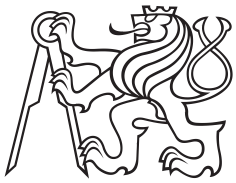
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Control Engineering

Control System for Morphing Lattice–based Structures

Dominik Fischer

Supervisor: Ing. Jiří Zemánek Ph.D.
Study program: Cybernetics and Robotics
May 2023

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Fischer Dominik** Personal ID number: **475380**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Control Engineering**

Study program: **Cybernetics and Robotics**

Branch of study: **Cybernetics and Robotics**

II. Master’s thesis details

Master’s thesis title in English:

Control system for morphing lattice-based structures

Master’s thesis title in Czech:

Řídicí systém pro flexibilní modulární struktury

Guidelines:

The objective of this diploma thesis is to design and develop control systems for morphing lattice-based structures. Specifically, the objectives are to:
1) Review the current state-of-the-art in the field of control systems for voxel-based flexible structures.
2) Design and develop a control system that can achieve desired static and dynamic behavior of the structure.
3) Validate the control system through simulation and experimental testing.
4) Analyze the performance of the control system and make recommendations for improvement.

Bibliography / sources:

[1] A. Abdel-Rahman, C. Cameron, B. Jenett, M. Smith, and N. Gershenfeld, “Self-replicating hierarchical modular robotic swarms,” Commun Eng, vol. 1, no. 1, Art. no. 1, Nov. 2022, doi: 10.1038/s44172-022-00034-3.
[2] B. Jenett, C. Cameron, F. Tourlomousis, A. P. Rubio, M. Ochalek, and N. Gershenfeld, “Discretely assembled mechanical metamaterials,” Science Advances, vol. 6, no. 47, p. eabc9943, Nov. 2020, doi: 10.1126/sciadv.abc9943.

Name and workplace of master’s thesis supervisor:

Ing. Jiří Zemánek, Ph.D. Department of Control Engineering FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **17.02.2023** Deadline for master's thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

Ing. Jiří Zemánek, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt	Student’s signature

Acknowledgements

First and foremost, I would like to express my gratitude to my supervisor, Jiří Zemánek, for the opportunities he has provided me with. Through his guidance, I have had the chance to explore fascinating places and learn invaluable lessons along the way.

I would also like to extend my thanks to the guys in the lab who have made the last two years truly enjoyable. To Loi Do, for introducing me to Bernard Koopman and the exciting world of DMD. Also, for his persistence in kindly reminding me how many minutes were left untill the submission deadline. To Dominik (Homča) Hodan, for being my lab neighbour and for all the Maté he used to give me during the lectures. I would further like to extend my thanks towards Krištof Pučejdl, for all the coffee and his practical and helpful insights regarding *almost* anything.

I am grateful to Madlenka, from whom I learned that even the coolest robot is nothing without a proper name. Thank you for bringing a human touch into the world of artificial.

Last but certainly not least, my deepest appreciation goes to my family. I am forever grateful for receiving the opportunity to study and their unwavering support and encouragement no matter where I have been.

Declaration

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 26. May 2023

Abstract

Metamaterials are materials engineered to exhibit properties that do not normally occur in nature. They usually comprise multiple building elements in repeating patterns and can be tailored to specific applications. This makes them a subject of great interest for numerous applications, including, but not limited to, designing prosthetics, building computers, and assembling space structures. While existing literature has predominantly focused on the structural design and static behaviour of metamaterials and lattice–based materials, their properties hold promise for building dynamic structures, including robots. This thesis centers on the design of control systems for such structures.

This thesis introduces three structures: the Voxel Tower, robot ŽůžO, and Digicomb. They enable the addressing of key steps and considerations in the controller design for lattice–based structures, such as actuation methods and model acquisition process.

The Voxel Tower serves as the focal point of this research, exploring two scenarios in control engineering: stabilisation together with disturbance attenuation and reference tracking. Linear Quadratic Regulator (LQR) and Koopman Model Predictive Control (KMPC) algorithms are employed to address these tasks. The implementation of said control strategies is supplemented with real hardware experiments and simulation results. The remaining two structures serve primarily as test platforms for assessing the suitability of mechanical metamaterials for building dynamic structures. Nevertheless, KMPC for reference tracking for ŽůžO is also designed and its performance is validated through simulations.

Keywords: metamaterials, lattice structures, digital materials, optimal control, feedback control, data–driven modelling

Supervisor: Ing. Jiří Zemánek Ph.D.
Katedra řídící techniky ČVUT,
Karlovo náměstí 13/E,
Praha 2

Abstrakt

Metamateriály jsou uměle navržené materiály vykazující vlastnosti, které se v přírodě běžně nevyskytují. Ve většině případů vznikají sestavením mnoha základních elementů do pravidelných vzorů a lze je přizpůsobit konkrétní úloze. Díky tomu jsou zajímavé pro řadu oblastí, mezi které patří návrh protéz, sestavování počítačů a stavba vesmírných struktur. Dosavadní výzkum se soustředil především na návrh struktury a studium statického chování metamateriálů a mřížkových materiálů, nicméně díky svým vlastnostem by mohly najít uplatnění i při stavbě dynamických struktur, včetně robotů. V této práci se zaměříme na návrh řídicích systémů právě pro tyto struktury.

V této práci představíme tři struktury: Voxelovou věž, robota ŽůžO a Digicomb. Na nich ukážeme řešení klíčových kroků při návrhu řídicího systému právě pro mřížkové struktury a popíšeme skutečnosti, které je potřeba při návrhu zohlednit. Mezi tato témata patří vyřešení způsobu aktuace mřížkových struktur a jejich modelování.

