuralComponents 6 can only model a building with a rectangular floorplan and can therefore not model the exact floorplan shown in Figure 6.2. Since the tool assumes that the floors are infinitely rigid, the shape of the floor does not affect the structural analysis. However, the size of the floor affects the amount of wind that is applied on the building, since the wind is taken as a uniformly-distributed load along the building's length and width. The floorplan should be changed such that the application of wind load will remain the same as for the original building. The following equivalent floorplan is used for the case study. The walls are labelled from left to right as they will be constructed in the tool. The cores are separated into representative constituent shear walls.

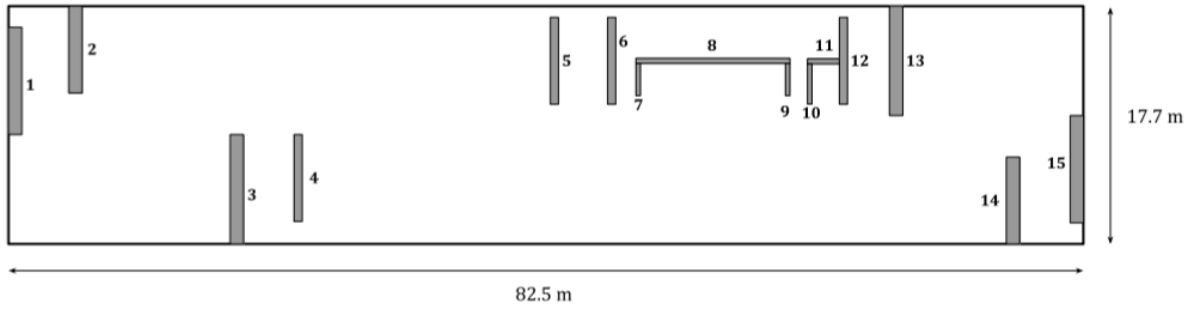


Figure 6.3 Equivalent building layout for case study

The properties of the shear walls as specified as inputs into the case study are shown in Table 6.1. The walls from 1 to 15 are specified from left to right in Figure 6.3. The origin is specified at the bottom-left corner of the floor. "x" and "y" refer to the x and y coordinates of the centre point of the shear wall. The supported floor area is estimated based on the layout of columns on the floorplan. Walls 7 to 12 support the elevators of the building in addition to areas of the floor. Extra dead load and live load from an elevator has not been incorporated in the tool, so to account for this extra load, the supported floor area is slightly exaggerated for these walls.

Table 6.1 Shear wall properties for case study

Wall	x (m)	y (m)	Length in y (m)	Length in x (m)	Supported floor area (m^2)
1	0.5	11.4	7.74	1	25
2	4.6	13.3	6.39	1	45
3	16.7	3.8	7.73	1	25
4	21	4.5	6.27	0.6	36
5	45.5	13.3	6.27	0.6	36
6	49.3	13.3	6.39	0.6	25
7	50.9	12	2.4	0.2	10
8	56.2	13.4	0.3	11	36
9	61.6	12	2.4	0.2	10
10	63	11.7	3.1	0.2	10
11	64.2	13.4	0.3	2.5	10
12	65.7	13.1	6.27	0.6	25
13	70	13.8	7.73	1	25
14	78.1	4.9	6.39	1	45
15	82	6.8	6.64	1	25

The floor is 82.5 metres long and 17.7 metres deep. In the model, the bottom-left corner of the floor is at the origin, so the coordinates of the floor's centre-point are (41.25, 8.85) on the xy-plane. The thickness of the floor is not known, so a floor thickness of 0.2 metres is used as a representative thickness in this case study.

The material of the walls and floors is C30/37 in this case study: Young's modulus, material density and material strength are specified accordingly. A wind load of 1.6 kPa is used for the analysis; this is the peak velocity wind pressure for a 90-metre-tall building in a Rural part of Area II in the Netherlands. A rural area is chosen because there are no other tall buildings around the EWI building. A live load of 2.5 kPa is used for the floors; this is the specified distributed live load for floors in an office building. For this analysis, the dead load is the self-weight of the concrete of the building. The building has twenty-three floors, mostly with a floor-to-floor height of 3.75 metres; however the ground floor is twice as tall as the rest of the floors. For the case study, the building is estimated to have twenty-four floors of 3.75 metre height each. Table 6.2 shows the additional input properties used for the case study.

Table 6.2 Additional input properties used for case study

Property	**Value**
Wind in x (kPa)	1.6
Wind in y (kPa)	1.6
Live load (kPa)	2.5
f_{ck} (kPa)	30000
Young's modulus (GPa)	33
Material density (kN/m^3)	24.5
Floor height (m)	3.75
Number of floors	24

To properly analyse the building, four analyses need to be run:

1. Wind load in x-direction, SLS
2. Wind load in x-direction, ULS
3. Wind load in y-direction, SLS
4. Wind load in y-direction, ULS

The deflection requirements will be checked for the Service Limit State, and the strength and stability requirements will be checked for the Ultimate Limit State.

For the Ultimate Limit State, a load factor of 1.35 is used for the dead load, and a load factor of 1.5 is used for the live load and wind load.

The construction of the model and output of the analyses is described below. Figure 6.4 shows the entire model construction for the case study.

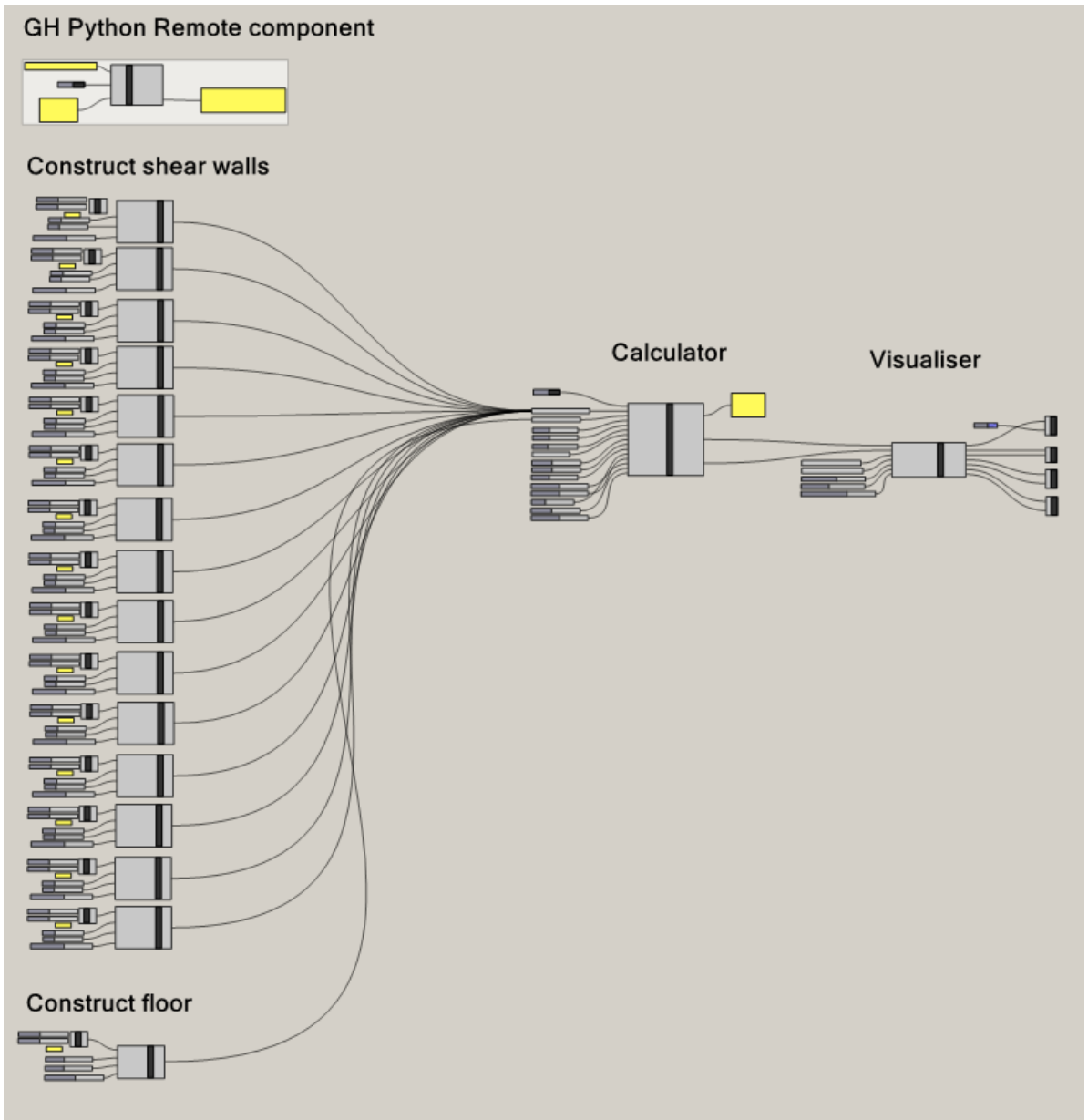


Figure 6.4 Model construction in Grasshopper for case study

As can be seen in Figure 6.4, the GH Python Remote component must be present on the Grasshopper canvas in order to allow the model to run. The individual shear walls must be created with separate "Construct shear wall" components, and then added individually into the Calculator. Only one "Construct floor" component is needed for the analysis. Figure 6.5 shows the "Construct shear wall" component for Wall 1, and Figure 6.6 shows the "Construct floor" component for the whole building.

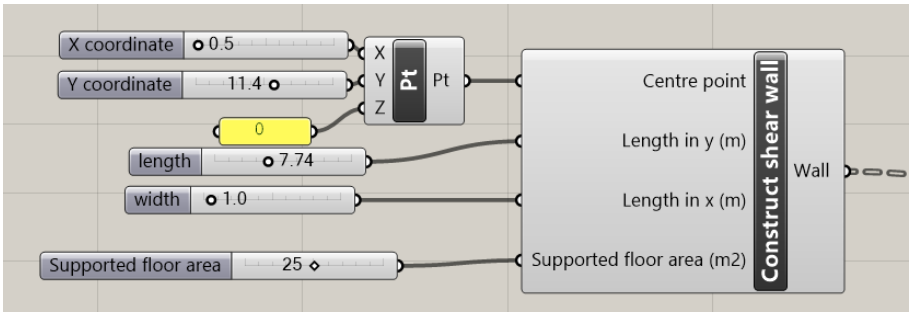


Figure 6.5 Construct shear wall component in case study

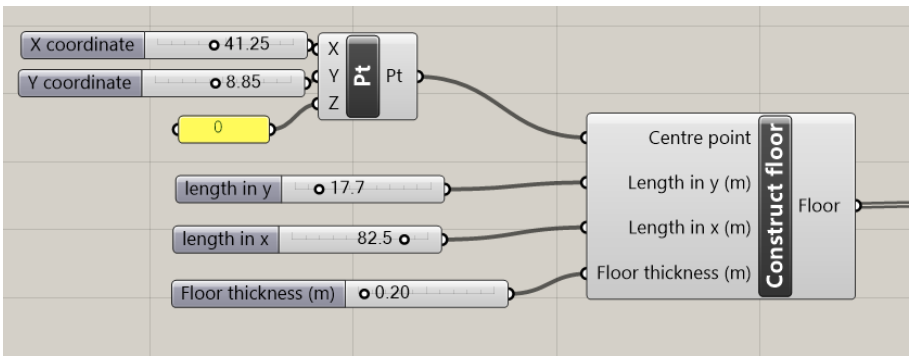


Figure 6.6 Construct floor component in case study

The "Pt" component shown in the left of both images is a Grasshopper component that allows the user to create a point in Rhinoceros with specified x, y and z coordinates. For both the shear walls and floor, the x and y coordinates may vary, but the z-coordinate must always be zero.

The user does not necessarily need to specify the coordinates of the centre point this way. Grasshopper contains empty components such as the "Data" or "Point" component which can be used to reference information originating from Rhinoceros. If the user prefers, they can draw points directly in Rhinoceros, and reference these points into Grasshopper to be used as input into their "Construct shear wall" or "Construct floor" component. This gives the user to ability to change the location of their walls and floor by dragging around the centre points directly in Rhinoceros, rather than changing number sliders in Grasshopper.

Figure 6.7 shows the Calculator component used in the case study.

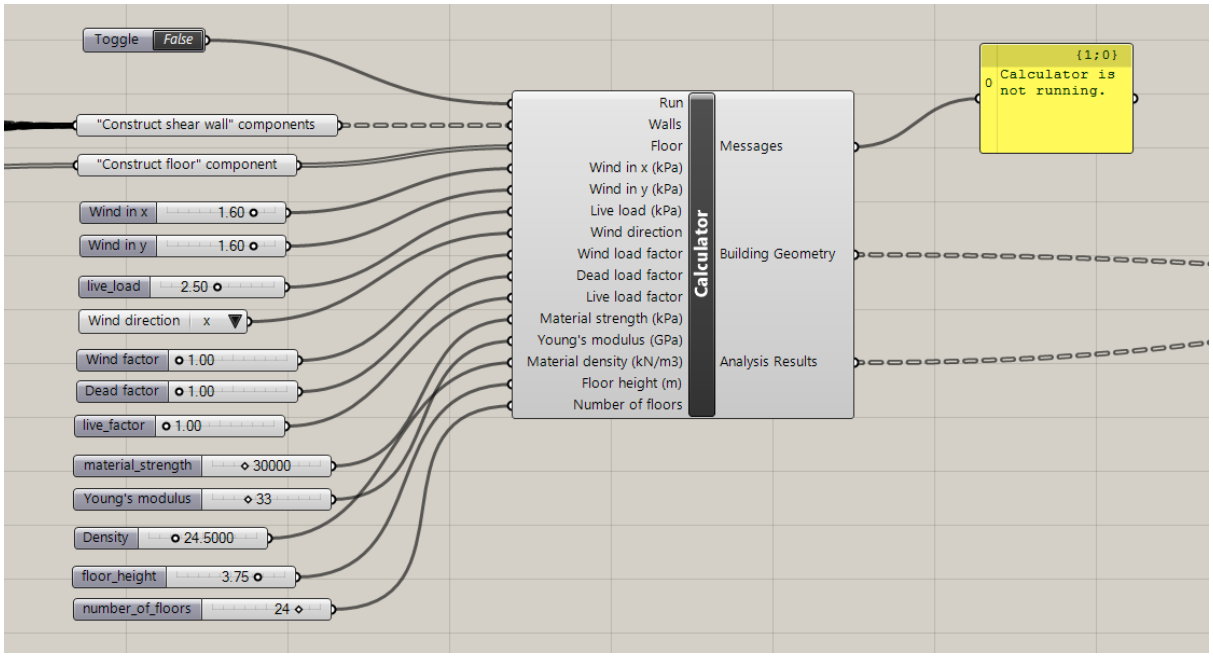


Figure 6.7 Calculator component in case study

The particular values shown as inputs into the calculator in Figure 6.7 represent Analysis 1: wind load in the x-direction, SLS. To run an analysis in ULS, the user simply needs to change the values in the wind load, dead load and live load factor inputs. To run an analysis with wind in the x-direction, the user can change the "wind direction" option from "x" to "y".

All the "Construct shear wall" components are entered into the "Walls" input together. To attach more than one shear wall to this input, the user must press the "Shift" key while entering the wall components. The "Construct floor" component is entered directly into the "Floor" input. Into the "Run" input, the user enters a Boolean toggle which specifies either "True" or "False". In Figure 6.7 the toggle indicates False, which means that the analysis is not running. This is reflected in the "Messages" output. The toggle should always be set to False while the user is changing inputs and only be set to True when the user wants to run an analysis. Otherwise, a new analysis will run whenever the user changes an input value, which will make the model very slow when the user is trying to change multiple inputs.

Figure 6.8 shows the Visualiser component for the case study.

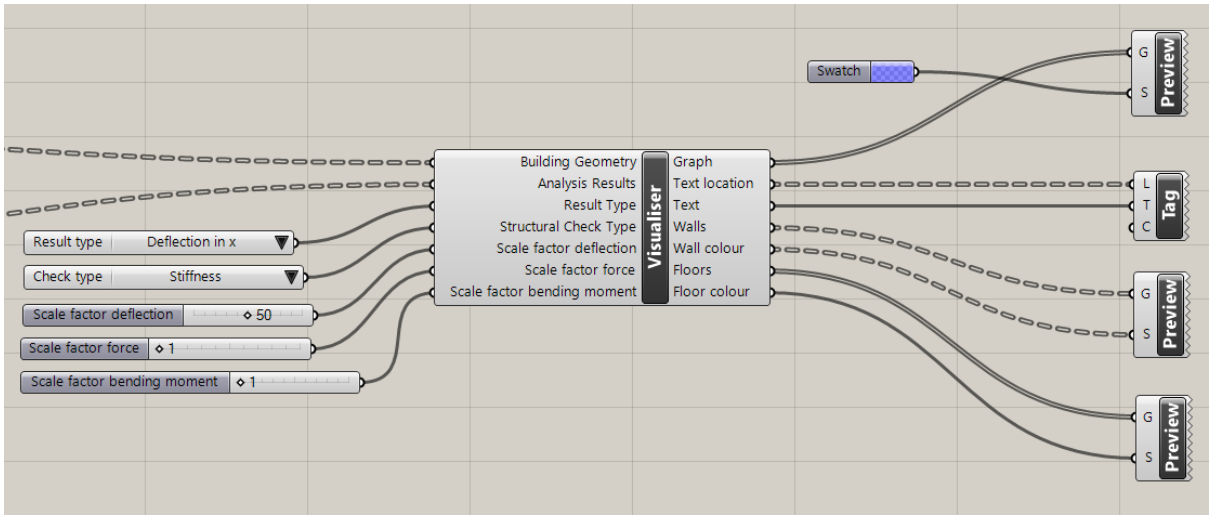


Figure 6.8 Visualiser component in case study

The "Building Geometry" and "Analysis Results" inputs are taken directly from the outputs of the Calculator of the same name. The user can specify what result type they would like to see by toggling through the options of "Result type", and they can specify what structural check they would like to see by toggling through the different options of "Check type". The user can also apply a scale factor to the visualisation of the deflection, force or bending moment. As stated in Section 5.3, the default scale factor for deflection is 1, the default visualisation equivalency for force is 5000 kN per metre, and the default visualisation equivalency for bending moment is 50000 kNm per metre. The inputs for the scale factors are used to enhance these default views.