Jádro této práce tvoří Voxelová věž, na které budeme zkoumat dva problémy z oblasti automatického řízení: stabilizaci spolu s odmítáním poruch a sledování reference. K řešení těchto úloh použijeme lineárně kvadratický regulátor (LQR) a Koopmanův prediktivní regulátor (KMPC), jejichž implementaci doplníme reálnými experimenty. Přestože zbylé dvě struktury slouží především k otestování, zda jsou mechamické metamateriály vhodné pro stavbu dynamických struktur, navrhneme a simulačně ověříme KMPC algoritmus pro sledování trajektorie také pro ŽůžA.

Klíčová slova: metamateriály, mřížkové struktury, digitální materiály, optimální řízení, zpětnovazební řízení, na datech založené modelování

Překlad názvu: Řídicí systém pro flexibilní modulární struktury

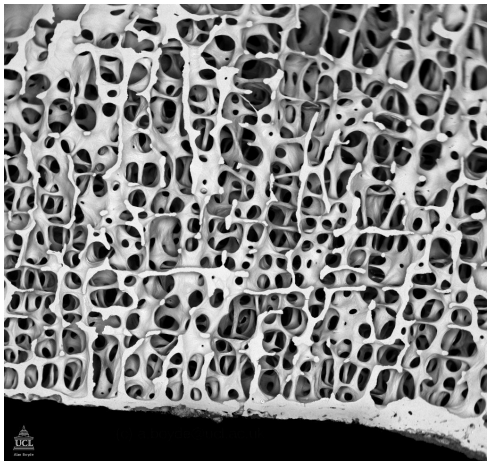
Contents

Chapter 1

Introduction

The presence of elements with lattice-based architecture can be observed in various natural structures. Robert Hook's observation in 1655 highlighted the lattice structure found in different types of wood, including cork [Hoo65]. Similarly, the internal structure of bones also exhibits a lattice arrangement (Fig. 1.1a). Lattice-based materials have a significant merit: a high stiffness-to-weight ratio in comparison to monolithic structures [Ash05]. This appealing characteristic makes them particularly interesting for space applications, where utilizing less material not only reduces the weight of structures but also facilitates transportation. Lattices have captured the interest of researchers across multiple domains, including robotics, civil engineering, and architecture. As a proof of that, the renowned Eiffel Tower stands as a testament to the potential of lattice structures. The periodic nature of the underlying structure of a material aligns with the concept of material digitalisation. It also opens up significant possibilities for developing automated assembly systems for such materials. Ongoing research in this area is being conducted, for instance, by the Center for Bits and Atoms (CBA) group at the Massachusetts Institute of Technology (MIT), USA.In this introductory chapter, we describe the fundamental concepts utilised in this work, including advantages and applications of mechanical metamaterials, laying the groundwork for the designing control algorithms for morphing lattice–based structures.

To understand the idea of digitalising physical materials, consider this perspective. While 3D printers and CNC cutters driven by digital computers are commonly used for fabrication, they still rely on analog materials prone to errors. Consequently, even identical parts printed on the same 3D printer can differ slightly. To address this issue, we can introduce the notion of 'discrete' or 'digital' directly into the material. By assembling discrete parts to form the



(a) : Electron microscop scan of a normal human bone architecture (Picture by Alan Boyde from the Bone Research Society).

(b) : Eiffel Tower in Paris.

Figure 1.1: Lattice structures in nature and around the world.

structure, the random errors are mitigated. Having discrete blocks instead of a continuous structure offers also other advantages. Firstly, it simplifies the process of making changes or repairs to the system since malfunctioning blocks can be easily replaced. Additionally, this approach enables the assembly of large, heterogeneous structures by combining individual blocks with (possibly) different properties. To draw an analogy to a well–known example, the concept of 'digital materials' shares similarities with the principle of constructing objects using LEGO blocks.

Metamaterials are engineered materials with unique properties not found in natural materials. For example, so called negative–index metamaterials possess unusual optical properties, allowing them to exhibit a negative refractive index. This means that they can refract light in the opposite direction compared to conventional materials, allowing for unconventional devices such as superlenses and cloaking devices to be explored. Metamaterials are created by arranging elements in repeating patterns, making them an exemplary case of digital materials. An example of that is given by Benjamin Jenett in his PhD thesis, where he presents a collection of discrete building elements that embody the defining features of metamaterials. They posses distinct physical properties and when assembled, these blocks demonstrate characteristics such as a negative Poisson's ratio, where the material shrinks perpendicular to the applied force rather than expanding [Jen20].

The inherent discrete nature of digital materials enables the exploration of their automated assembly methods. Nature can serve as a source of inspiration, where collective efforts of organisms like ants or bees demonstrate

collaborative completion of tasks. Our body iteslf is made up of 20 amino acids constituting the building blocks of proteins. [WPN14] present a termite–inspired team of small robots capable of cooperative construction, communication, and consensus-building for structure assembly. A different automatic assembly system tailored for building lattice based structures is presented in [JC17] and [JARCG19]. The authors utilise inchworm archetype robots with grippers for attachment, traversal, and material placement. [TM08] proposes a similar principle of robot attachment and transportation of building modules. [vMGM+23] presents an alternative approach. Instead of relying on robots to assemble the structure, the authors introduce a method for automated folding of carefully prepared material into origami lattices.

Prior to the assembly, the design of the resulting structure of the material needs to be established. Creating metamaterials involves considering multiple criteria and determining the optimal topology of individual building elements to achieve desired properties. [TDZK22] addresses this challenge by employing clustering and topology optimization strategies in the design of a modular beam. Another approach, demonstrated in [CTdR+16], utilizes a combinatorial strategy to design even aperiodic metamaterials. Moving to the study of static properties, three structures, namely a bridge, a boat, and a shelter built from lattice–based materials, under static load are studied in [JCGC16]. Similar study of static characteristics of mechanical metamaterials is also described in [CG13].