Figure 6.9 and Figure 6.10 show the geometry of the building for the case study in Rhinoceros, before any analysis has been run. Figure 6.9 shows the perspective view and Figure 6.10 shows the top view.

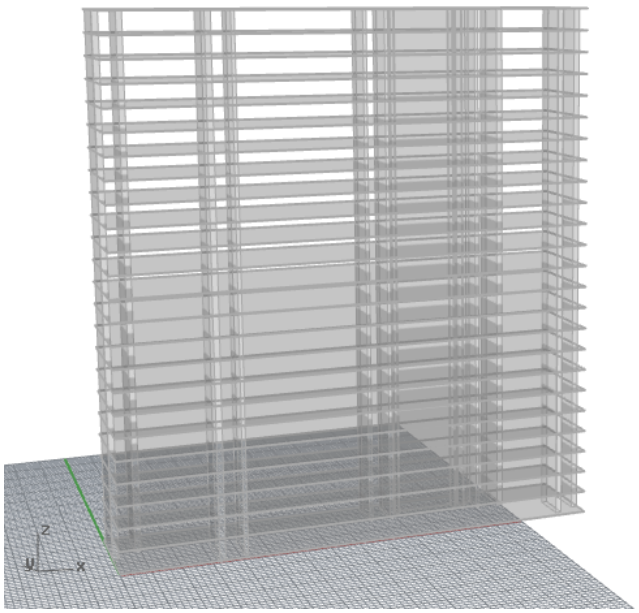


Figure 6.9 Case study building geometry, perspective view

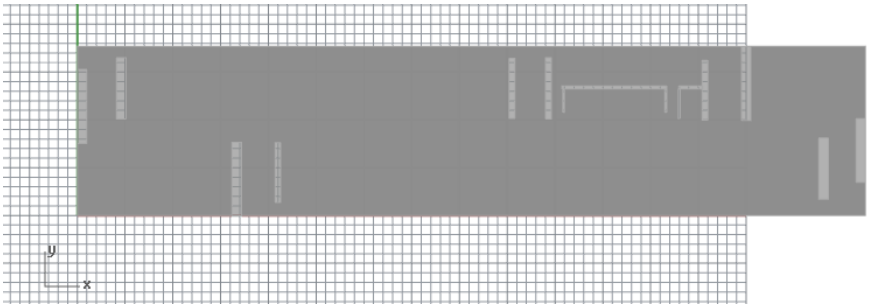


Figure 6.10 Case study building geometry, top view

Now, the analysis results are checked. The four analyses specified above are run, and the results are described in the following section.

6.2 Results of the analyses

Analysis 1: Wind load in the x-direction, SLS

The Calculator is run with all the same properties as specified in Figure 6.7, except the wind direction is changed to "x". To run an analysis, the Boolean toggle is switched to True.

In Analysis 1, the deflection and stability of the walls are checked.

Figure 6.11 shows an image of the deflection in the x-direction from Analysis 1, with a scale factor of ten. The structural integrity check is set to the option "Stiffness". In Figure 6.11, the "Text Tag" output of the Visualiser is turned off to avoid clutter in the image. The floors are also not shown so the walls can be seen more clearly.

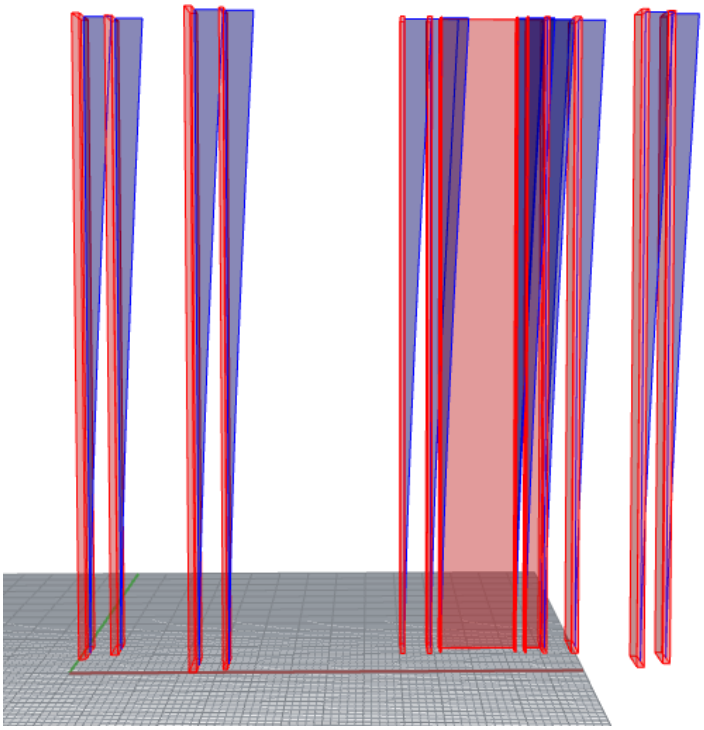


Figure 6.11 Stiffness check, deflection in x-direction for Analysis 1 with scale factor of 10

All walls are red, which means that the stiffness check fails. This indicates that the deflection exceeds the height divided by five hundred. Figure 6.12 shows a zoomed-in view of the top of Walls 1 and 2. In Figure 6.12, the scale factor on deflection has been reduced to five. To show the result for deflection, the "Text Tag" component is turned on.

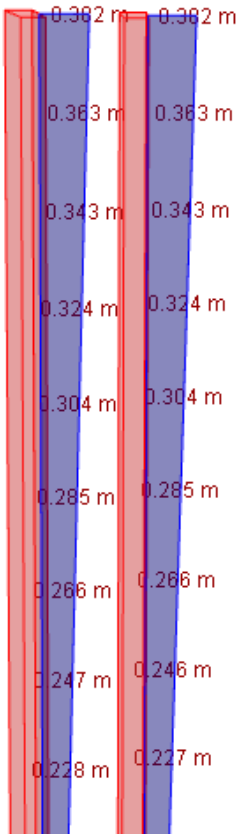


Figure 6.12 Deflection in x direction for Walls 1 and 2, scale factor 5

The maximum deflection on these walls is 0.382 m. For a 90-metre tall building, the maximum deflection at the top should be 0.18 m, and this deflection is exceeded in the analysis by more than a factor of 2.

Figure 6.13 shows the text in the "Messages" output of the Calculator for Walls 1 and 2, which expresses that the deflection in the x-direction is too large. Note that the "Messages" output contains information about all the walls in the analysis, but for brevity only the results of Walls 1 and 2 are shown in Figure 6.13.

```
                                                                    {1;0}
 0 Running...
 1
 2 Wall 1
 3 Foundation stiffness in x: 1965592 kNm/rad
 4 Foundation stiffness in y: 85251184 kNm/rad
 5 Warning! Deflection of Wall 1 in x-direction exceeds height/500
 6
 7 Wall 2
 8 Foundation stiffness in x: 1620702 kNm/rad
 9 Foundation stiffness in y: 48134835 kNm/rad
10 Warning! Deflection of Wall 2 in x-direction exceeds height/500
11
```

Figure 6.13 Messages output for Analysis 1, Walls 1 and 2

Analysis 2: Wind load in the x-direction, ULS

To check the strength of the members, Analysis 2 is now run. In this analysis, the wind load is still applied in the x-direction, but the load factors are changed. The dead load factor is changed to 1.35, and the wind and live load factors are changed to 1.5.

Figure 6.14 shows the shear force in the x-direction on all walls with a visualisation scale factor of 20. The check for "strength (shear)" is turned on to check the shear force.

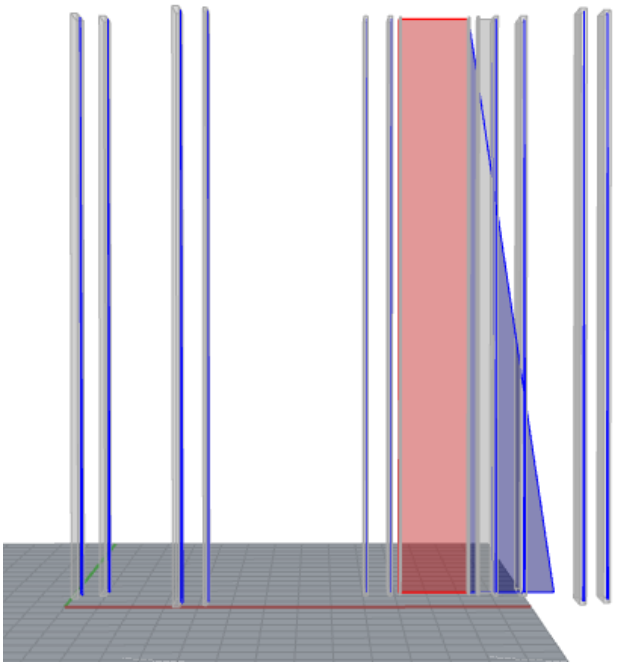


Figure 6.14 Shear force in x-direction for Analysis 2, scale factor 20

Figure 6.15 shows the bending moment in the x-direction of all the walls with a visualisation scale factor of 5. The check for "strength (compressive)" is turned on to check the bending moment.

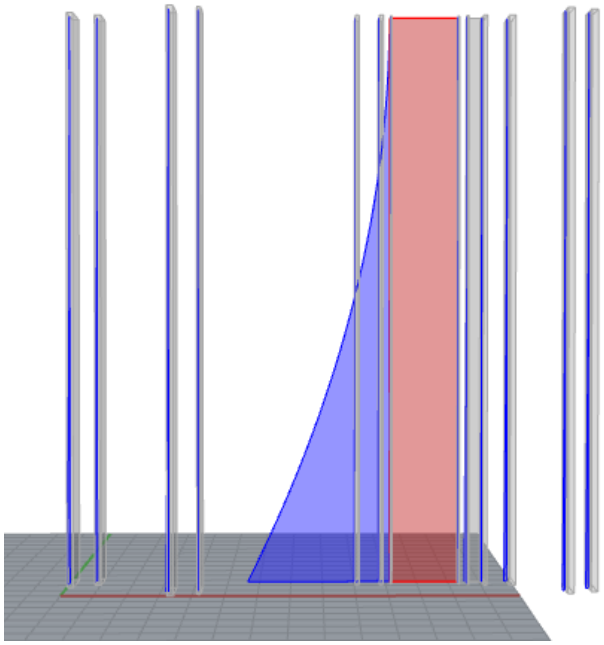


Figure 6.15 Bending moment in x-direction for Analysis 2, scale factor 5

Figure 6.16 shows the normal force at the base of each wall caused by the dead and live load on the walls, with a visualisation scale factor of 3. In this image the check for "strength (compressive)" is still turned on. In this image the "Text Tag" is also turned on so the values of the normal forces are visualised.

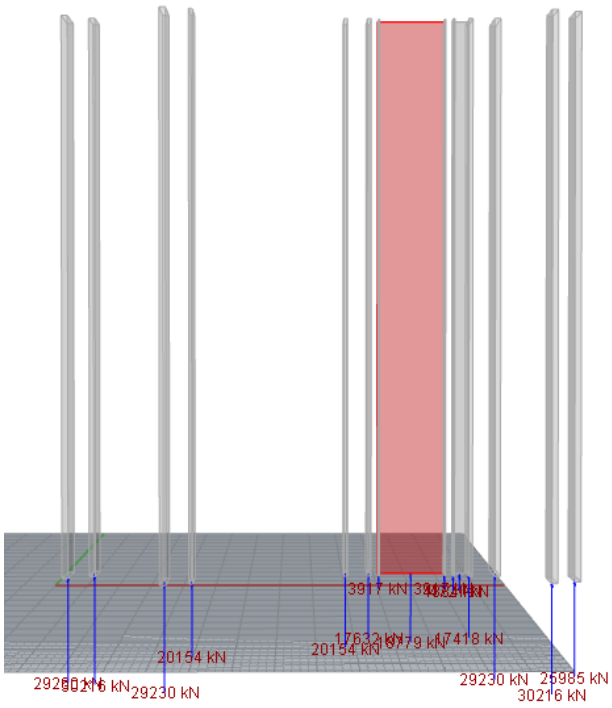


Figure 6.16 Normal force at base of walls for Analysis 2, scale factor 3

This test indicates that both the compressive and shear strength fail on Wall 8 of the building.

To validate these checks, they are calculated for Wall 8. First the compressive stress is checked. The analysis indicates that the bending moment at the base of Wall 8 in the x-direction is -225932 kNm. The normal force at the base of Wall 8 is 18779 kN. The maximum compressive stress on Wall 8 can be calculated as follows.

$$\sigma_x = -\frac{abs(M_x)}{W_x} - \frac{N}{A}$$

$$\sigma_x = -\frac{225932\ kNm}{\frac{(0.3m)(11m)^2}{6}} - \frac{18779\ kN}{(0.3m)(11m)}$$

$$\sigma_x = -43035\ kPa$$

The design compressive strength of C30/37 concrete is $f_{ck}/15$ = 20000 kPa, which indicates that it is not strong enough for this stress. Even high-strength C50/60 concrete has a design compressive strength of 33333 kPa which is too low. To improve the design, the dimensions of the wall need to be increased, or more walls needed to be added to support wind in the x-direction.

Now the shear force at the base of Wall 8 is checked. The analysis indicates that the shear force in the x-direction at the base of Wall 8 is 3376 kN. The resisting shear force is calculated as follows:

$$k = 1 + \sqrt{\frac{200}{d}} = 1 + \sqrt{\frac{200}{7740mm}} = 1.161$$

$$V_{Rd} = \left(0.035 \cdot k^{\frac{3}{2}} \cdot \sqrt{f_{ck}} + 0.15 \cdot \frac{N}{A}\right) \cdot A$$

$$V_{Rd} = \left(0.035 \cdot 1.161^{\frac{3}{2}} \cdot \sqrt{30000\ kPa} + 0.15 \cdot \frac{18779kN}{7.74m^2}\right) \cdot 7.74m^2$$

$$V_{Rd} = 2875.5\ kN$$

The actual shear force of 3376 kN exceeds the resisting shear force of 2875.5 kN, so the check is calculated correctly.

Figure 6.17 shows the results for stability of the building, with no graphs or floors shown. Walls 8 and 11 are red, indicating there is tension in the foundation of these walls. To improve this problem, more vertical load could be applied on these walls, or the stiffness in the x-direction of the system could improve to limit the large bending moments causing tension.

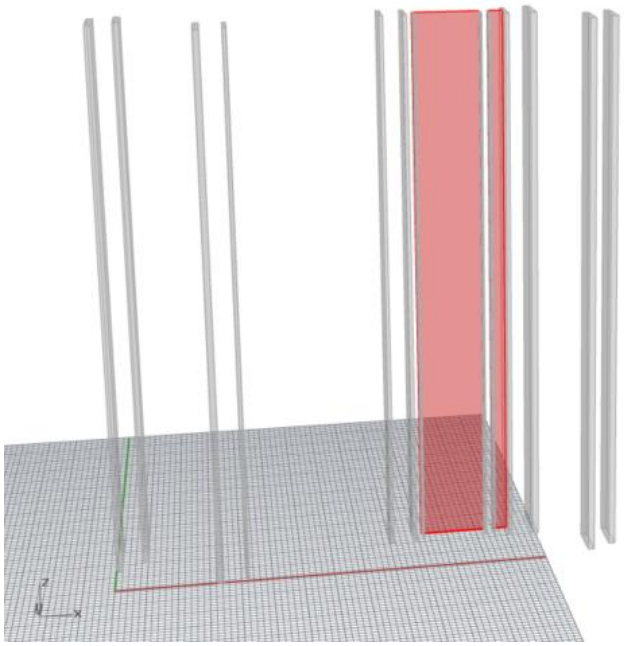


Figure 6.17 Stability check for Analysis 4

To verify the accuracy of results, equilibrium is checked for the system. The total external force from the wind load is calculated as follows:

$$F_{x,total} = 1.5(1.6\ kPa)(17.7m)(90m) = 3823.2\ kN$$

The total external bending moment from the wind load is calculated as follows. Note that a second-order factor is applied to the bending moment. This second-order factor for the x-direction was calculated as part of the analysis to be 1.4874.

$$M_{x,total} = \frac{1.4874(1.5)(1.6\ kPa)(17.7\ m)(90m)^2}{2} = 255898.2\ kNm$$

The total external force and bending moment in the y-direction are both zero.

Table 6.3 shows all shear forces and bending moments at the base of each shear wall in the x-direction and y-direction. At the bottom, the sum of the values on all walls is shown.

Table 6.3 Shear forces and bending moments at base of shear walls for Analysis 2

Wall	Shear force in x (kN)	Shear force in y (kN)	Bending moment in x (kNm)	Bending moment in y (kNm)
1	66	-104	-4385	5669
2	54	-53	-3616	2862
3	66	-62	-4402	3357
4	12	-16	-771	880
5	11	4	-766	-228
6	12	8	-781	-424
7	0	0	-11	-9
8	3376	0	-225933	-1
9	0	0	-11	-18
10	0	1	-14	-41
11	40	0	-2652	0
12	11	21	-766	-1143
13	65	77	-4373	-4176
14	54	55	-3636	-3006
15	56	69	-3774	-3722
Total	**3823**	**0**	**-255891**	**0**

Table 6.3 shows that equilibrium conditions are met for shear force and bending moment. Small differences in value are due to rounding error.

Analysis 3: Wind load in the y-direction, SLS

In Analysis 3, wind load is applied in the y-direction and all load factors are 1.0.

Figure 6.18 shows the deflection in the y-direction with a scale factor of 20. In Figure 6.18 the "stiffness" check is turned on. All walls are red indicating that the deflection in the y-direction exceeds the allowable deflection for the building.

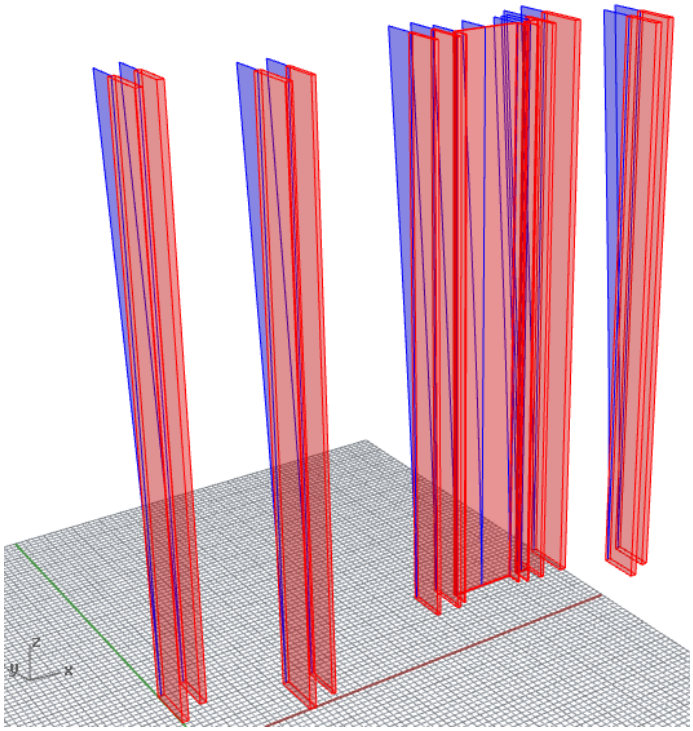


Figure 6.18 Deflection in the y-direction for Analysis 3, scale factor 20

Figure 6.19 shows the value of the deflection of Walls 1 and 2 in the y-direction. This deflection is 0.273 metres for both walls, which exceeds the allowable deflection of 0.18 metres by a factor of 1.5, indicating that the stiffness check is correct.

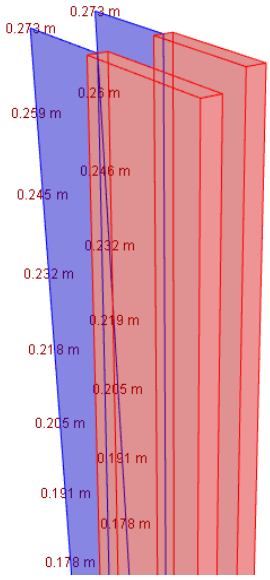


Figure 6.19 Deflection in y-direction on Walls 1 and 2 for Analysis 3

Analysis 4: Wind load in the y-direction, ULS

Analysis 4 is the last analysis to run. In this analysis, wind load is applied in the y-direction and the dead load factor is 1.35 and the live and wind load factors are 1.5.

Figure 6.20 shows the shear force in the y-direction with a visualisation scale factor of 20. In the image, the check for "strength (shear)" is turned on.

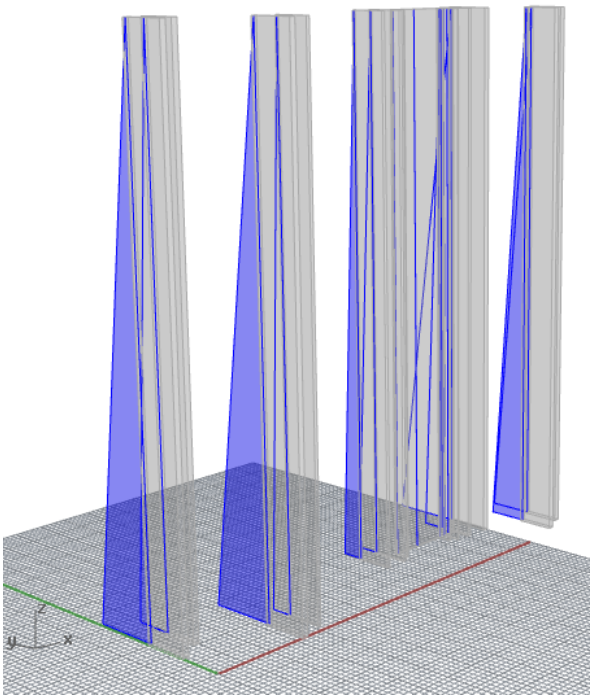


Figure 6.20 Shear force in y-direction for Analysis 4 with shear strength check

Figure 6.20 shows that the shear strength is not exceeded when wind is applied in the y-direction, since all walls are grey.

Figure 6.21 shows the bending moment in y-direction with a visualisation scale factor of 5. In Figure 6.21, the check for "strength (compressive)" is turned on.

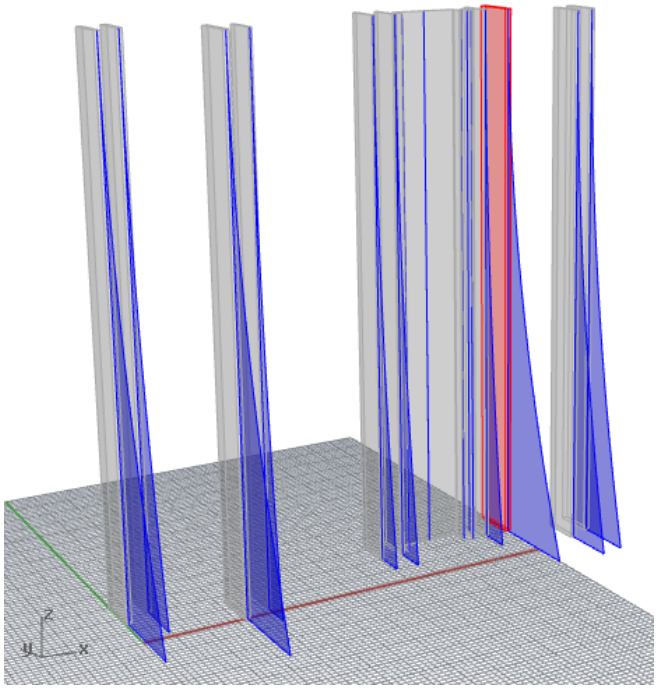


Figure 6.21 Bending moment in y-direction for Analysis 4 with compressive strength check

Figure 6.22 shows the normal force at the base of the walls, with compressive strength check again turned on.

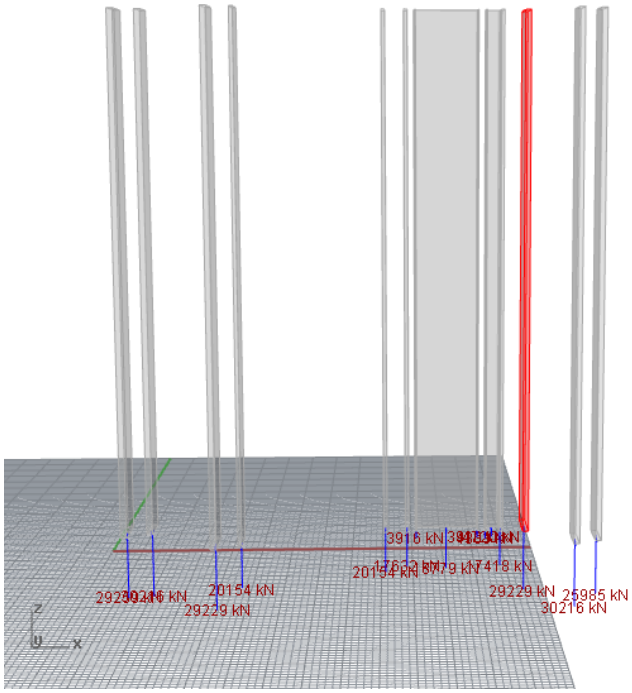


Figure 6.22 Normal force at base of walls for Analysis 4 with compressive strength check

Figure 6.21 and Figure 6.22 indicate that all walls have sufficient compressive strength except Wall 13.

The stability results are checked again for the ULS case. Figure 6.23 shows the stability check on the walls with no graphs shown.

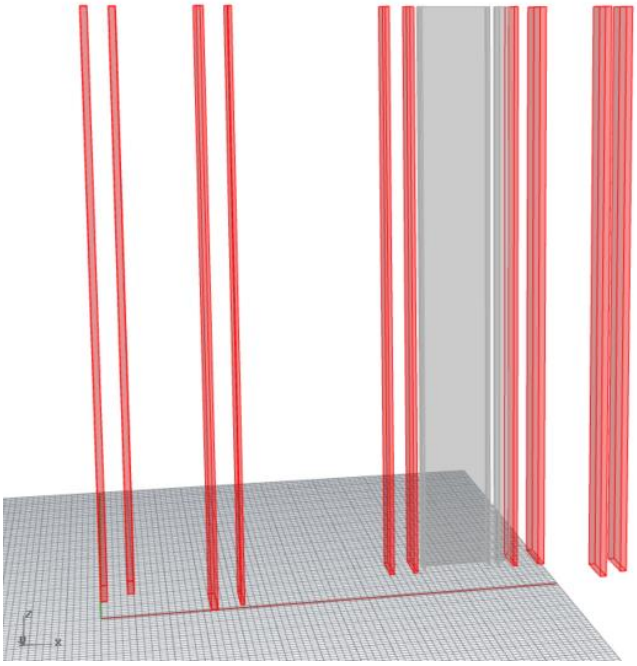


Figure 6.23 Stability check for Analysis 4

Figure 6.23 shows that there is tension in the foundation of all walls except for Walls 8 and 11, opposite to analysis 2. This indicates the bending moments at the base of these walls is too large compared to the normal force applied on the walls.

To verify the results of this analysis, the equilibrium of the system is checked. The total external force in the y-direction is calculated as follows:

$$F_{y,total} = 1.5(1.6\ kPa)(82.5m)(90m) = 17820\ kN$$

The total external bending moment from the wind load is calculated as follows. Again, a second-order factor is applied to the bending moment. This second-order factor for the y-direction was calculated as part of the analysis to be 1.2056.

$$M_{x,total} = \frac{1.2056(1.5)(1.6\ kPa)(82.5\ m)(90m)^2}{2} = 966770.6\ kNm$$

Table 6.4 shows all the forces and bending moments at the base of the shear walls from Analysis 4, and the sum from all the walls.

Table 6.4 Shear forces and bending moments at base of shear walls for Analysis 4

Wall	Shear force in x (kN)	Shear force in y (kN)	Bending moment in x (kNm)	Bending moment in y (kNm)
1	0	2842	-4	-154169
2	0	1604	1	-87048
3	0	2869	-24	-155643
4	0	922	-4	-50012
5	0	940	0	-51015
6	0	998	0	-54165
7	0	18	0	-958
8	-1	2	60	-103
9	0	18	0	-966
10	0	38	0	-2084
11	0	0	1	-24
12	0	956	0	-51842
13	0	2994	2	-162455
14	0	1702	-18	-92354
15	0	1916	-14	-103940
Total	**-1**	**17819**	**0**	**-966778**

Table 6.4 shows that equilibrium conditions are met in Analysis 4.

6.3 Validation of results

Based on this case study, the user interface is judged on the following criteria: ease of model construction, speed of analysis and visualisation of results. The judgements of these criteria are described below:

1. **Ease of model construction**
   The method of model construction is easy to understand. There is a logical progression from the "Floor construction" components, to the "Calculator" component, to the "Visualiser" component. However, this particular case study involved the creation of many different "Construct shear wall" components, which was time-consuming. The model construction would be better if a faster way to construct the stability elements was developed. One way to speed up the model construction could be to provide Floorplan construction components with more specific geometry than the "Construct shear wall" component, for example "Construct core", "Construct U-shape" or "Construct I-shape". This would limit the number of different components needed in the analysis.

   In addition, the tool can only model rectangularly-shaped floors. In the case study, the actual floorplan had to be estimated as a rectangular shape to model it in the tool. Although the shape of the floor does not have consequences on the calculation method, it greatly affects the visualisation of the design, especially when collaborating with an architect or stakeholders. To make the tool more useful for collaboration, the "Construct floor" component should be expanded to allow for more floor shapes than just rectangles.

2. **Speed of analysis**
   Each analysis in the case study took roughly twenty seconds to solve. This analysis time is a bit slow and is the result of the large number of "Construct shear wall" components entered into the analysis. The analysis time could be sped up by reducing the number of start components entered in the analysis; a way to do this could be to create start components with more specific geometry as described above.

3. **Visualisation of results**
   The method of visualisation of results is clear: the user can choose 1) which result they would like to see in terms of deflection, force or bending moment and 2) which structural integrity check they would like to verify in terms of stiffness, strength or stability. In this way, the user can keep track of what results or what check they are looking at. The structural integrity check clearly indicates which walls have problems by turning their colour red, which is easy for the user to understand.

   However, when there are too many different elements in the analysis, the visualisation of results looks somewhat cluttered. Graphs for deflection, shear force, and bending moment for a given wall sometimes overlap adjacent walls, and this make it difficult to identify exactly what wall the graphs belong to. A possible way to solve this problem would be to allow the user to isolate an individual wall and see the graphs for that wall only.

   Another problem is the labelling of the walls in the analysis. The "Messages" output of the Calculator provides the recommended foundation stiffnesses and warnings for Wall 1, Wall 2, Wall 3, etc. The wall number is defined by the order in which the "Construct shear wall" components are entered into the Calculator. However, it is not always clear to the user which wall corresponds to which number. A way to solve this problem would be label each wall with its corresponding number in the Visualiser.

It is interesting that the results from this case study indicate that the building is structurally inadequate, although the case study is based on an existing, constructed building. There are a few explanations for this. Firstly, the stability system in this case study has been simplified compared to the actual stability system of the building, and this may make the building seem weaker than it actually is. For example, Walls 7, 8 and 9 represent a core surrounding four elevators. Since there are many openings in front of the core for the elevator doors, only the back of the core was included in the case study as an approximation of this core, and also to limit the amount of shear walls in the analysis. The extra walls that were left out are shown in red in Figure 6.24.

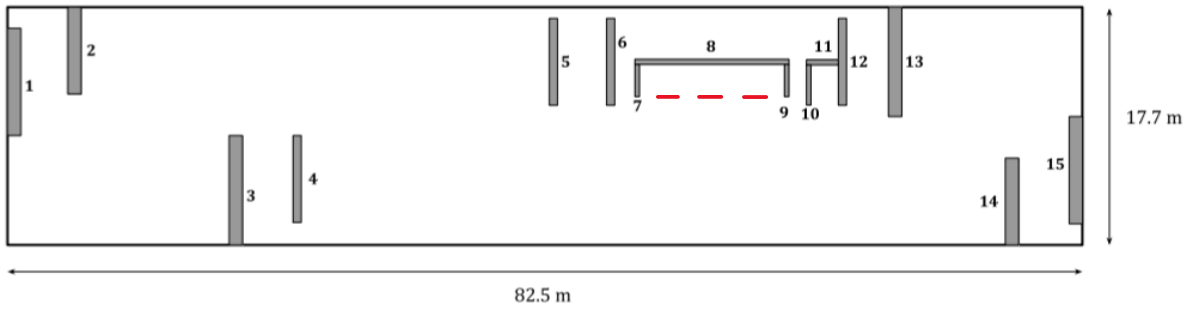


Figure 6.24 Extra shear walls for case study

In addition, the stability elements were modelled prismatically in the case study (because the tool can only model prismatic building plans); however this is not the case for the actual building. In

the actual building plan, Walls 4, 5, 6 and 12 are each one metre thick at the base of the building and become thinner as the building becomes taller, reaching a minimum thickness of 0.4 metres. In the case study, a wall thickness of 0.6 m was used for all of these walls for the entire building height; this is the thickness of the walls at the 10th floor.

Another explanation for the results is likely that the "recommended" foundation stiffness used for the analysis was considerably weaker than the actual foundation stiffness. The foundation stiffness used in the analysis was calculated based on the initial bending moment at the base of the shear walls from wind load only, and does not include second-order effect. When wind was applied in both the x and y-directions, the stiffness check failed, which indicates that the foundation stiffness used for this case study was clearly not strong enough. To improve the tool, the foundation stiffness used in the analysis should be increased to a more realistic value.