Modularity, a property often employed in reconfigurable structures, is facilitated by materials that can be automatically assembled from discrete building blocks. One advantage of modular systems is their simplified repair process, as malfunctioning modules can be directly replaced. Additionally, reconfigurable systems offer the ability to perform different tasks using the same set of parts, but in different configurations, eliminating the need for multiple dedicated systems. Achieving various configurations can be done through self-reconfiguration, as seen in self-reconfigurable robots [MK07] like PolyBot [YDR00] and SUPERBOT [SMS06]. Alternatively, external interference can be employed, making the process of changing configuration similar to the automated (dis)assembly of the structure.

Digital metamaterials, possessing the properties of modularity, and capability of being automatically assembled, and tailored to specific use cases, offer intriguing possibilities across various applications. They find relevance in fields such as soft robotics, production of wearable devices, prosthetics, and fashion industry [MJS+18]. Additionally, research is being conducted on the assembly of electronic devices [LGG16], potentially enabling optimised computational devices tailored for specific problem-solving tasks. The ap-

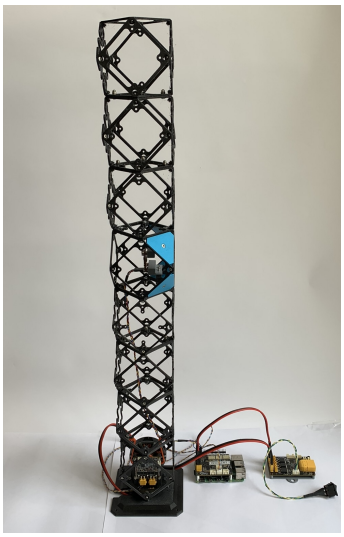
plications of digital metamaterials extend even to the construction of space structures [JC17], showcasing their diverse potential.

For the sake of clarity it is important to shed more light into the terminology describing the material used in this work. Terms such as metamaterial, lattice–based material, modular material, and digital material are general terms that may encompass a range of different materials and can sometimes overlap in their definitions. The materials studied in this work embody all of these concepts and exhibit properties common to all four definitions. Therefore, the names may be used interchangeably depending on the specific focus and the property being emphasized. Additionally, in this text, some of the individual discrete building blocks will be referred to as *voxels*, which can be understood as a volumetric version of a 2D pixel.

In the previous discussions, we have highlighted the appealing properties and potential applications of lattice-structured metamaterials, primarily focusing on static structures like advanced computers or space stations. However, less attention has been given to the challenges of building robots and actuated dynamical systems using digital materials, which is the main focus of this work.

In this text, we introduce a mobile robot and two other dynamical structures constructed from mechanical metamaterials. We address a range of tasks associated with their development and discuss specific steps involved in implementing control systems, including selecting appropriate modeling frameworks and designing efficient actuation mechanisms. Our goal in this thesis is not to develop a single unified control approach for all lattice structures. Instead, we individually study the three structures depicted in Figures 1.2, 1.3, and 1.4. From our observations, we try to derive generalized insights that can facilitate the future development of a unifying framework.

The structure of this text is as follows. Chapter 2 provides a summary of the current state of the art in the field of controlled flexible and metamaterial structures. In Chapter 3, we introduce the types of metamaterials used for building the structures described in this work. Chapter 4 then describes three possible modelling approaches that can be used to model digitial materials and discusses their applicability to the problem at hand. Chapters 5, 6 and 7 focus on the three presented structures: the Voxel Tower, the ŽůžO robot, and Digicomb, respectively. The Chapter on the Voxel Tower is the most elaborate as it complements the design of control algorithms with real experiments. The last structure in Chapter 7 is presented mostly for completeness and to showcase also different type of mechanical metamaterial used for building dynamical structures. Finally, Chapter 8 concludes the text by summarising the main results and discussing potential avenues for future work.

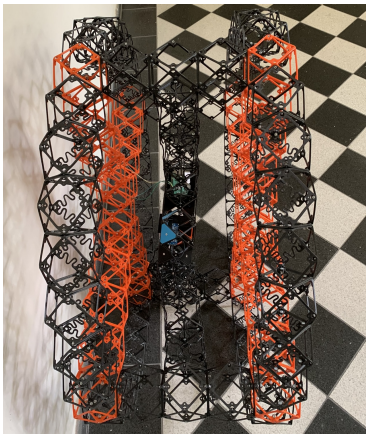


(a) : Front view.

(b) : Side view.

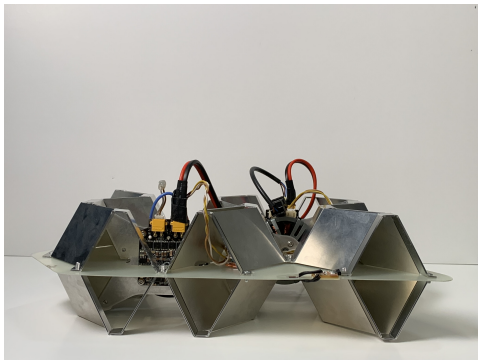
Figure 1.2: Voxel Tower.

(a) : Front view.

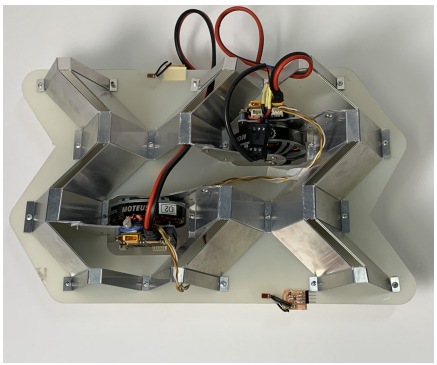


(b) : Side view.