7 Discussion

7.1 Reflection on the objectives

The main objective of the project is stated as follows:

> *Develop a tool, based on StructuralComponents 5, that provides early-stage structural validation for flexible topologies of concrete mid-rise buildings made of shear walls, cores and floors.*

The main objective was split into four sub-objectives. A discussion on how each of these objectives was met in the project is provided below.

1. Investigate the requirements of the tool to allow for the conceptual design of a mid-rise building

To achieve this objective, the conceptual design process and the current state-of-the-art in conceptual design tools was studied. After this initial investigation, a general framework of the conceptual design process was developed. From this framework, key qualities of the conceptual design process were identified as defined as objectives for the development of the tool.

2. Develop a flexible calculation method to determine the forces and deflections to a sufficient degree of accuracy for varying arrangements of shear walls, cores and floors

To achieve this objective, a single calculation method was developed to analyse various configurations of shear walls and/or cores on a floorplan. This calculation method had to determine the deflection, force and bending moment on each shear wall/core to a sufficient degree of accuracy. A method of analysis using differential equations instead of FEM was chosen because differential equations allow 1) more insight into the structural behaviour, 2) a simpler form of model construction and 3) quicker analysis than FEM.

Two analysis methods were initially considered: 1) the Super Element Method (used in StructuralComponents 5) and 2) Assuming the floors were infinitely rigid. Method 1) was rejected because it was inflexible and difficult to adapt to custom configurations of shear walls and cores; Method 2) was used for further development. To test Method 2), three different "test" configurations were proposed, and a calculation method was developed for each test. Results from each test were compared against FEM results and analysed. The best of the three test configurations was chosen based on the criteria of accuracy and flexibility, and further developed. The fully-developed final configuration was then expanded into an automated form, wherein the number and position of stability elements could vary.

3. Develop an intuitive user interface for the tool

To achieve this objective, a user interface for the tool was developed. Like StructuralComponents 5, the tool was implemented as a group of components in Grasshopper. The calculation method developed in the previous objective was transferred into Grasshopper. The key qualities of the conceptual design process identified as part of the first objective were used as drivers for the development of the user interface.

4. Examine the applicability of the tool to a real building design

To achieve the last objective, a case study was performed on a pre-existing building, using the tool. A model of the building was constructed in the tool and was analysed in both SLS and ULS with the wind applied in two different directions. The user interface of the tool was judged based on the criteria 1) ease of model construction, 2) speed of analysis and 3) visualisation of results.

Through these four sub-objectives, the main objective of the project was successfully achieved. A new tool was developed, based on StructuralComponents 5, to analyse and validate custom configurations of shear walls and cores (and differently-shaped stability elements) on a floorplan. Through a case study, it was shown that the tool can be applied to complex configurations of stability elements.

7.2 Limitations

The limitations of the project are split into two different categories: limitations of the calculation method and limitations of the user interface. These are described below.

Limitations of the calculation method

The limitations noted in the calculation method are listed as follows:

1. Torsion

As shown in Section 4.4, the rigid floor calculation method used for the tool cannot accurately predict the torsion around stability elements. In the test configuration of this section, the torsion stresses were demonstrated to be very small in relation to bending stresses, and it was concluded that torsion could be excluded from the analysis on that basis.

However, for some building configurations, torsion around stability elements can become more significant. The torsion tends to become more significant if the building is shorter and stockier, or if the floorplan is highly asymmetric. For this reason, it would be a good idea to study the torsion around stability elements in more detail, and determine exactly what the limiting cases are for omitting torsion around the stability elements.

2. Out-of-plane floor effects

In the calculation method used for the tool, out-of-plane floor effects were ignored, based on the assumption that the tool is used to design a building with pre-cast floors (hinged connections between the floors and walls). However, there is also potential for a concrete building to be constructed with cast-in-place floors, in which the out-of-plane floor effects are significant for the analysis.

The GSA models used to verify the calculation method in Chapter 4 included out-of-plane floor effects. By comparing the results of the GSA models with the results of the calculation method, it could be seen that out-of-plane floor effects have a significant effect on the bending moment equilibrium of the building. The out-of-plane floor effects become more significant as the floor becomes thicker and deeper. For this reason, it would be interesting to further study the out-of-plane floor effects and incorporate them into the tool.

To avoid improper use of the current tool, it would be a good idea to provide the user with a warning that the tool is only applicable for buildings with minimal out-of-plane floor effects.

3. Non-prismatic buildings

The calculation method was only implemented for prismatic buildings. This limits the use of the tool because many mid-rise buildings are non-prismatic.

A method that is known to effectively analyse non-prismatic buildings is the Super Element Method, which was used in the previous version of StructuralComponents. This method was considered to be used in this version of the tool, but was rejected early because of its apparent lack of adaptability to custom configurations of stability elements. However, the benefit that was lost by not using the Super Element Method was the ability to model non-prismatic buildings.

For this reason, it may be a good idea to perform a more in-depth analysis of the Super Element Method and its applicability to custom configurations of stability elements. Alternatively, the limitations of using a rigid floor method to analyse non-prismatic buildings could be studied; it is possible this calculation method could be adapted for non-prismatic buildings.

4. Foundation stiffness

The foundation stiffness was not studied in extensive detail in this project. As an approximation for the foundation stiffness, the "recommended" stiffness at the base of each shear wall was calculated based on a deflection requirement of height/1000 from wind load only, and this recommended stiffness was used in further parts of the analysis, such as the calculation of the second-order effect.

However, as shown in the case study in Chapter 6, this "recommended" stiffness clearly is not stiff enough to maintain a total deflection requirement of height/500. This is, in part, caused by the fact that the spring stiffness was calculated before the second-order effect was applied to the deflection/bending moment (this was done because the spring stiffness was required to calculate the second-order factor). Additionally, the actual foundation stiffness that an engineer uses for their building will likely be different than this recommended stiffness due to various aspects of foundation design, such as the strength of the foundation piles or the stiffness of the ground itself. An in-depth analysis of building foundation design should be performed and a more accurate method to determine the recommended foundation stiffness needs to be developed for the tool.

Furthermore, it may be a good idea to simply allow the designer to add the foundation stiffness as an input into the analysis, in case the designer already knows the stiffness of their foundation.

5. Expansion joints

Expansion joints are frequently used in large concrete buildings. Expansion joints are needed to relieve stress in the floor which can be caused by a number of reasons, such as thermal expansion/contraction of the concrete, or constrained deformation of a floor between two very stiff stability elements. As a result of restraints on the scope of the project, expansion joints were not included in the tool. However, to realistically analyse a mid-rise concrete building, expansion joints should be included in the analysis in a future version of the tool.

Limitations of the user interface

The limitations of the user interface are listed below:

1. Model construction

The user uses two different components to build a floorplan: the "Construct floor" component and the "Construct shear wall" component. Only one "Construct floor" component is needed for a model; however, the user must add a new "Construct shear wall" component whenever they would like to add a new shear wall to their floorplan. Additionally, if the user wants to create a different type of stability element, such as a core or I-shape, they must use multiple "Construct shear wall" components to construct this stability element.

For a floorplan with many stability elements, the model construction process can be somewhat tedious and time-consuming. The model construction could be improved if more floorplan construction components were available, such as a "Construct core" or "Construct I-shape" component. This would also speed up the analysis time because fewer separate components would need to be added to the analysis.

2. Visualisation of building design

There is one significant limitation on the visualisation of the building design: the "Construct floor" component can only model a rectangular floor. To analyse a building with a non-rectangular cross-section, the designer needs to approximate their floorplan as a rectangular section based on how the wind load is applied on it. This was demonstrated in the case study in Chapter 6.

This limits the use of the tool in terms of architectural design and collaboration with stakeholders, because the actual building shape cannot be fully visualised. It is possible to implement different floor shapes into the existing calculation method; the tool simply needs to calculate the equivalent wind load in x and y-direction based on the non-standard floorplan.

3. Visualisation of results

As demonstrated by the case study in Chapter 6, when there are too many stability elements in the analysis, the visualisation of graphs for deflection, shear force and bending moment can appear cluttered, and graphs sometimes overlap adjacent walls. A way to solve this problem could be to allow the user to isolate an individual stability element and view results for this element only.

8 Conclusions

The main objective of the project is restated:

> *Develop a tool, based on StructuralComponents 5, that provides early-stage structural validation for flexible topologies of concrete mid-rise buildings made of shear walls, cores and floors.*

This main objective was split into four sub-objectives. Conclusions relating to each sub-objective are provided below.

1. Investigate the requirements of the tool to allow for the conceptual design of a mid-rise building

To achieve this objective, the conceptual design process was studied and a high-level framework of the conceptual design phase of a building design was developed. It was found that the conceptual design phase is a divergent and highly iterative process that involves developing many options in different levels of detail. A good conceptual design must be justifiable, and the designer must have confidence in their design. Based on this description, it was concluded that the following characteristics are desirable for a conceptual design tool:

- Quick and easy generation of design alternatives
- Easy changes to the design
- Parallel investigation/comparison of alternatives
- Constant feedback on the design criteria
- Visualisation of design and analysis results

2. Develop a flexible calculation method to determine the forces and deflections to a sufficient degree of accuracy for varying arrangements of shear walls, cores and floors

To achieve this objective, a flexible calculation method was developed to calculate the deflection, shear force and bending moment for varying configurations of stability elements connected by infinitely rigid floors. To verify the validity of this "rigid-floor method", three test configurations were developed using this method and compared against finite element models with flexible floors. Both rigid-floor and flexible-floor models were analysed and discussed. The conclusions found in this phase of the project are stated as follows.

- Torsion around stability elements cannot be predicted when the floors are assumed to be infinitely rigid. This was demonstrated by the analysis of a floor configuration consisting of a shear wall and core in Section 4.4. In this analysis, both the value and location of maximum torsion around the core as calculated by the rigid-floor method were inaccurate compared to the flexible-floor model. This inaccuracy occurred because rigid floors restrain the rotation of the core much more than flexible floors do, leading to a large difference in the torsional behaviour.

- Torsion stresses are generally small compared to bending stresses. This was demonstrated by the comparison of torsion stresses and bending stresses for the building configuration described in Section 4.4. For this building configuration, the maximum bending stress in the core was fifty-five times as large as the torsion stress

around the core. As shown by further tests in Section 4.4, torsion stress became more significant when the building became highly non-symmetric or if the building became short and stocky, but even in these cases the torsion stress was less than the bending stress.

- Out-of-plane floor effects have a significant effect on the bending moment of a building. This was demonstrated in Section 4.5 by comparing the finite element model, which contained out-of-plane floor effects, to the rigid-floor model, which contained no out-of-plane floor effects. An example of a situation where the out-of-plane effects had a significant effect on the bending moment is shown in Table 8.1 (from Table 4.9 of Section 4.5).

  Table 8.1 Bending moment in x-direction for Test 3, wind applied in y

Wall	Rigid-floor model	Finite element model	% difference
Wall 1	-27.7	-36.6	-24.3%
Wall 2	31.2	436.6	-92.9%
Wall 3	-3.5	-2.3	52.2%

  The inaccuracy shown in Table 8.1 occurred because the out-of-plane force in the floors of the finite element model created an extra normal force on the stability elements which did not exist in the rigid-floor model. The bending moment equilibrium for the finite element model had to include the contribution of this extra normal force. Therefore, the bending moments on the stability elements in the finite element model were different than those in the rigid-floor model, because they had to compensate for this extra normal force.

- The thinner the floors are, the more accurate the rigid-floor method becomes. This was demonstrated in Section 4.5; when the floor thickness was reduced from 0.26 metres to 0.1 metres, the results of the rigid-floor method became considerably more accurate when compared to the finite element results. The reason for this is related to the out-of-plane floor effects, as discussed in the previous point. The thinner the floor becomes, the less significant the out-of-plane floor effects become, so the assumption of no out-of-plane floor effects in the rigid-floor method becomes closer to reality.

- The rigid-floor method provides a sufficiently accurate prediction (within 10%) of the *governing results* for deflection, shear force and bending moment on stability elements for mid-rise concrete buildings with minimal out-of-plane floor effects. Minimal out-of-plane floor effects occur when the floors are considered "hinged" to the stability elements; an example of this is a building with pre-cast concrete-steel deck floors. "Governing results" refers to the maximum results for deflection, shear force and bending moment on each stability element; these results are governing in design because they define the necessary dimensions of each stability element. This conclusion was demonstrated in Section 4.7 by comparing the "governing results" of the rigid-floor model against a finite element model with 0.1-metre floor thickness. In this comparison, the results differed by a maximum of 3%.

Based on the results of this section, it is concluded that the developed calculation method is sufficient for the design of mid-rise concrete building with pre-cast floors, supported laterally by shear walls and/or cores. It is however not adequate for buildings with cast-in-place floors, especially if the floors are thick. To adapt the method for cast-in-place floors, out-of-plane floor effects need to be included in the analysis.

3. Develop an intuitive user interface for the tool

To achieve this objective, the calculation method of the previous phase was incorporated into a tool wherein a conceptual building design can be visualised and validated on the basis of stiffness, strength, and stability. The tool was developed as a group of components in Grasshopper and the calculation method was written in Python via Grasshopper's "Python Script" component.

At the end of this phase, it is concluded that StructuralComponents 6 has been successfully developed. The tool allows the user to create a conceptual design of a prismatic, rectangular building with a customisable amount and arrangement of shear walls. For any given design, the tool calculates the deflection, shear force and bending moment along each shear wall in the x and y-direction, given applied dead and live loads. The tool also calculates the normal force at the base of each shear wall. The tool performs checks for stiffness, strength and stability of the building, and provides warnings if any of these checks fail.

4. Examine the applicability of the tool to a real building design

To achieve this final objective, a case study was performed on the tool. The tool was evaluated on the ease of model construction, speed of analysis and visualisation of results. It was concluded from the case study that the tool can be effectively used for complex configurations of stability elements on a floorplan. However, for a building with many different stability elements, model construction can be time-consuming and graphs for deflection, shear force and bending moment can appear cluttered. Additionally, the visualisation of the building design is limited because only rectangular floors can be modelled.

It is concluded that the main objective was achieved. A calculation method was developed that can effectively determine the limiting behaviour for design of varying horizontal configurations of stability elements in mid-rise concrete buildings with minimal out-of-plane floor effects. This calculation method was incorporated into a simple tool wherein a designer can create a custom conceptual building design and analyse its behaviour on the basis of strength, stiffness and stability.

StructuralComponents 6 provides a significant step forward in the StructuralComponents project by addressing the previous limitations in analysing horizontal variations in building floorplans. The research performed during this project provides a solid groundwork for new developments of StructuralComponents that address the design of buildings with flexible arrangements of stability elements on the horizontal plane.

9 Recommendations

Based on the conclusions of the project, several recommendations are suggested for further development of StructuralComponents 6.

Recommendations to improve the method of structural analysis are listed below:

- **Torsion:** The effects of torsion on stability elements should be further studied. It should be determined if there are limiting cases where it is necessary to include torsional stresses in the analysis.

- **Out-of-plane floor effects:** If the user would like to design a building with rigid floor-to-wall connections, the tool could provide the user with a check to determine if the out-of-plane floor effects are small enough that the tool is still applicable to their design. Additionally as a future extension to the tool, out-of-plane floor effects could be included in the analysis so the tool can be applied to more building types.

- **Non-prismatic buildings:** To apply the calculation method to more flexible building designs, it should be extended to non-prismatic buildings. An investigation should be performed to determine if this is possible when assuming that floors are infinitely rigid.

- **Foundation stiffness:** An improved method to calculate the recommended stiffness at the base of the foundation should be developed.

- **Expansion joints:** The calculation method should be modified to include expansion joints in the floors, because expansion joints are very common in concrete buildings. Additionally, the tool could inform the user whether or not expansion joints are needed. For example, if the mass of a concrete floor exceeds a certain limit, thermal effects can cause cracking of the floor; the tool could recommend the need for an expansion joint based on thermal expansion of concrete.

Recommendations to improve the user interface are listed as follows:

- **Model construction:** Currently, the only type of stability element that can be modelled in StructuralComponents 6 is a shear wall. The user can construct more complex stability elements from combinations of shear walls, but this is a slow process. To improve the speed of model construction, the library of available stability elements should be expanded to other types of stability elements, such as cores or I-shapes.

- **Floor shape:** The "Construct Floor" component should be modified to model non-rectangular floorplans.

- **Visualisation of results:** The visualisation of results could be improved. Improvements could include: isolating graphs for individual stability elements, labelling stability elements by number, or providing an output "report" for the model.