Figure 1.3: Robot ŽůžO.



(a) : Side view.



(b) : Top view.

Figure 1.4: Digicomb.

Chapter 2

Related Work

Despite the concept of metamaterials and lattice structures not being entirely new, the literature on active control of such structures is still limited. As mentioned earlier in Chapter 1, researchers in the field of metamaterials have shown interest it two main areas. Firstly, there has been a focus on the structural design of these materials to achieve desired properties, such as directional compliance. Examples of such studies include [CTdR+16, SSP+19, TDZK22]. Secondly, there has been considerable attention given to investigating the static behaviour and properties of metamaterials [JCGC16].

Nevertheless, some research on active control of metamaterials exists. In a recent paper [PRFJ+23], elements of active control are introduced into structures composed of similar building blocks as those utilised in this thesis. The authors formulate an optimization problem to achieve nature–inspired motion in a snake–like swimming robot and also present foundations for a controlled morphing wing. Morphing and flexible structures receive considerable attention in the field of aerospace [BBA+11]. For instance, in [GHC+21] feedback linearisation control is designed for a morphing missile.

However, not only mechanical metamaterials (like in [PRFJ+23]) are being studied, as there is also interest e.g. in acoustic metamaterials. Usage of these materials is motivated by the presence of band gaps, which are frequency ranges with high levels of sound wave attenuation. In [CDM17], it is shown that the narrow band gap of an array of Helmholtz resonators can be extended by incorporating feedforward control through embedded loudspeakers. Another approach to creating a new band gap using proportional feedback control, this time using piezoelectric actuators, is described in [CZL20].

While not labelled the same, several problems studied in the field of control of metamaterials share similarities with tasks encountered in other well–established domains. Among those problems is the task of vibration control [Pre18], extensively explored in the context of flexible structures. In [HTD06], an active vibration control strategy for a flexible beam utilising genetic algorithms is presented and validated in simulations. A conventional approach to damping is presented in [Bal78], where state feedback control is employed and experimentally validated using a real beam. Additionally, [Mei13] investigates vibration control in complex planar structures, such as L–shaped and portal frames. The mitigation of oscillations is not limited to artificially manufactured scenarios and arises in numerous real–world situations. For example, [SSS90] presents strategies for controlling structures subjected to seismic excitation, and [PG07] provides a comprehensive description of feedforward and integral force feedback control of lattice–structured satellite boom.

Over the years, flexible structures have received significant research attention, also leading to the application of more advanced control strategies in their domain. For instance, optimal control techniques for flexible structures were of significant interest, particularly towards the end of the 20th century [GN81, SKM89]. Notably, research in [PF90] focused on time–optimal bang-bang control for achieving rest–to–rest motions of flexible structures. Despite the passage of time, the interest in controlling flexible structures remains strong, and state–of–the–art control techniques continue to be employed. For instance, in [AWP20], nonlinear Model Predictive Control (MPC) is applied to a multibody system consisting of deformable beams.

To put this work into the context of current research, we outline our main goals and their alignment with the state of the art in controlled mechanical meatamaterials. Our primary focus is on structures akin to those presented in [PRFJ$^+$23]. These structures have so far only been subjected to feedforward control and precomputed input sequences. In this thesis, we aim to advance further and to develop feedback and optimal control algorithms for these structures. This will facilitate the solution of more complex tasks and expand the potential applications of lattice–based structures.

Chapter 3

Digital Materials

One of the main advantages of mechanical metamaterials is their versatility. Just through different combinations of the individual building blocks, the resulting structures can be tailored to various applications. Digital materials can be used for building large static structures, such as a bridge or a shelter [JCGC16] as well as for assembling electronic circuits [LGG16]. It all comes down to designing the right building blocks for the task at hand. In this chapter we describe the digital material utilised in this work. The building blocks were inspired by discrete material systems developed by the Center for Bits and Atoms at MIT and can be divided into two groups. The first group is derived from a cellular material published in [JCGC16, JCT+20, Jen20]. It consists of 3D printed faces that can be connected to form voxels with various physical properties (rigid, compliant, chiral and auxetic). The second type of metamaterial comprises foldable hylite (an aluminium composite with polypropylene core and aluminium cover sheets) plates that can form much more durable structures.

In this chapter, we provide an overview of the individual building elements used in this work. We start by describing the rigid voxel in Section 3.1, followed by the introduction of semi-compliant and actuated voxels in Sections 3.2 and 3.3, respectively. Additionally, we dedicate Section 3.4 to the description of the second group of digital material – the *Discrete Folded Material*.

During the intial stages of our work, we experimented with the original voxel–based material from CBA. However, we found that certain modifications to the building blocks were necessary to better align the material with our

requirements. These modifications concerning the rigid and compliant faces and their details are described in their respective sections.

Regarding the fabrication of individual voxel faces, we 3D print them from PETG filament utilising an off-the-shelf MK3S+[1] Průša printer. Using the GSWRP (graduate student with rivet pliers) method, individual faces can be connected to constitute a cuboctahedral voxel, described by two characteristic dimensions: lattice pitch P and strut length L (Fig. 3.1a). Resulting voxels are then screwed together along their faces to form larger structures (Fig. 3.1c).

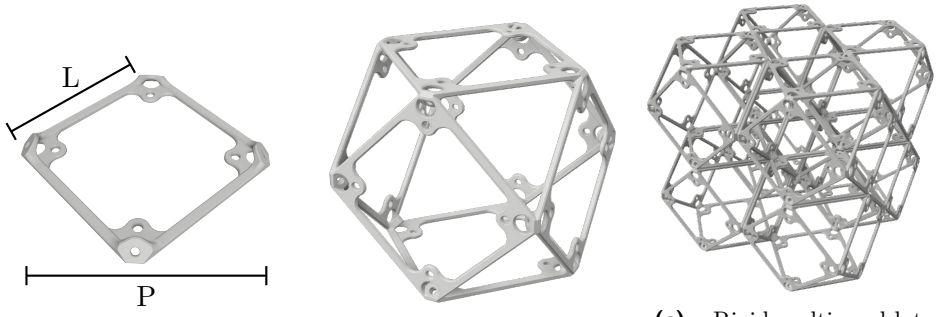
A comprehensive analysis of mechanical properties of voxels is beyond the scope of this work and extensive studies on this topic have been documented in other publications. We will provide a brief overview, highlighting the main differences between individual voxels and their respective applications. For a more in–depth examination of properties of the metamaterials used, we recommend referring to sources such as [JCT+20] or [Jen20].

3.1 Rigid Voxel

The rigid voxel, depicted in Fig.3.1b, consists of six rigid faces (Fig. 3.1a) and is designed for applications requiring structural integrity and minimal deformation under external loads. The rigid face from [Jen20] differs from the one used in this work only in size. To achieve compatibility with other hardware, such as the BLDC motor used for actuation, we scaled up the dimensions of the face. The modified face now has lattice pitch of $P = 9$ cm and strut length of $L = 6$ cm, compared to the original dimensions of $P_o = 7$ cm and $L_o = 5$ cm.

In the structures investigated in this thesis, the rigid voxel serves two primary purposes. Firstly, it functions as a housing for sensors and various components, including batteries and microcontrollers. Secondly, it acts as reinforcements, enhancing the stiffness of the structures. An example of both is the application of the rigid voxel in constructing ŽůžO (Chapter 6).

[1] https://www.prusa3d.com/product/original-prusa-i3-mk3s-kit-3/

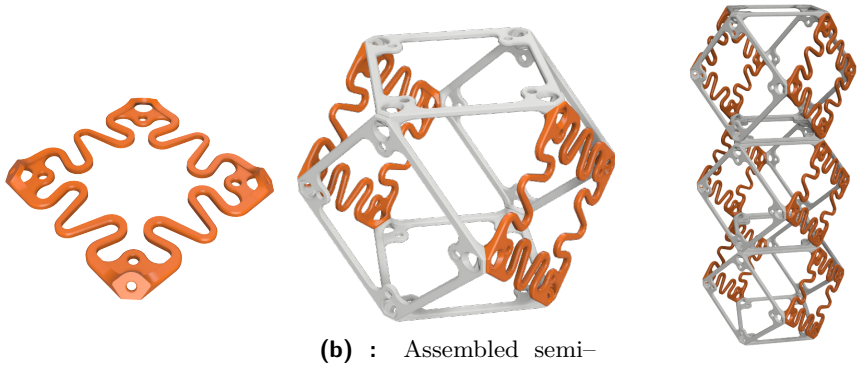


(a) : A single rigid face.

(b) : Rigid voxel.

(c) : Rigid multivoxel lattice.

Figure 3.1: The decomposition of a rigid voxel lattice.



(a) : A single compliant face.

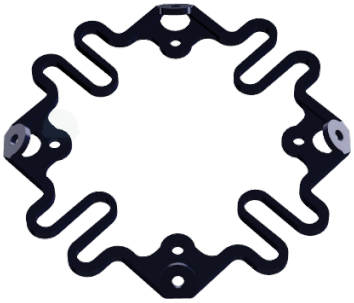
(b) : Assembled semi–compliant voxel.

(c) : Semi–compliant multivoxel beam.

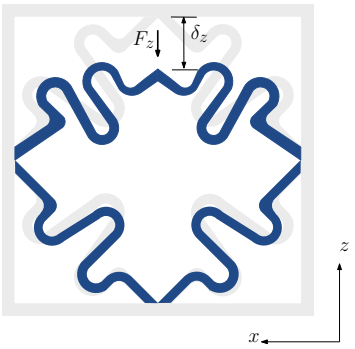
Figure 3.2: The decomposition and use case of a semi–compliant voxel.

3.2 Semi–compliant Voxel

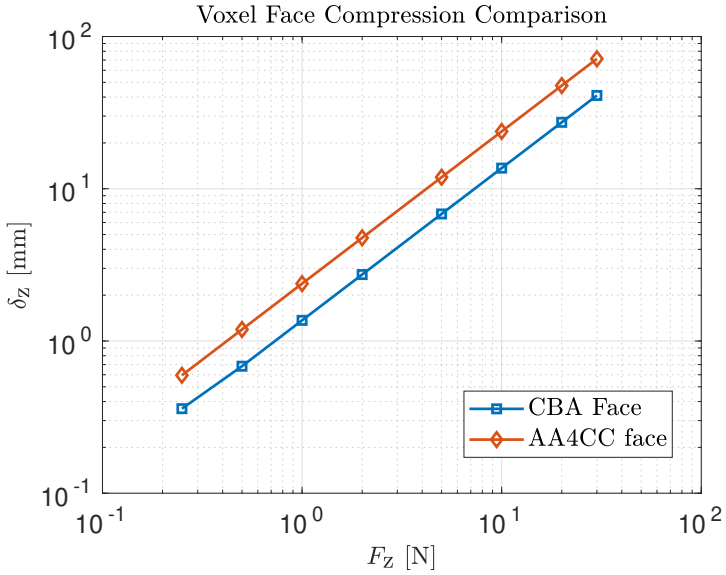
The semi-compliant voxel (Fig. 3.2b) is composed of four rigid and two compliant faces. While multiple configurations of compliant and rigid faces are possible, resulting in various voxels, we detail the most prevalent configuration found in our structures. Our particular arrangement of the compliant faces allows the voxel to maintain stiffness in compression, while permitting bending along one axis. This behavior is achieved through the coordinated action of the compliant faces: when one face is compressed, the other extends. It can be utilised in applications, where planar bending is desired, such as the Voxel Tower, described in Chapter 5. Furthermore, the semi–compliant voxel constitutes the backbone of the actuated voxel, introduced in the following Section 3.3.