10 References

Altair. (2019a). *What is Altair Inspire?* Retrieved from 20 December 2018 https://solidthinking.com/product/inspire/

Altair. (2019b). *The Altair Inspire Platform.* Retrieved 20 December 2018 from https://solidthinking.com/inspireplatform/

Autodesk. (2016). *Dynamo*. Retrieved 22 August 2019 from https://dynamobim.org/

Bangal, J. (2018). *Top 5 Things You Need to Know about the Altair Inspire Platform.* Retrieved 20 December 2018 from http://www.goengineer.com/2018/09/06/altair-inspire/blog/

Benjamin, O. (2019). *Sympy – limitations in the computation ability of dsolve function?* [Response to post]. Retrieved 22 May 2019 from https://stackoverflow.com/questions/56133257/sympy-limitations-in-the-computation-ability-of-dsolve-function

Bentley Systems Incorporated. (2019). *GenerativeComponents.* Retrieved 22 August 2019 from https://www.bentley.com/en/products/product-line/modeling-and-visualization-software/generativecomponents

BLOCK Research Group. (2012). *eQUILIBRIUM: an interactive, graphic statics-based learning platform for structural design.* Retrieved 22 August 2019 from http://block.arch.ethz.ch/equilibrium/

Bovenberg, A. C. (2015). *StructuralComponents 4: Conceptual building models with structural design justification* (Master's thesis). Delft, The Netherlands: TU Delft.

Braha, D. & Maimon, O. (1997). The design process: properties, paradigms, and structure. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 27*(2), 146-166. doi: 10.1109/3468.554679

Breider, J. (2008). *StructuralComponents: development of parametric associative design tools for the structural design of high-rise buildings* (Master's thesis). Delft, The Netherlands: TU Delft.

BusinessWire. (2016). *PTC Announces Creo 4.0 for Smarter Design.* Retrieved 24 October 2018 from https://www.businesswire.com/news/home/20161115006063/en/PTC-Announces-Creo-4.0-Smarter-Design

Coenders, J. (2011). *NetworkedDesign: next generation infrastructure for computational design* (PhD dissertation TU Delft). Delft, the Netherlands: VSSD

Coenders, J. & Wagemans, L. (2006). *Structural Design Tools: The next step in modelling for structural design.* doi: 10.2749/222137806796184860

CURT. (2004). *Collaboration, integrated information, and the project lifecycle in building design, construction and operation (CURT Whitepaper 1202).* Cincinnati, OH: The Construction Users Roundtable.

Dassault Systèmes. (2012). *Solidworks Mechanical Conceptual* [datasheet]. Retrieved 24 October 2018 from https://www.solidworks.com/sw/docs/SOLIDWORKS-Mechanical-Conceptual-datasheet.pdf

Dassault Systèmes. (2014). *Eight Things You Need to Know About SolidWorks Mechanical Conceptual.* Retrieved 24 October 2018 from https://blogs.solidworks.com/solidworksblog/2014/01/eight-things-you-need-to-know-about-solidworks-mechanical-conceptual.html

Davidson, S. (2019). *Grasshopper: Algorithmic Modeling for Rhino.* Retrieved 22 August 2019 from https://www.grasshopper3d.com/

Digital Structures. (2018). *GH Python Remote.* Retrieved 22 May 2019 from https://www.food4rhino.com/app/gh-python-remote

Dycore. (2019). *De technische productinformatie van onze massieve plaatvloer.* Retrieved 28 June 2019 from http://dycore.wpengine.com/producten/massieve-plaatvloer/technische-productinformatie-massieve-plaatvloer

Gigerenzer, G., & Selten, R. (2001). *Bounded Rationality: The Adaptive Toolbox.* United States of America: Massachusetts Institute of Technology.

Ham, P. & Terwel, K. C. (2017). *Structural calculations of High Rise Structures.* Delft, The Netherlands: TU Delft.

Hohrath, B. (2018). *StructuralComponents 5: Super element based tool for early design collaboration applied to mid-rise buildings* (Master's thesis). Delft, The Netherlands: TU Delft.

Institution of Structural Engineers. (2011). *Structural Design - The Engineer's Role.* Institution of Structural Engineers. Retrieved 13 December 2018 from https://app.knovel.com/hotlink/toc/id:kpSDTER001/structural-design-engineers/structural-design-engineers

Khandani, S. (2005). *Engineering Design Process: Education Transfer Plan.* Retrieved 13 December 2018 http://www.iisme.org/ETPExemplary.cfm

Macleamy, P. (2010). *The Future of the Building Industry (3/5): The Effort Curve* [Video file]. Retrieved 28 August 2018 from https://www.youtube.com/watch?v=9bUlBYc_Gl4

Maplesoft. (2019). *The Essential Tool for Mathematics*. Retrieved 22 August 2019 from https://www.maplesoft.com/products/Maple/

Matrix Software. (2019). *MatrixFrame: Software for structural engineers.* Retrieved 22 August 2019 from https://www.matrix-software.com/structural-engineers/matrix-frame

Mueller, V. (2009). Conceptual Design Tools: Establishing a framework for specification of concept design tools. *Digitizing Architecture: Formalization and Content [4th International Conference Proceedings of the Arab Society for Computer Aided Architectural Design (ASCAAD 2009)/ISBN 978-99901-06-77-0], Manama (Kingdom of Bahrain), 11-12 May 2009, 103-120.* doi: 10.13140/RG.2.1.2374.8643

National Institute of Building Sciences. (2014). *Frequently Asked Questions About the National BIM Standard-United States.* Retrieved 13 December 2018 from https://web.archive.org/web/20141016190503/http://www.nationalbimstandard.org/faq.php#faq1

NEN-EN 1991-1-1+C1. (2011). *Eurocode 1: Actions on structures – Part 1-1: General actions – Densities, self-weight, imposed loads for buildings.* Koninklijk Nederlands Normalisatie-instituut.

NEN-EN 1991-1-4+A1+C2. (2011). *Eurocode 1: Actions on structures – Part 1-4 : General actions – Wind actions.* Koninklijk Nederlands Normalisatie-instituut.

NEN-EN 1992-1-1. (2005). *Eurocode 2: Design of concrete structures – Part 1-1: General rules and rules for buildings.* Koninklijk Nederlands Normalisatie-instituut.

NEN-EN 1992-1-1+C2/NB. (2016). *National Annex to NEN-EN 1992-1-1+C2 Eurocode 2: Design of concrete structures – Part 1-1: General rules and rules for buildings.* Koninklijk Nederlands Normalisatie-instituut.

Oasys. (2019). *Structural Analysis and Design Software – GSA Suite.* Retrieved 22 August 2019 from https://www.oasys-software.com/products/structural/gsa-suite/

Pal, S. (2014). Conceptual Design Software Tools. *Cadalyst.* Retrieved 24 October 2018 from http://www.cadalyst.com/early-design/conceptual-design/conceptual-design-software-tools-19144

Piacentino, G. (2014). *list_to_tree.py* [Python function]. Retrieved 30 May 2019 from https://gist.github.com/piac/ef91ac83cb5ee92a1294

Przemieniecki, J. (1968). *Theory of Matrix Structural Analysis.* United States of America: McGraw-Hill, Inc.

PTC. (2018a). *PTC Announces Creo 5.0, the Latest Version of Its Award-Winning CAD Solution.* Retrieved 24 October 2018 from https://www.ptc.com/en/news/2018/ptc-announces-creo-5

PTC. (2018b). *Creo 5.0: What's New* [Video file]. Retrieved 24 October 2018 from https://www.ptc.com/en/products/cad/creo/whats-new

Python Software Foundation. (2019). *Python.* Retrieved 22 August 2019 from https://www.python.org/

Reyes, A. (2017). *Announcing the Next Generation Design Platform: NX 12.* Retrieved 24 October 2018 from https://community.plm.automation.siemens.com/t5/NX-Design-Blog/Announcing-the-Next-Generation-Design-Platform-NX-12/ba-p/440033

RIBA. (2013). *RIBA Plan of Work 2013.* Retrieved 14 June 2019 from https://www.ribaplanofwork.com/Default.aspx

Robert McNeel & Associates. (2019). *Rhinoceros*. Retrieved 22 August 2019 from https://www.rhino3d.com/

Rolvink, A. (2010). *StructuralComponents: A parametric and associative toolbox for conceptual design of tall building structures* (Master's thesis). Delft, The Netherlands, TU Delft.

Rolvink, A., Mueller, C., & Coenders, J. (2014). State on the Art of Computational Tools for Conceptual Structural Design. *International Association for Shell and Spatial Structures (IASS) Symposium 2014, Brasilia, Brazil.*

Shearer, M. (2010). *Analyzing and Creating Forms: Rapid Generation of Graphic Statics Solutions through RhinoScript* (Master's thesis). Cambridge, Massachusetts: Massachusetts Institute of Technology.

Siemens. (2018). *NX for Design streamlines and accelerates the product development process.* Retrieved 24 October 2018 from https://www.plm.automation.siemens.com/global/en/products/nx/nx-for-design.html

Simone, A. (2011). *An Introduction to the Analysis of Slender Structures.* Delft, The Netherlands: TU Delft.

Sriram, D., Stephanopoulos, G., Logcher, R., Gossard, D., Groleau, N., Serrano, D., & Navinchandra, D. (1989). Knowledge-Based System Applications in Engineering Design: Research at MIT. *AI Magazine, 10*(3), 79-96. doi: 10.1609/aimag.v10i3.758

Steenbergen, R. (2007). *Super Elements in High-Rise Buildings under Stochastic Wind Load* (PhD dissertation, TU Delft). Delft, the Netherlands: Uitgeverij Eburon.

Sympy Development Team. (2018). *SymPy.* Retrieved 22 August 2019 from https://www.sympy.org/en/index.html

TU Delft. (2009). *EWI, Mekelweg 4, Delft, 10e verdieping.* (36-01-10-S010-P01). [Architectural drawing]

TU Delft. (2016). *Concrete Building Structures: Reader CIE3340/CIE4281.* Delft, The Netherlands: TU Delft.

van de Weerd, B. M. (2013). *StructuralComponents: A client-server software architecture for FEM-based structural design exploration* (Master's thesis). Delft, The Netherlands: TU Delft.

von Buelow, P. (2008). Suitability of genetic based exploration in the creative design process. *Digital Creativity, 19(1),* 51-61. doi: 10.1080/14626260701847522

von Buelow, P. (2011). Genetically Enhanced Parametric Design for Performance Optimization. *Proceedings of the 7th Seminar of the IASS Structural Morphology Group International*

White Lioness technologies. (2019). *Packhunt.io*. Retrieved 22 August 2019 from https://www.packhunt.io/

List of Figures

List of Tables

Appendix A – Super Element Method

Structural analysis is performed in StructuralComponents 5 using the super element method as devised by Steenbergen in his PhD dissertation *Super Elements in High-Rise Buildings under Stochastic Wind Load* (2007). This section will discuss the origins of super elements and Steenbergen's formulation of the super element method.

In traditional finite element analysis, a structure or object is subdivided into numerous small elements and a stiffness matrix K is constructed for the entire system of elements. The stiffness matrix relates the forces (f) to the displacements (d) in every single element according to Hooke's law: f = Kd. However, if one would only like to determine the displacements at the locations where external forces are applied, such a detailed stiffness matrix is not necessary for analysis. To address this issue, the concept of stiffness matrix "condensation" was developed. In this method, all of the finite elements that correspond to a zero external force are "condensed" together, so that a stiffness matrix can be constructed for the system that only relates the known forces to their corresponding unknown displacements (Przemieniecki, 1968).

In 2007, Steenbergen developed a new type of super element for the structural analysis of buildings. In Steenbergen's version, the super elements are formulated differently than in the original approach. Instead of deriving the stiffness matrix for a super element by condensation of the constituent finite elements, the stiffness matrix is derived using symbolic differentiation. Steenbergen argues that the main benefit of using symbolic differentiation rather than FEM is to provide more insight into the structural behaviour of a model. Although FEM programs produce quite accurate results for the structural behaviour of a building under a certain set of loads, they cannot show which structural parameters govern the response of the building to these loads. For example, floor stiffness can have a significant effect on the behaviour of cores and shear walls, but this relationship can be difficult to detect from the output of FEM software (Steenbergen, 2007).

Steenbergen (2007) analyses concrete buildings composed of only floors and stability elements (cores and shear walls), subject to uniform wind pressure in the x and y directions. The wind load is transferred to the stability elements via the in-plane direction of the floors. The floors are considered as flexural beams supported by pin supports (in the case of shear walls) or fixed supports (in the case of cores). The floors have no contribution to the system in the out-of-plane direction. Shear walls and cores are considered as flexural beams fixed at the foundation. The cores have torsional rigidity whereas the shear walls have no torsional rigidity. For both floors and stability elements, shear lag is ignored (Steenbergen, 2007).

Super elements are formulated from configurations of shear walls and/or cores that are uniform over a certain height. One individual super element comprises of a system of stability elements and floors acting together. Forces from the floors are approximated as uniformly distributed loads along the stability elements (rather than discrete point loads at the floor locations). An image of a super element composed of one shear wall and one core from Steenbergen (2007) is shown in Figure A.1.

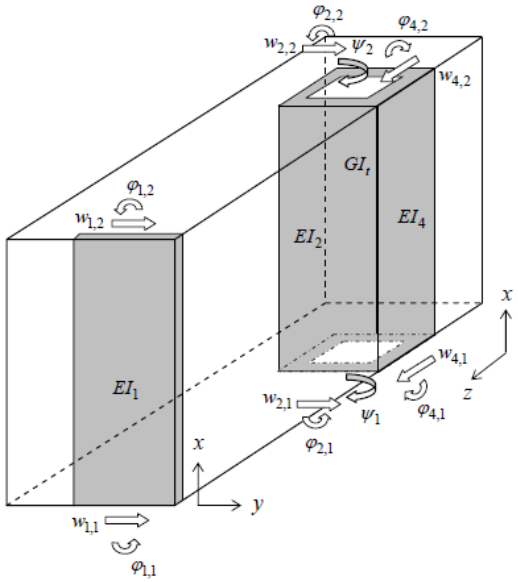


Figure A.1 Super element composed of a shear wall and core (Steenbergen, 2007)

To derive the stiffness matrix for a super element, Steenbergen uses the following method:

The stiffness matrix describes the relationship between the forces and the displacements at the *upper and lower* boundaries of the super element:

$$f = Kd \quad (1)$$

where f is the force vector, d is the displacement vector and K is the stiffness matrix at the nodes of the system.

A system of differential equations can be formulated to describe the relationship between the displacement of the stability elements and the force applied on them. By setting the external forces to zero, a homogeneous solution can be found to describe the displacements of the stability elements. The homogeneous solution has n unknown constants C. The displacements at the boundaries of the super element are represented by vector d.

$$HC = d \quad (2)$$

$$C = H^{-1}d \quad (3)$$

where d is the displacement vector, C is the unknown constants in the homogeneous solution, and H is the terms in front of the unknown constants.

The homogenous solution can be derived to determine expressions representing the force in the super element. The vector f represents the forces at the boundaries of the super element.

$$f = GC \quad (4)$$

where f is the force vector, C is the unknown constants in the homogeneous solution, and G is the terms in front of the unknown constants.

Substituting (3) into (4):

$$f = G H^{-1} d \qquad (5)$$

Substituting (5) into (1), the stiffness matrix can be derived:

$$K = G H^{-1} \qquad (6)$$

The stiffness matrix K can be used to find the unknown displacements d, which can then be used with expression (3) to determine the constants C. Combining the solved homogeneous solution with the particular solution, a complete expression for the displacements of the stability elements in a super element can be determined (Steenbergen, 2007).

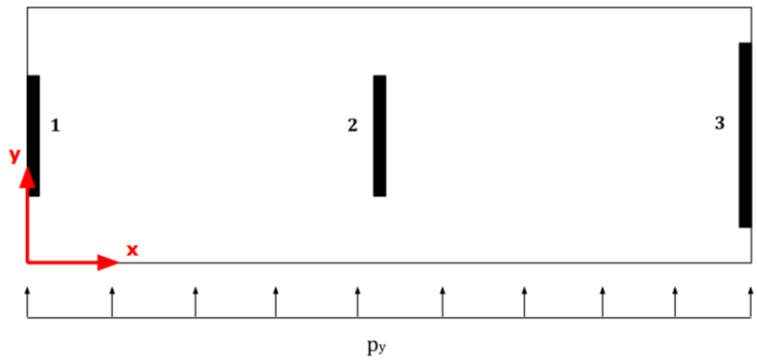
Different super elements can be combined together to create more complex building types; however, the combination of super elements is limited. Different super elements can only be combined vertically (and not horizontally). This is because the stiffness matrix is oriented in the vertical direction – the nodes of the system are at the base and the top of each super element. To combine super elements vertically, one simply needs to create a global stiffness matrix from the local super element matrices and define appropriate boundary and matching conditions. In the horizontal direction however, the super element is defined by a system of differential equations representing the relationships between the stability elements and the floors connecting them. If the floorplan changes, this system of differential equations must be redefined to match the new situation, and a whole new super element must be created (Steenbergen, 2007).

Appendix B – Maple Scripts

Test 1: Three shear walls connected by rigid floors

```
> restart;
> with(plots):
```

1-dimensional example with 3 walls, as shown in the image below:



1) Define inputs

```
> py:= 1.45: (*kPa*)
> E:= 30000000: (*kPa*)
> height:=48.0: (*m*)
> x1:=0: width1:=6.0: thickness1:=0.4: (*m*)
> x2:=20: width2:=6.0: thickness2:=0.4: (*m*)
> x3:=40: width3:=12.0: thickness3:=0.4: (*m*)
> floor_length:= 40: (*m*)
```

2) Calculate stiffness values

```
> EI1:= E*thickness1*width1^3/12:
> EI2:= E*thickness2*width2^3/12:
> EI3:= E*thickness3*width3^3/12:
```

3) Determine floor rotational centre, eccentricity of wind load and eccentricity of floors

```
> rot_centre:= (EI1*x1 + EI2*x2 + EI3*x3)/(EI1+EI2+EI3):

> cog_wind:= floor_length/2:

> e:= rot_centre - cog_wind: (*m*)

> a1:= rot_centre-x1: a2:= rot_centre-x2: a3:= rot_centre-x3:
```

4) Construct ODE's and solve for equations

```
> ODE1:= EI1*diff(u1(z),z$4) + EI2*diff(u2(z),z$4) +
EI3*diff(u3(z),z$4) = py*floor_length;
```

$$ODE1 := 2.160000000\ 10^{8} \frac{d^4}{dz^4} u1(z) + 2.160000000\ 10^{8} \frac{d^4}{dz^4} u2(z) + 1.728000000\ 10^{9} \frac{d^4}{dz^4} u3(z) = 58.00$$

```
> ODE2:= a1*EI1*diff(u1(z),z$4) + a2*EI2*diff(u2(z),z$4) +
a3*EI3*diff(u3(z),z$4) = py*floor_length*e;
```

$$ODE2 := 7.344000000\ 10^{9} \frac{d^4}{dz^4} u1(z) + 3.024000000\ 10^{9} \frac{d^4}{dz^4} u2(z) - 1.036800000\ 10^{10} \frac{d^4}{dz^4} u3(z) = 812.0000000$$

```
> equilib:= (u1(z)+u3(z))/2=u2(z);
```

$$equilib := \frac{u1(z)}{2} + \frac{u3(z)}{2} = u2(z)$$

```
> sol:= dsolve({ODE1, ODE2, equilib},{u1(z), u2(z), u3(z)}):
assign(sol):

> u1:= u1(z): u2:=u2(z): u3:=u3(z):

> phi1:=-diff(u1,z): kappa1:=diff(phi1,z): M1:=EI1*kappa1:
V1:=diff(M1,z):

> phi2:=-diff(u2,z): kappa2:=diff(phi2,z): M2:=EI2*kappa2:
V2:=diff(M2,z):

> phi3:=-diff(u3,z): kappa3:=diff(phi3,z): M3:=EI3*kappa3:
V3:=diff(M3,z):
```

5) Apply boundary conditions

```
> z:=0: eq1:=u1=0: eq2:=phi1=0: eq3:=u2=0: eq4:=phi2=0:

> z:=height: eq5:=M1=0: eq6:=V1=0: eq7:=M2=0: eq8:=V2=0:
```

```
>
sol:=solve({eq1,eq2,eq3,eq4,eq5,eq6,eq7,eq8},{_C1,_C2,_C3,_C4,_C5,_C
6,_C7,_C8}): assign(sol): z:='z':
```

6) Evalute the maximum values (for comparison to FEM results)

```
> u1_max:= eval(u1,z=height); u2_max:= eval(u2,z=height); u3_max:=
eval(u3,z=height);
```

$$u1\_max := 0.06953209757$$

$$u2\_max := 0.03911180489$$

$$u3\_max := 0.008691512262$$

```
>   V1_max:=   eval(V1,z=0);   V2_max:=   eval(V2,z=0);   V3_max:=
eval(V3,z=0);
```

$$V1\_max := 1086.439024$$

$$V2\_max := 611.1219515$$

$$V3\_max := 1086.439033$$

```
>   M1_max:=   eval(M1,z=0);   M2_max:=   eval(M2,z=0);   M3_max:=
eval(M3,z=0);
```

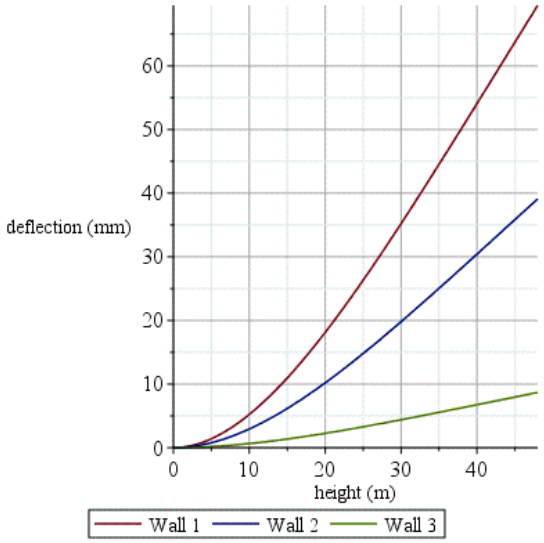
$$M1\_max := -26074.53657$$

$$M2\_max := -14666.92684$$

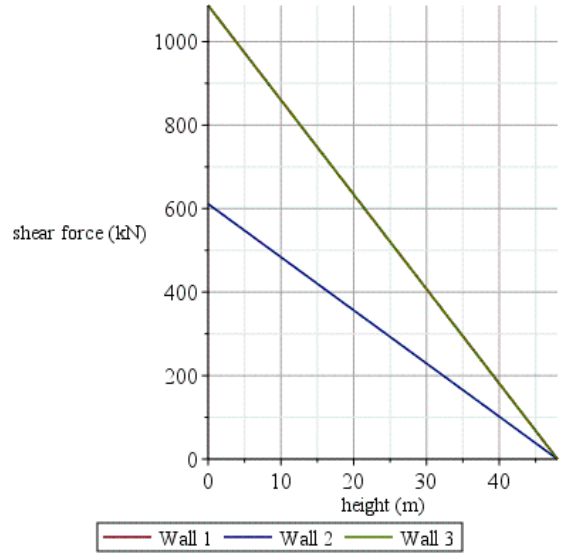
$$M3\_max := -26074.5368$$

7) Plot results

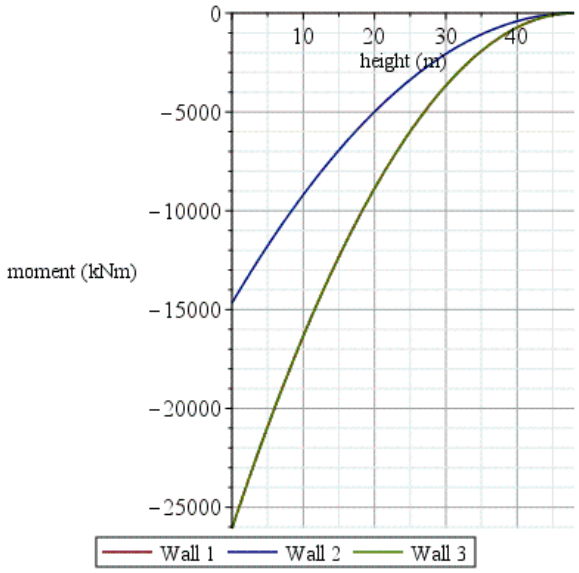
```
> plot([u1*1000,u2*1000,u3*1000],z=0..height,legend=["Wall 1", "Wall
2",   "Wall   3"],gridlines=true,labels=["height   (m)",   "deflection
(mm)"]);
```




```
>  plot([V1,V2,V3],z=0..height,legend=["Wall   1",   "Wall   2",   "Wall
3"],gridlines=true,labels=["height (m)", "shear force (kN)"])
```



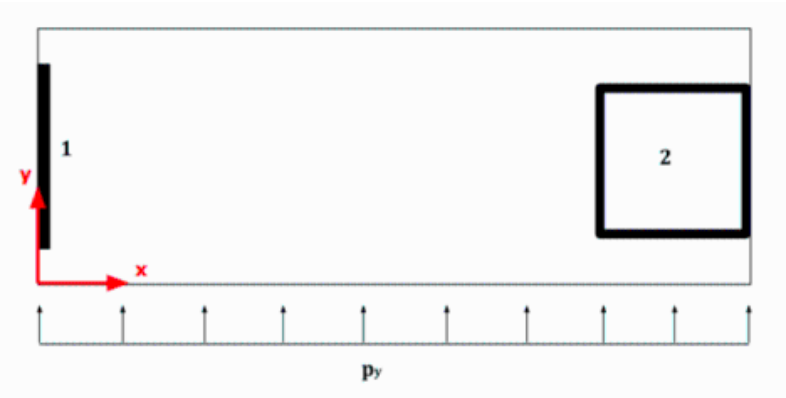

```
> plot([M1,M2,M3],z=0..height,legend=["Wall 1", "Wall 2", "Wall 3"],gridlines=true,labels=["height (m)", "moment (kNm)"]);
```



Test 2: Shear wall and core connected by rigid floors

```
> restart;
> with (plots):
```

Core and shear wall connected by infinitely rigid floor



1) Define inputs

```
> py:= 1.45: (*kPa*)
> E:= 30000000: (*kPa*)
> vpoisson:= 0.2:
> height:= 48.0:
> floor_length:= 40: (*m*)
> x1:= 0: width1:= 8.0: thickness1:= 0.4: (*m*)
> x2:= 36: width2:= 6.0: thickness2:= 0.2: (*m*)
```

2) Calculate stiffness values

```
> G:= E/(2*(1+vpoisson)):
> EI1:= E*thickness1*width1^3/12:
> EI2:= E*((width2*width2^3)/12 - 1/12*(width2-2*thickness2)*(width2-
2*thickness2)^3):
>  GIt:=  G*(2*((width2-thickness2)*(width2-thickness2))^2)/((width2-
thickness2)/thickness2+(width2-thickness2)/thickness2):
```

3) Construct ODE's and solve for equations

```
> ODE1:= EI1*diff(u1(z),z$4) + EI2*diff(u2(z),z$4) =
py*(floor_length);
```

$$ODE1 := 5.120000000\ 10^{8} \frac{d^4}{dz^4} u1(z) + 7.813759998\ 10^{8} \frac{d^4}{dz^4} u2(z) = 58.00$$

```
> ODE2:= x1*EI1*diff(u1(z),z$4) + x2*EI2*diff(u2(z),z$4) -
GIt*diff(psi(z),z$2) = py*(floor_length)^2/2;
```

$$ODE2 := 2.812953599\ 10^{10} \frac{d^4}{dz^4} u2(z) - 4.877800000\ 10^{8} \frac{d^2}{dz^2} \psi(z) = 1160.000000$$

```
> ODE3:= (u2(z)-u1(z))/(x2-x1) = psi(z);
```

$$ODE3 := \frac{u2(z)}{36} - \frac{u1(z)}{36} = \psi(z)$$

```
> sol:= dsolve({ODE1, ODE2, ODE3},{u1(z), u2(z), psi(z)}):
assign(sol):

> u1:= u1(z): u2:=u2(z): psi:=psi(z):

> phi1:=-diff(u1,z): kappa1:=diff(phi1,z): M1:=EI1*kappa1:
V1:=diff(M1,z):

> phi2:=-diff(u2,z): kappa2:=diff(phi2,z): M2:=EI2*kappa2:
V2:=diff(M2,z):

> T:= GIt*diff(psi, z):
```

4) Apply boundary conditions

```
> z:=0: eq1:=u1=0: eq2:=phi1=0: eq3:=u2=0: eq4:=phi2=0:

> z:=height: eq5:=M1=0: eq6:=V1=0: eq7:=M2=0: eq8:=V2=0:

>
sol:=solve({eq1,eq2,eq3,eq4,eq5,eq6,eq7,eq8},{_C1,_C2,_C3,_C4,_C5,_C
6,_C7,_C8}): assign(sol): z:='z':
```

5) Evalute the maximum values (for comparison to FEM results)

```
> u1_max:= simplify(eval(u1,z=height)); u2_max:=
simplify(eval(u2,z=height));
```

$$u1\_max := 0.03401825561$$

$$u2\_max := 0.02696354778$$

```
> V1_max:= simplify(eval(V1,z=0)); V2_max:= simplify(eval(V2,z=0));
```

$$V1\_max := 1310.001484$$

$$V2\_max := 1473.998515$$

```
> M1_max:= eval(M1,z=0); M2_max:= eval(M2,z=0);
```

$$M1\_max := -30528.86646$$

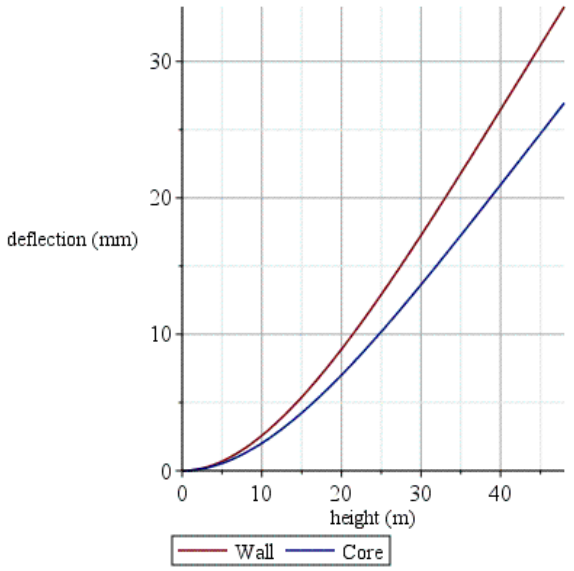
$$M2\_max := -36287.13353$$

```
> T_max:= simplify(eval(T,z=height));
```

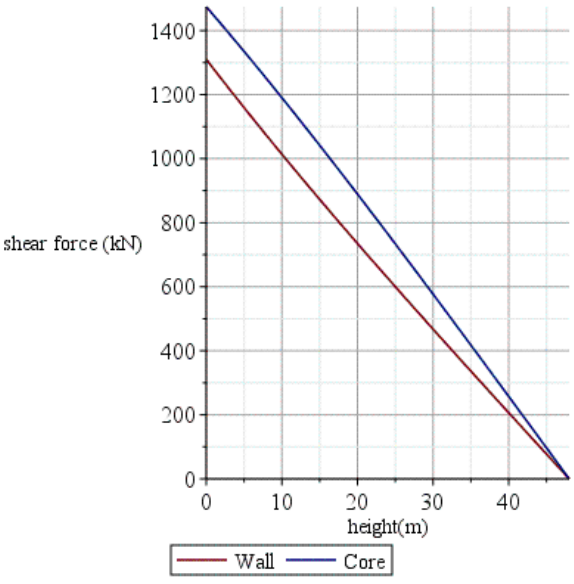
$$T\_max := -2616.053483$$

6) Plot results

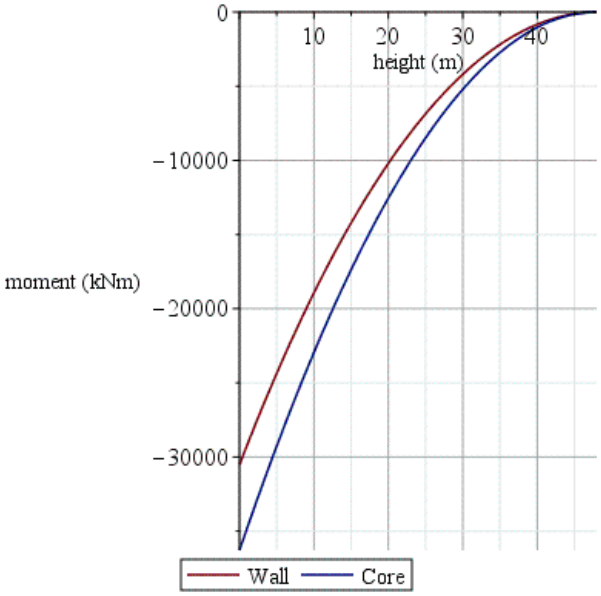
```
>                plot([u1*1000,u2*1000],                    z=0..height,
legend=["Wall","Core"],gridlines=true,labels=["height              (m)",
"deflection (mm)"]);
```



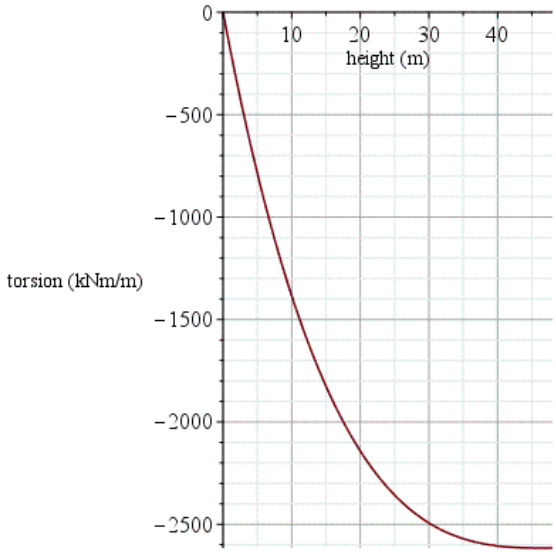

```
>
plot([V1,V2],z=0..height,legend=["Wall","Core"],gridlines=true,label
s=["height(m)", "shear force (kN)"]);
```



```
> plot([M1,M2],z=0..height,legend=["Wall", "Core"],gridlines=true,labels=["height (m)", "moment (kNm)"])
```




```
> plot(T, z=0..height,gridlines=true,labels=["height (m)", "torsion (kNm/m)"])
```

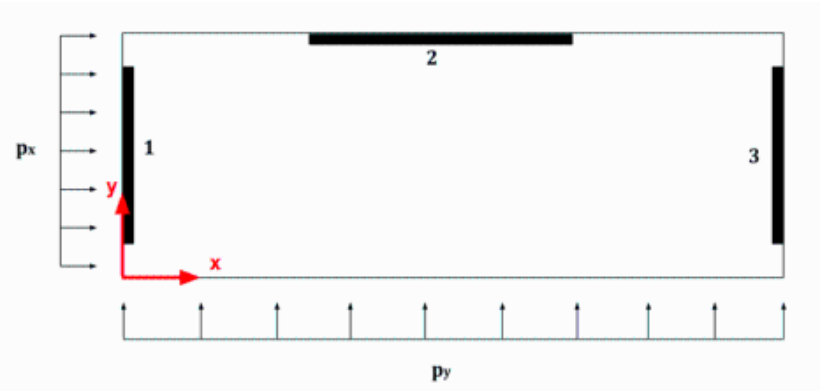



```
>
```

Test 3: Three shear walls connected by a rigid floor – two-dimensional

```
> restart;
> with(plots):
```

2-dimensional example with 3 walls, as shown in the image below:



The number belonging to each wall is labelled next to it. The z-axis points out of the page.