(a) : Original compliant face used at CBA [Jen20].



(b) : Graphical illustration of the measured deformation.



(c) : Comparison of the compliant faces under compression.

Figure 3.3: Testing of the original compliant face.

The design of the compliant face shown in Fig. 3.2a was modified from the original presented in [Jen20] (Fig. 3.3a). This redesign is credited to Krištof Pučejdl[2], member of the AA4CC group at CTU and the main motivation behind it was to address the limited compression and deformation capabilities of the original face. To showcase the improved compression ability, we conducted a series of finite element analysis simulations in Fusion 360, vertically compressing the faces and measuring the induced displacement (Fig. 3.3b). The results in Fig. 3.3c demonstrate that the new compliant face allows nearly twice as much displacement compared to the original face. Moreover, it maintains the linear relationship between the applied force F_z and the induced displacement δ_z.

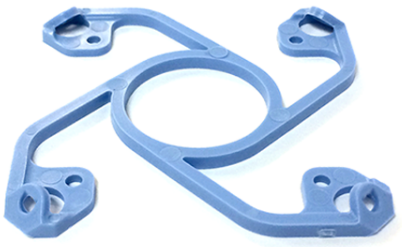
[2]http://aa4cc.dce.fel.cvut.cz/users/ 4pxtdnlfpf

3.3 Actuated Voxel

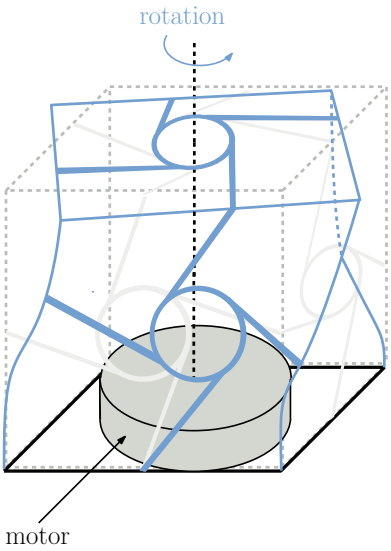
Various methods of actuation for metamaterials were studied in literature. Piezoelectric actuators in the form of patches attached to the individual metamaterial cells were employed in [LMWW20]. The viability of tendon-based actuation was studied e.g. in [PRFJ+23] and it showed to be a reasonable choice even when considering larger lattice structures, such as a snake–like beam. The tendons are attached on one end to a servo motor and on the other end to the point where the transmitted force should act. Thanks to being a straightforward method in principle, tendon–based actuation has achieved a considerable attention throughout the years [JKID90, OKH14], especially in the field of soft robotics [VCG+16], where tendons are used for solving tasks such as stiffness control [LCCY94]. Furthermore, [Jen20, PRFJ+23] successfully translated this method into the field of actuation and control of mechanical metamaterials.

Initially, we also relied on tendons for actuating voxel–based structures. This choice was a natural first step, as it followed the inspiration from the CBA group and the method was relatively easy to implement. However, there are a several flaws with this approach. Foremost, tendon–based actuation does not fit into the philosophy of digital materials. The essence of digital materials lies in their modularity and reconfigurability. The characteristics of individual modules should arise through entirely self–contained properties such as compliance, the ability to gather information, or the capacity to initiate motion, either individually or in combination. By combining such modules we can build structures with desired specifications. For instance, the need for increased driving force can be addressed by serially connecting multiple actuation modules rather than designing a new component. Additionally, the installation of tendons spanning multiple voxels would be significantly more challenging to automate compared to connecting a single block. These factors motivated us to develop a self-contained method of actuation that can be fully integrated within a single voxel

We will call the voxel containing the actuation simply as *actuated voxel*. It denotes a module capable of self–deformation, thereby inducing motion in the overall structure. When designing the actuated voxel, the key consideration is determining the specific shape change required to achieve the desired motion of the entire structure. This opens up various possibilities for realising the actuated voxel. For instance, a module generating rotational motion (Fig. 3.4) can be constructed using chiral faces from [Jen20]. Chiral faces exhibit twisting in response to axial strain, making them suitable for this application. A standard motor connected to the botom of the voxel could be utilised to initiate rotation. Individual chiral faces would not oppose the rotation while



(a) : The chiral voxel face [JCT+20].



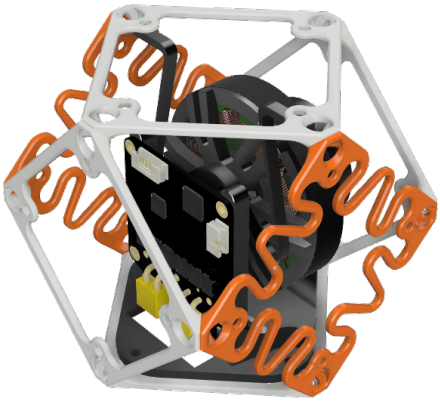
(b) : Chiral–based actuation principle.

Figure 3.4: The illustration of an rotating actuated voxel.

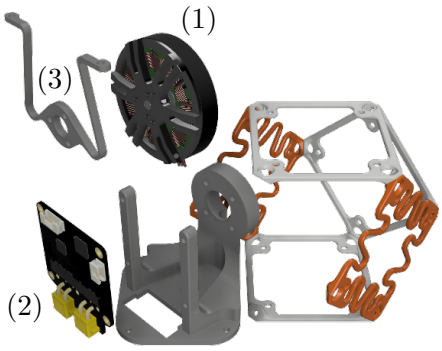
still providing enough structural support. However, our particular application does not require rotational or twisting motion. Instead, we found that a voxel capable of tilting the upper face by stretching and contracting its sides aligns better with our needs. We have physically implemented this actuation principle, resulting in the specific type of actuated voxel described below.