1) Define inputs

```
> px:= 0: py:=1.45: (*kPa*)
> E:= 30000000: (*kPa*)
> height:= 48.0: (*m*)
> l:= 40.0: w:= 15.0: (*m*)
> x1:= 0: y1:=7.5: width1:=6: thickness1:=0.4: (*m*)
> x2:= 20: y2:=15: width2:=0.2: thickness2:=8: (*m*)
> x3:= 40: y3:=7.5: width3:=6: thickness3:=0.2: (*m*)
```

2) Calculate stiffness values. For my analysis, EIy1 refers to stiffness *in the direction* of y.

```
> EIy1:= E*thickness1*width1^3/12:
> EIy2:= E*thickness2*width2^3/12:
> EIy3:= E*thickness3*width3^3/12:
> EIx1:= E*width1*thickness1^3/12:
> EIx2:= E*width2*thickness2^3/12:
> EIx3:= E*width3*thickness3^3/12:
```

3) Determine floor rotational centre, eccentricity of wind load and eccentricity of floors

```
> x_centre:= (x1*EIy1 + x2*EIy2 + x3*EIy3)/(EIy1 + EIy2 + EIy3);
```

$$x\_centre := 13.33662389$$

```
> y_centre:= (y1*EIx1 + y2*EIx2 + y3*EIx3)/(EIx1 + EIx2 + EIx3);
```

$$y\_centre := 14.96849230$$

```
> cog_px:= w/2:

> cog_py:= l/2: (*for both assuming uniformly distributed loads*)

> ey:= cog_px - y_centre;
```

$$ey := -7.468492300$$

```
> ex:= x_centre - cog_py;
```

$$ex := -6.66337611$$

```
> a1 := x_centre-x1: a2 := x_centre-x2: a3 := x_centre-x3:

> b1 := y1-y_centre: b2 := y2-y_centre: b3 := y3-y_centre:
```

4) Construct ODE's and solve for equations

```
> uy1(z) := uy(z)+f(z)*(a1):

> uy2(z) := uy(z)+f(z)*(a2):

> uy3(z) := uy(z)+f(z)*(a3):

> ux1(z) := uz(z)+f(z)*(b1):

> ux2(z) := uz(z)+f(z)*(b2):

> ux3(z) := uz(z)+f(z)*(b3):

> eq1:= EIy1*diff(uy1(z),z$4) + EIy2*diff(uy2(z),z$4) +
EIy3*diff(uy3(z),z$4) = py*l:

eq2:= EIx1*diff(ux1(z),z$4) + EIx2*diff(ux2(z),z$4) +
EIx3*diff(ux3(z),z$4) = px*w:

eq3:= a1*EIy1*diff(uy1(z),z$4) + a2*EIy2*diff(uy2(z),z$4) +
a3*EIy3*diff(uy3(z),z$4) + b1*EIx1*diff(ux1(z),z$4) +
b2*EIx2*diff(ux2(z),z$4) + b3*EIx3*diff(ux3(z),z$4) = py*l*ex +
px*w*ey:

> sol:= dsolve({eq1,eq2,eq3},{uy(z),uz(z),f(z)}): assign(sol):

> uy1:= eval(uy1(z)): uy2:= eval(uy2(z)): uy3:= eval(uy3(z)):

> ux1:= eval(ux1(z)): ux2:= eval(ux2(z)): ux3:= eval(ux3(z)):

> phiy1:=-diff(uy1,z): kappay1:=diff(phiy1,z): My1:=EIy1*kappay1:
Vy1:=diff(My1,z):

> phiy2:=-diff(uy2,z): kappay2:=diff(phiy2,z): My2:=EIy2*kappay2:
Vy2:=diff(My2,z):
```

```
> phiy3:=-diff(uy3,z): kappay3:=diff(phiy3,z): My3:=EIy3*kappay3:
Vy3:=diff(My3,z):

> phix1:=-diff(ux1,z): kappax1:=diff(phix1,z): Mx1:=EIx1*kappax1:
Vx1:=diff(Mx1,z):

> phix2:=-diff(ux2,z): kappax2:=diff(phix2,z): Mx2:=EIx2*kappax2:
Vx2:=diff(Mx2,z):

> phix3:=-diff(ux3,z): kappax3:=diff(phix3,z): Mx3:=EIx3*kappax3:
Vx3:=diff(Mx3,z):
```

5) Apply boundary conditions

```
> z:=0: bc1:=uy1=0: bc2:=phiy1=0: bc3:=uy2=0: bc4:=phiy2=0:
bc5:=ux1=0: bc6:=phix1=0:

> z:=height: bc7:=My1=0: bc8:=Vy1=0: bc9:=My2=0: bc10:=Vy2=0: bc11:=
Mx1=0: bc12:=Vx1=0:

>
sol:=solve({bc1,bc2,bc3,bc4,bc5,bc6,bc7,bc8,bc9,bc10,bc11,bc12},{_C1
,_C2,_C3,_C4,_C5,_C6,_C7,_C8,_C9,_C10,_C11,_C12}): assign(sol):
z:='z':
```

6) Evaluate the maximum values (for comparison to FEM results)

```
> uy1_max = eval(uy1,z=height); uy2_max = eval(uy2,z=height); uy3_max
= eval(uy3,z=height); (*m*)
```

$$uy1\_max = 0.08905411440$$

$$uy2\_max = 0.1335500169$$

$$uy3\_max = 0.1780459192$$

```
> ux1_max = eval(ux1,z=height); ux2_max = eval(ux2,z=height); ux3_max
= eval(ux3,z=height); (*m*)
```

$$ux1\_max = 0.01661586524$$

$$ux2\_max = -0.00007009816554$$

$$ux3\_max = 0.01661586524$$

```
> Vy1_max = eval(Vy1,z=0); Vy2_max = eval(Vy2,z=0); Vy3_max =
eval(Vy3,z=0); (*kN*)
```

$$Vy1\_max = 1391.470538$$

$$Vy2\_max = 1.545717788$$

$$Vy3\_max = 1390.983745$$

```
> Vx1_max = eval(Vx1,z=0); Vx2_max = eval(Vx2,z=0); Vx3_max =
eval(Vx3,z=0); (*kN*)
```

$$Vx1\_max = 1.153879530$$

$$Vx2\_max = -1.298114642$$

$$Vx3\_max = 0.1442349413$$

```
> My1_max = eval(My1,z=0); My2_max = eval(My2,z=0); My3_max =
eval(My3,z=0); (*kNm*)
```

$$My1\_max = -33395.29292$$

$$My2\_max = -37.09722691$$

$$My3\_max = -33383.60988$$

```
> Mx1_max = eval(Mx1,z=0); Mx2_max = eval(Mx2,z=0); Mx3_max =
eval(Mx3,z=0); (*kNm*)
```

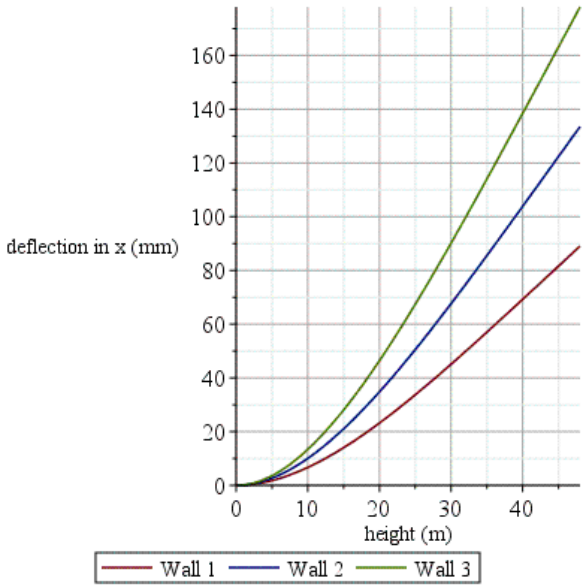
$$Mx1\_max = -27.69310872$$

$$Mx2\_max = 31.15474652$$

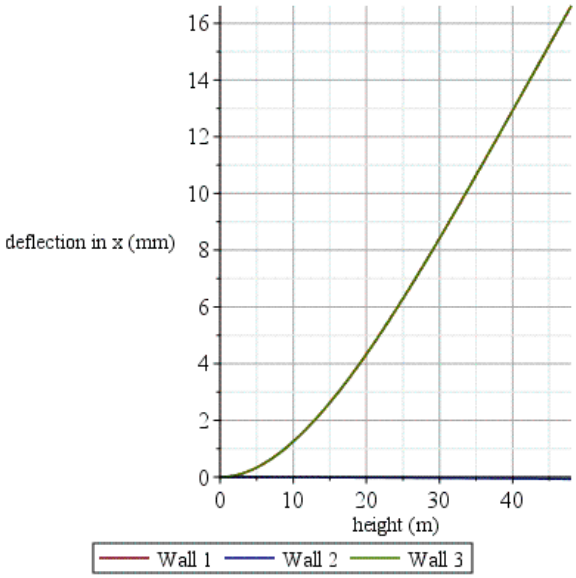
$$Mx3\_max = -3.461638589$$

7) Plot results

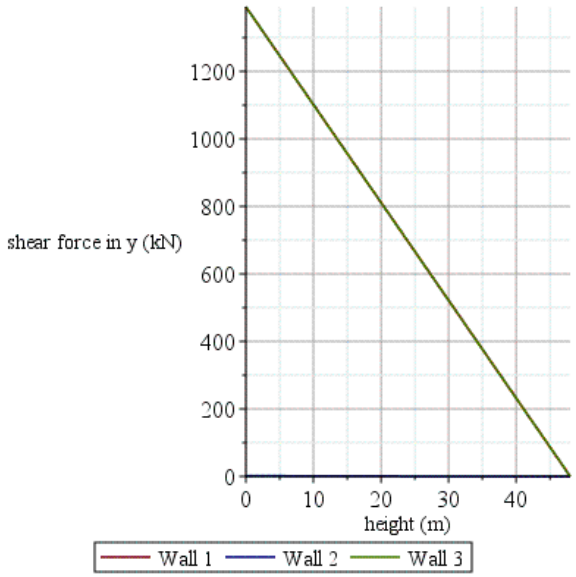
```
> plot([uy1*1000,uy2*1000,uy3*1000],z=0..height,legend=["Wall 1",
"Wall 2", "Wall 3"],gridlines=true,labels=["height (m)", "deflection
in x (mm)"]);
```



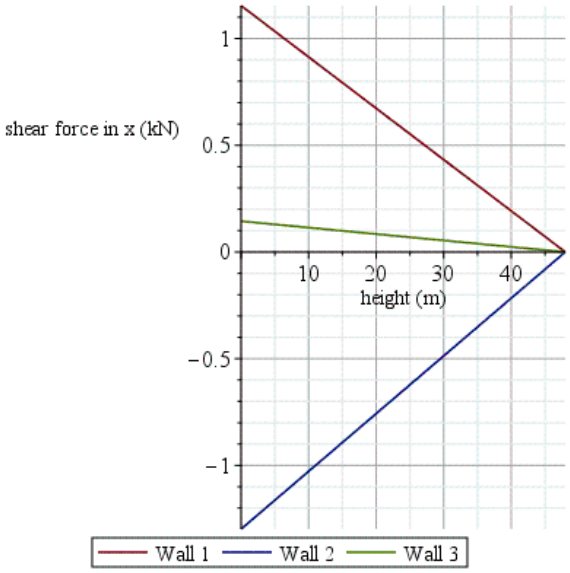

```
> plot([ux1*1000,ux2*1000,ux3*1000],z=0..height,legend=["Wall 1",
"Wall 2", "Wall 3"],gridlines=true,labels=["height (m)", "deflection
in x (mm)"]);
```



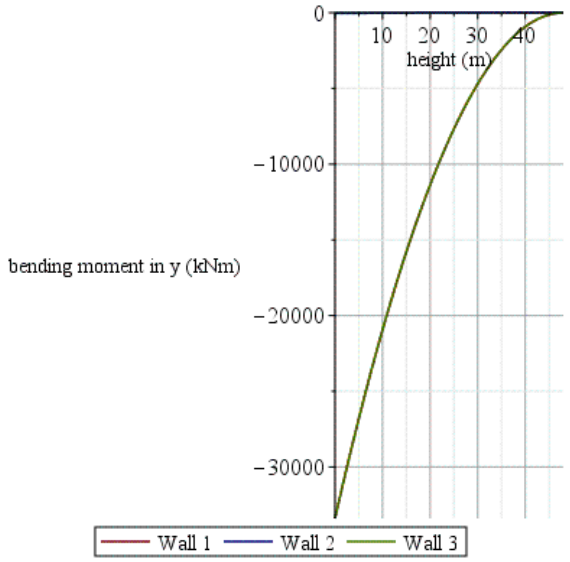

```
> plot([Vy1,Vy2,Vy3],z=0..height,legend=["Wall 1", "Wall 2", "Wall
3"],gridlines=true,labels=["height (m)", "shear force in y (kN)"]);
```



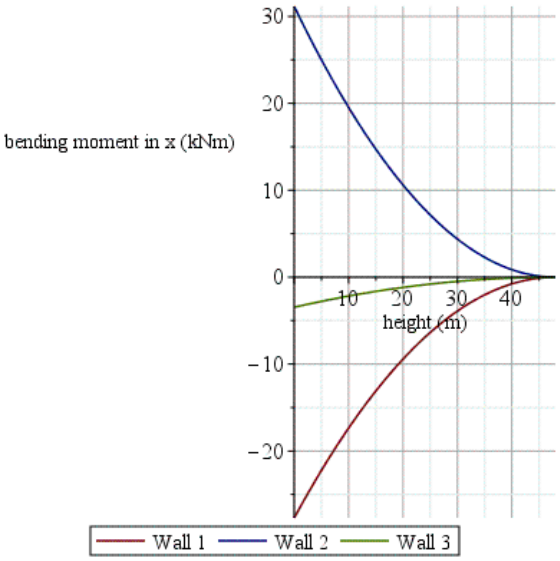

```
> plot([Vx1,Vx2,Vx3],z=0..height,legend=["Wall 1", "Wall 2", "Wall
3"],gridlines=true,labels=["height (m)", "shear force in x (kN)"]);
```




```
> plot([My1,My2,My3],z=0..height,legend=["Wall 1", "Wall 2", "Wall 3"],gridlines=true,labels=["height (m)", "bending moment in y (kNm)"]);
```

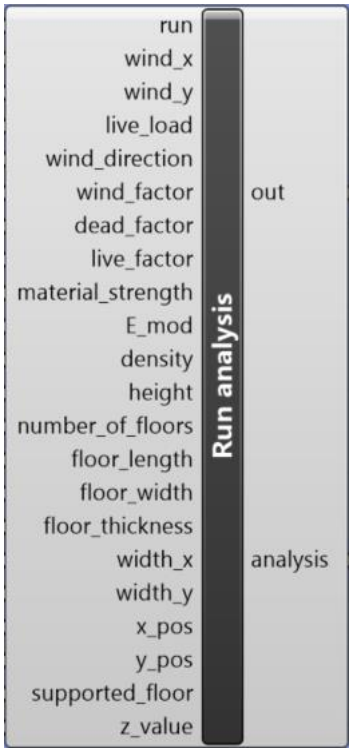



```
> plot([Mx1,Mx2,Mx3],z=0..height,legend=["Wall 1", "Wall 2", "Wall 3"],gridlines=true,labels=["height (m)", "bending moment in x (kNm)"]);
```


bending moment in x (kNm)
30
20
10
0
−10
−20
10
20
30
40
height (m)
Wall 1
Wall 2
Wall 3

>

Appendix C – Python Script




```
import math

import time
start = time.time()

import scriptcontext as sc
sp = sc.sticky['sympy']

#definitions
def calculate_moment_inertia(width,thickness):
    moment_inertia = 1/12*thickness*(width**3)
    return moment_inertia

def list_to_tree(input, none_and_holes=True, source=[0]):
    """Transforms nestings of lists or tuples to a Grasshopper DataTree"""
    # written by Giulio Piacentino, giulio@mcneel.com
    # source: https://gist.github.com/piac/ef91ac83cb5ee92a1294
    from Grasshopper import DataTree as Tree
    from Grasshopper.Kernel.Data import GH_Path as Path
    from System import Array
    def proc(input,tree,track):
        path = Path(Array[int](track))
        if len(input) == 0 and none_and_holes: tree.EnsurePath(path);
return
        for i,item in enumerate(input):
            if hasattr(item, '__iter__'): #if list or tuple
                track.append(i); proc(item,tree,track); track.pop()
            else:
                if none_and_holes: tree.Insert(item,path,i)
                elif item is not None: tree.Add(item,path)
```

```
    if input is not None: t=Tree[object]();proc(input,t,source[:]);return t

#start analysis
if run == True:
    print "Running..."

    px = wind_x
    py = wind_y
    live_load = live_load
    wind_dir = wind_direction
    fw = wind_factor
    fd = dead_factor
    fl = live_factor
    fck = material_strength
    E_mod = E_mod
    density = density
    height = height
    num_floors = number_of_floors
    f_length = floor_length
    f_width = floor_width
    f_thickness = floor_thickness
    w_xwidth = width_x
    w_ywidth = width_y
    w_xpos = x_pos
    w_ypos = y_pos
    sup_area = supported_floor
    nodes = z_value

    everything = []

    #define symbols
    z, m = sp.var('z m')

    ht = float(height)

    #empty lists
    Cx_all = []
    Cy_all = []
    Vx_all = []
    Vy_all = []
    Mx_all = []
    My_all = []
    ux_all = []
    uy_all = []

    #set load factors to 1 if not specified by user
    if fw:
        fw = fw
    else:
        fw = 1

    if fd:
        fd = fd
    else:
        fd = 1

    if fl:
        fl = fl
    else:
        fl = 1
```

```
    #tell the user to add a wind load if it's missing
    if px:
        Qx = fw*px*f_width
    else:
        Qx = 0
        print "Please enter a wind load in the x-direction"

    if py:
        Qy = fw*py*f_length
    else:
        Qy = 0
        print "Please enter a wind load in the y-direction"

    #determine area, stiffness values, cog of loads and eccentricity
    A = []
    Wx = []
    Wy = []
    EIx = []
    EIy = []
    width_x = []
    width_y = []
    x = []
    y = []
    x_centre_num = 0
    y_centre_num = 0
    steiner_x = []
    steiner_y = []

    number_of_walls = len(w_xwidth)

    for wall in range(0,number_of_walls):
        w_x = float(w_xwidth[wall])
        w_y = float(w_ywidth[wall])
        x_value = float(w_xpos[wall])
        y_value = float(w_ypos[wall])
        single_A = w_x*w_y
        single_Wx = w_y*w_x**2/6
        single_Wy = w_x*w_y**2/6
        single_EIx = E_mod*1000000*calculate_moment_inertia(w_x,w_y)
        single_EIy = E_mod*1000000*calculate_moment_inertia(w_y,w_x)
        x_centre_num += x_value*single_EIy
        y_centre_num += y_value*single_EIx
        width_x.append(w_x)
        width_y.append(w_y)
        x.append(x_value)
        y.append(y_value)
        A.append(single_A)
        Wx.append(single_Wx)
        Wy.append(single_Wy)
        EIx.append(single_EIx)
        EIy.append(single_EIy)

    EIx_tot = sum(EIx)
    EIy_tot = sum(EIy)

    x_centre = x_centre_num/EIy_tot
    y_centre = y_centre_num/EIx_tot
    cog_px = f_width/2 #assuming a uniformly-distributed wind load
    cog_py = f_length/2 #same
    ex = x_centre - cog_py #opposite direction from bending equilibrium &
ref. axes
```

```
    ey = cog_px - y_centre

    #determine vertical load from dead load and live load
    N = []
    wall_load = []
    for wall in range(0,number_of_walls):
        N_single = fd*A[wall]*ht*density
        wall_load_single = N_single +
sup_area[wall]*num_floors*(fd*density*f_thickness + fl*live_load)
        N.append(N_single)
        wall_load.append(wall_load_single)

    dead_floor_total = fd*density*f_length*f_width*f_thickness*num_floors
    live_floor_total = fl*f_length*f_width*num_floors*live_load

    N_tot = sum(wall_load)

    #initialise two wind analyses, one with Qx and one with Qy
    Qx_all = [Qx, 0]
    Qy_all = [0, Qy]

    num_analyses = len(Qx_all)

    for num in range(0,num_analyses):
        Qx = Qx_all[num]
        Qy = Qy_all[num]

        #compile system of equations
        #note: eq1 and eq2 are force equilibrium in x and y, eq3 is moment
equilibrium
        eq1_lhs = 0
        eq2_lhs = 0
        eq3_lhs = 0
        eq1_rhs = Qx
        eq2_rhs = Qy
        eq3_rhs = Qx*ey + Qy*ex

        #Get functions for the displacements
        ux, uy, phi = sp.symbols('ux uy phi', cls=sp.Function)

        a_list = []
        b_list = []

        for wall in range(number_of_walls):
            a = x_centre-x[wall]
            b = y[wall]-y_centre
            ux_diff = sp.diff(ux(z) + phi(z)*b,z,z,z,z)
            uy_diff = sp.diff(uy(z) + phi(z)*a,z,z,z,z)
            eq1_lhs += EIx[wall]*ux_diff
            eq2_lhs += EIy[wall]*uy_diff
            eq3_lhs += b*EIx[wall]*ux_diff + a*EIy[wall]*uy_diff
            a_list.append(a)
            b_list.append(b)

        eq1 = sp.Eq(eq1_lhs,eq1_rhs)
        eq2 = sp.Eq(eq2_lhs,eq2_rhs)
        eq3 = sp.Eq(eq3_lhs,eq3_rhs)

        eq = eq1,eq2,eq3
```

```
        #Stack overflow post:
https://stackoverflow.com/questions/56133257/sympy-limitations-in-the-
computation-ability-of-dsolve-function
        #code (Benjamin, 22 May 2019)
        derivs = [u(z).diff(z, 4) for u in (ux, uy, phi)]
        (sol,) = sp.solve(eq, derivs, dict=True)
        eq = [sp.Eq(du, sol[du]) for du in derivs]

        #Boundary conditions for the global variables
        ics1 =
{ux(0):0,ux(z).diff(z).subs(z,0):0,ux(z).diff(z,2).subs(z,ht):0,ux(z).diff(
z,3).subs(z,ht):0}
        ics2 =
{uy(0):0,uy(z).diff(z).subs(z,0):0,uy(z).diff(z,2).subs(z,ht):0,uy(z).diff(
z,3).subs(z,ht):0}
        ics3 =
{phi(0):0,phi(z).diff(z).subs(z,0):0,phi(z).diff(z,2).subs(z,ht):0,phi(z).d
iff(z,3).subs(z,ht):0}

        #solving for the global variables
        ux_sol = sp.dsolve(eq[0],ics=ics1)
        uy_sol = sp.dsolve(eq[1],ics=ics2)
        phi_sol = sp.dsolve(eq[2],ics=ics3)

        sol1 = sp.solve(ux_sol, ux(z))[0]
        sol2 = sp.solve(uy_sol, uy(z))[0]
        sol3 = sp.solve(phi_sol, phi(z))[0]

        #solve for spring values at the base of the walls
        Cx = []
        Cy = []

        #solve for nodal values
        Vx = []
        Vy = []
        Mx = []
        My = []
        ux = []
        uy = []


        #compile nested lists
        for wall in range(number_of_walls):
            ux_single = sol1 + sol3*b_list[wall]
            uy_single = sol2 + sol3*a_list[wall]
            Ox_single = -1*sp.diff(ux_single,z)
            Oy_single = -1*sp.diff(uy_single,z)
            Mx_single = EIx[wall]*sp.diff(Ox_single,z)
            My_single = EIy[wall]*sp.diff(Oy_single,z)
            Vx_single = sp.diff(Mx_single,z)
            Vy_single = sp.diff(My_single,z)

            #add springs
            Cx_single = abs(float(Mx_single.subs({z:0}))*1000)
            Cy_single = abs(float(My_single.subs({z:0}))*1000)

            Cx.append(Cx_single)
            Cy.append(Cy_single)
            Vx.append(Vx_single)
            Vy.append(Vy_single)
            Mx.append(Mx_single)
```

```
            My.append(My_single)
            ux.append(ux_single)
            uy.append(uy_single)

        Cx_all.append(Cx)
        Cy_all.append(Cy)
        Vx_all.append(Vx)
        Vy_all.append(Vy)
        Mx_all.append(Mx)
        My_all.append(My)
        ux_all.append(ux)
        uy_all.append(uy)

    Cx_recommended = Cx_all[0]
    Cy_recommended = Cy_all[1]

    #calculate factors for second-order effect
    Cx_num = 0
    Cy_num = 0
    for wall in range(0,number_of_walls):
        Cx_num += Cx_recommended[wall]*EIx[wall]
        Cy_num += Cy_recommended[wall]*EIy[wall]

    Cx_tot = Cx_num/EIx_tot
    Cy_tot = Cy_num/EIy_tot
    Qcrfx = 2*Cx_tot/ht
    Qcrfy = 2*Cy_tot/ht
    Qcrbx = 8*EIx_tot/(ht**2)
    Qcrby = 8*EIy_tot/(ht**2)
    nx = (1/Qcrfx + 1/Qcrbx)**(-1)/N_tot
    ny = (1/Qcrfy + 1/Qcrby)**(-1)/N_tot

    x_factor = nx/(nx-1)
    y_factor = ny/(ny-1)
    #print x_factor
    #print y_factor

    #determine appropriate wind results depending on direction of wind
    if wind_dir == "x":
        Vx = Vx_all[0]
        Vy = Vy_all[0]
        Mx = Mx_all[0]
        My = My_all[0]
        ux = ux_all[0]
        uy = uy_all[0]
    elif wind_dir == "y":
        Vx = Vx_all[1]
        Vy = Vy_all[1]
        Mx = Mx_all[1]
        My = My_all[1]
        ux = ux_all[1]
        uy = uy_all[1]


    #print " "
    #print "Recommended foundation stiffnesses:"

    Vx_total = []
    Vy_total = []
    Mx_total = []
    My_total = []
```

```
ux_total = []
uy_total = []

wall_col_stiffness = []
wall_col_comp_strength = []
wall_col_shear_strength = []
wall_col_stability = []

for wall in range(number_of_walls):
    Vx_wall = []
    Vy_wall = []
    Mx_wall = []
    My_wall = []
    ux_wall = []
    uy_wall = []
    ux_single = ux[wall]
    uy_single = uy[wall]
    Mx_single = Mx[wall]
    My_single = My[wall]
    Vx_single = Vx[wall]
    Vy_single = Vy[wall]

    Mx_base = []
    My_base = []

    #add springs
    Cx_single = Cx_recommended[wall]
    Cy_single = Cy_recommended[wall]
    print " "
    print "Wall %d" %(wall+1)
    print "Foundation stiffness in x: %d kNm/rad" %Cx_single
    print "Foundation stiffness in y: %d kNm/rad" %Cy_single
    ux_tot = x_factor*(ux_single - (Mx_single.subs({z:0}))/Cx_single*z)
    uy_tot = y_factor*(uy_single - (My_single.subs({z:0}))/Cy_single*z)
    Mx_tot = x_factor*Mx_single
    My_tot = y_factor*My_single

    for index in range(len(nodes)):
        m = float(nodes[index])
        ux_subs = float(ux_tot.subs({z:m}))
        uy_subs = float(uy_tot.subs({z:m}))
        Mx_subs = float(Mx_tot.subs({z:m}))
        My_subs = float(My_tot.subs({z:m}))
        Vx_subs = float(Vx_single.subs({z:m}))
        Vy_subs = float(Vy_single.subs({z:m}))
        Vx_wall.append(Vx_subs)
        Vy_wall.append(Vy_subs)
        Mx_wall.append(Mx_subs)
        My_wall.append(My_subs)
        ux_wall.append(ux_subs)
        uy_wall.append(uy_subs)

    #strength, stability, stiffness checks
    ux_max = ux_wall[len(nodes)-1]
    uy_max = uy_wall[len(nodes)-1]
    Vx_max = Vx_wall[0]
    Vy_max = Vy_wall[0]
    Mx_max = Mx_wall[0]
    My_max = My_wall[0]

    Mx_base.append(Mx_max)
```

```
        My_base.append(My_max)

        #checks
        stiffness_check = "pass"
        comp_strength_check = "pass"
        shear_strength_check = "pass"
        stability_check = "pass"

        normal_stress_wall = wall_load[wall]/A[wall]
        bending_stress_wall_x = Mx_max/Wx[wall]
        bending_stress_wall_y = My_max/Wy[wall]

        comp_stress_wall_x = normal_stress_wall +
abs(bending_stress_wall_x)
        comp_stress_wall_y = normal_stress_wall +
abs(bending_stress_wall_y)

        #shear strength
        kx = 1 + math.sqrt(200/(width_x[wall]*1000))
        if kx > 2:
            kx = 2

        ky = 1 + math.sqrt(200/(width_y[wall]*1000))
        if ky > 2:
            ky = 2

        shear_resistance_x =
A[wall]*(0.035*kx**(3/2)*math.sqrt(fck)+0.15*wall_load[wall]/A[wall])
        shear_resistance_y =
A[wall]*(0.035*ky**(3/2)*math.sqrt(fck)+0.15*wall_load[wall]/A[wall])

        #stability
        tension_wall_x = abs(bending_stress_wall_x) - normal_stress_wall
        tension_wall_y = abs(bending_stress_wall_y) - normal_stress_wall

        #perform checks for stiffness and strength
        if abs(ux_max) > ht/500:
            print "Warning! Deflection of Wall %d in x-direction exceeds
height/500" %(wall+1)
            stiffness_check = "fail"

        if abs(uy_max) > ht/500:
            print "Warning! Deflection of Wall %d in y-direction exceeds
height/500" %(wall+1)
            stiffness_check = "fail"

        if comp_stress_wall_x > fck/1.5:
            print "Warning! Compressive stress of Wall %d in x-direction
exceeds material compressive strength" %(wall+1)
            comp_strength_check = "fail"

        if comp_stress_wall_y > fck/1.5:
            print "Warning! Compressive stress of Wall %d in y-direction
exceeds material compressive strength" %(wall+1)
            comp_strength_check = "fail"

        if Vx_max > shear_resistance_x:
            print "Warning! Shear force on Wall %d in x-direction exceeds
material shear strength; shear reinforcement required" %(wall+1)
            shear_strength_check = "fail"
```

```
        if Vy_max > shear_resistance_y:
            print "Warning! Shear force on Wall %d in y-direction exceeds
material shear strength; shear reinforcement required" %(wall+1)
            shear_strength_check = "fail"

        if tension_wall_x > 0:
            print "Warning! There is tension in the foundation of Wall %d
in x-direction" %(wall+1)
            stability_check = "fail"

        if tension_wall_y > 0:
            print "Warning! There is tension in the foundation of Wall %d
in y-direction" %(wall+1)
            stability_check = "fail"


        #change colour of wall if any check fails
        if stiffness_check == "fail":
            wall_col_stiffness_single = "255,0,0 (63)" #red
        else:
            wall_col_stiffness_single = "169,169,169 (63)" #grey

        if comp_strength_check == "fail":
            wall_col_comp_strength_single = "255,0,0 (63)"
        else:
            wall_col_comp_strength_single = "169,169,169 (63)"

        if shear_strength_check == "fail":
            wall_col_shear_strength_single = "255,0,0 (63)"
        else:
            wall_col_shear_strength_single = "169,169,169 (63)"

        if stability_check == "fail":
            wall_col_stability_single = "255,0,0 (63)"
        else:
            wall_col_stability_single = "169,169,169 (63)"

        Vx_total.append(Vx_wall)
        Vy_total.append(Vy_wall)
        Mx_total.append(Mx_wall)
        My_total.append(My_wall)
        ux_total.append(ux_wall)
        uy_total.append(uy_wall)

        wall_col_stiffness.append(wall_col_stiffness_single)
        wall_col_comp_strength.append(wall_col_comp_strength_single)
        wall_col_shear_strength.append(wall_col_shear_strength_single)
        wall_col_stability.append(wall_col_stability_single)


    #set wall colour for no checks
    wall_col = []
    for wall in range(number_of_walls):
        wall_col_single = "169,169,169 (63)"
        wall_col.append(wall_col_single)

    everything.append(Vx_total)
    everything.append(Vy_total)
    everything.append(Mx_total)
    everything.append(My_total)
    everything.append(ux_total)
```

```
    everything.append(uy_total)
    everything.append(wall_load)
    everything.append(wall_col)
    everything.append(wall_col_stiffness)
    everything.append(wall_col_comp_strength)
    everything.append(wall_col_shear_strength)
    everything.append(wall_col_stability)

    #convert nested lists to Grasshopper trees
    analysis = list_to_tree(everything)

    print " "
    print "Forces and displacements solved."
    print 'Analysis time:', time.time() - start, 'seconds'

else:
    everything = []

    Vx = [[0]]
    Vy = [[0]]
    Mx = [[0]]
    My = [[0]]
    ux = [[0]]
    uy = [[0]]
    wall_load = [0]

    #set wall colour when calculator is not running
    wall_col = []
    number_of_walls = len(width_x)
    for wall in range(number_of_walls):
        wall_col_single = "169,169,169 (63)"
        wall_col.append(wall_col_single)

    everything.append(Vx)
    everything.append(Vy)
    everything.append(Mx)
    everything.append(My)
    everything.append(ux)
    everything.append(uy)
    everything.append(wall_load)
    everything.append(wall_col)
    everything.append(wall_col)
    everything.append(wall_col)
    everything.append(wall_col)
    everything.append(wall_col)

    analysis = list_to_tree(everything)

    print "Calculator is not running."
```