Our specific actuated voxel comprises three main components: a 5208[3] brushless motor (1), an FOC Moteus Driver r4.11[4] (2), and a mechanical motion transmitter (3) (Fig. 3.5b). The motor was selected for its compatibility with the driver, which integrates a magnetic encoder for precise motor control, microcontroller, and CAN-FD interface. Unlike traditional servo or stepper motors, this combination enables us to operate the motor in torque control mode. The mechanical motion transmitter consists of two 3D printed slim compliant beams connecting the motor shaft to the top face of the semi–compliant voxel. The transmitter converts the rotation of the motor into push–pull motion acting on the top face of the voxel. This results in tilting the voxel as one compliant face is compressed and the other extended. We illustrate the motion principle in Figure 3.5d, which shows simulation results of a simplified voxel model under static applied torque. The simulation results, obtained using Fusion 360 software, also show stress distribution and the corresponding deformation. An important remark is that this method of actuation deforms the housing voxel itslef, distinguishing it from methods often used e.g. in modular robotics. There, movement is usually achieved through separate actuated joints that connect individual

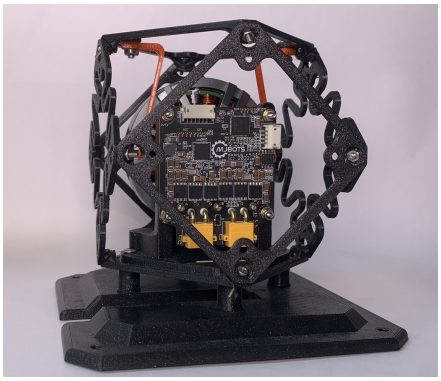
[3]https://mjbots.com/products/mj5208
[4]https://mjbots.com/products/moteus-r4-11



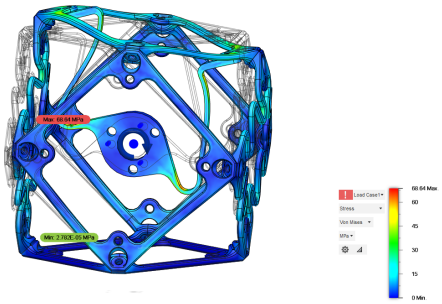
(a) : Render of the actuated voxel.



(b) : Decomposition into individual parts.



(c) : Real assembled prototype.



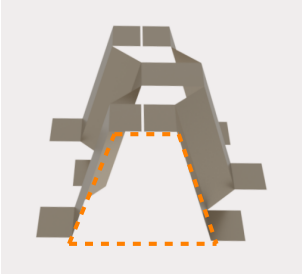
(d) : Simulation results of the actuated voxel under applied torque.

Figure 3.5: The actuated voxel utilised in this work.

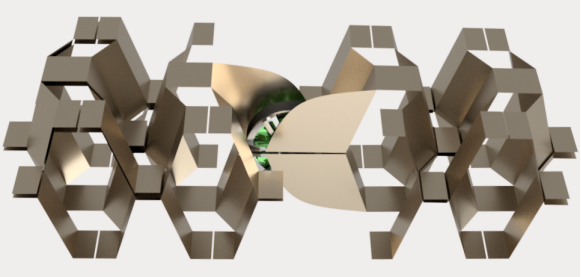
modules. In contrast, our actuated voxel is equal with other building blocks when viewed in the context of the whole structure. Meaning there is no need for any other further interface and the movement directly arises through the deformation of the modules.

3.4 Discrete Folded Material

The relatively high fragility of the aforedescribed voxels was a concern as we have spent quite some time with replacing broken voxel faces. Therefore, we describe also an alternative, much more durable kind of metamaterial: the Discretely Folded Material (DFM). It differs from the voxels above both in material and assembly. Voxels were 3D printed and then the individual faces were connected together using rivets. In contrast, the DFM consists of cleverly cut out parts of hylite that can be folded along predefined lines to form 3D structures. This material is credited to Alfonso Parra Rubio at the Center for Bits and Atoms, MIT. Because with DFM the researchers at CBA



(a) : A single discrete folded building block.



(b) : Example of a discrete folded structure.

Figure 3.6: Discrete folded material.

pursue building structures like morphing aircraft wing, its high load bearing capacity is of great importance. Besides that, another perk of this material is that we can assemble large and durable 3D structures from building blocks produced quite simply using ordinary 2D fabrication methods. Furthermore, the stiffness of the resulting structure can be easily adjusted by configuring individual building elements.

Regarding the components, there is basically just one type and when folded, it resembles a hollow zig–zag triangular prism (see Fig. 3.6a for better intuition). These compose structures by being attached to a flexible centroid (Fig. 1.4). One interesting property of the centroid is that it retains constant length when bending and the modules are designed to be compliant with that. While having a single governing shape, the parameters of the elements are tunable to achieve various properties. The stiffness of the whole structure is governed by the shape of the trapezoidal side profile of the building blocks (orange dashed line in Fig. 3.6a): the steeper the trapezoid, the more flexible the structure is. For a better idea of how a strucutre from this material can look like, see the Figure 3.6b. The structure in the figure acts as a joint, with an actuator in the middle that can rotate with the attached left and right parts. The individual building blocks are then mounted onto the centroid (the centroid is not visualised) in opposite orientations giving the structure a regular and symmetric 3D shape.

Chapter 4

Approaches to Mathematical Modelling and Simulation

The procedure of designing almost any control algorithm cannot do without a mathematical model. This model serves not only for the design process itslef but also for validating the resulting controller. We dedicate this chapter to describing the modelling frameworks and simulators employed in this work.

In general, three simulation and modelling approaches are possible, each carying its benefits and drawbacks. The first approach is to utilise some existing simulation tool. In the context of digital materials, a correctly chosen software would allow for describing the discrete nature of the material and provide a suitable environment for simulating the coupled dynamics of individual blocks. At the same time, this approach is limited by the capabilities of the chosen simulator and adding or modifying functionalities as required might prove difficult. Another approach is to abandon the discrete nature of the material and instead create a continuous approximation that captures the essential properties of the system. This approach is appealing due to its reliance on well–established physical principles and the ability to specify the desired level of detail during the design process. However, by adopting this approach, we sacrifice the discrete nature of the system, leading to a substantial simplification of the dynamics and potential loss of certain degrees of freedom. The third approach involves utilising a fully equation–free data–driven method to construct the model. The motivation behind this approach is following. Firstly, there are cases where the underlying dynamics are too complex to be accurately captured by equations. Secondly, utilizing correct data has the potential to reveal previously unknown relationships. However, it is important to note that the absence of equations and limited

prior knowledge of the system can sometimes impact the precision of the model.

First, in Section 4.1, we provide an overview of two existing simulation tools: one developed at MIT and the other here, at CTU. Then, in Section 4.2, we discuss the process of creating a model using conventional techniques at the price of abandoning the discrete nature of the material. Finally, in Section 4.3, we introduce an universal data-driven model acquisition framework called the *Dynamic Mode Decomposition* (DMD).

4.1 Existing Simulators

When it comes to deciding what tools can be used for simulating particular objects, often the idea of using a simulation software tailored to the studied structure comes to mind first. In this section, we describe two such simulators. Namely, the *Metavoxels* simulator and the *Rigid Body* simulator.

The Julia–based Metavoxels simulation and modelling software, developed by Amira Abdel–Rahman[1] at CBA, is a tool for designing and simulating voxel–based lattice structures. It utilises a node and edge representation, where nodes correspond to voxel connection points and edges represent the trusses of individual voxels. When simulating structures with a large number of elements the simulator allows for hierarchical simplification by grouping multiple nodes and edges into a single node, reducing the computational complexity of the simulation.

The core of the simulator resembles the *direct stiffness method* and utilises stiffness relations between individual elements to compute the resulting forces and displacements. Individual trusses, each defined by a pair of vertices, are modelled using the following *beam model*

$$\begin{bmatrix} F_n \\ M_n \end{bmatrix} = K \begin{bmatrix} D_n \\ \Theta_n \end{bmatrix}, \tag{4.1}$$

where F_n, M_n, D_n and Θ_n are nodal forces, moments, displacements, and roatations, respectively. K is here the stiffness matrix of a given edge/beam.

The Metavoxels framework offers potential for creating precise and computationally tractable models and simulations. It considers a wide range of

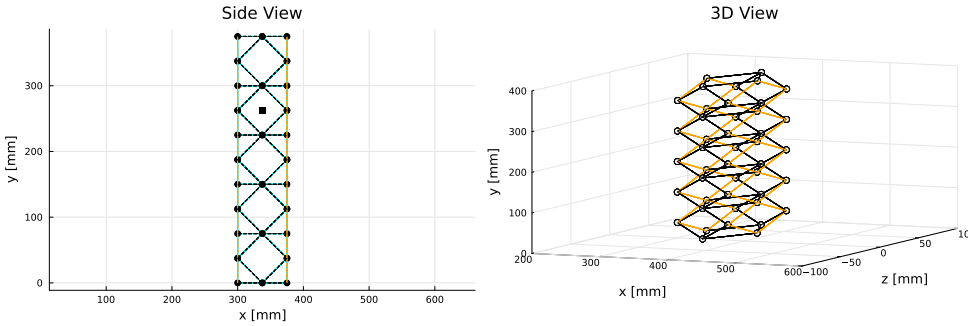
[1] https://amiraa.pages.cba.mit.edu/home/

material parameters, including coefficients of thermal expansion, Poisson's ratio, and various damping constants. However, this high level of customisability presents challenges when building models without precise knowledge of all these parameters and their role in the simulation. Moreover, not all parameters have clear physical interpretations, and their values were tuned to best fit the voxels and structures developed at CBA. Furthermore, it is important to note that the simulator is still under development, as Amira herself mentioned. Additionally, the simulator is best suited for voxel structures actuated by tendons, which is the primary means of actuation at CBA. While the Metavoxels simulator is an interesting tool, it can be challenging to utilise it effectively for creating simulations that facilitate the design of control algorithms.

Nevertheless, we tested the simulator by creating a downsized model of the Voxel Tower (Section 5). We kept some material parameters unchanged, while other we experimentally tuned to achieve a visual resemblance of the model to the real Tower. To enable deflection angle approximation, we integrated a new sensor interface into the simulator. To estimate the deflection, the sensing model utilises the relation $\phi = \text{atan}(y, x)$ with ϕ [deg] being the deflection angle and x [mm] and y [mm] the coordinates of the sensor. Additionally, we implemented a new actuation principle resembling the actuated voxel. Figure 4.1 displays the graphical output of the simulator, featuring 2D and 3D views of the Voxel Tower model. To validate the implemented changes, we experimentally tuned a straightforward velocity P damper $u = 3\dot{\phi}$, where u is the action applied to the nodes where the actuated voxel is acting. The response of the damped system is shown in the Figure 4.1c. Although the performance of the damper falls short of being optimal or satisfactory, the purpose of it was to test the integration of all the implemented features. However, due to the aforedescribed challenges, we made the decision not to continue investing in the framework. At the time of writing, the created model was not sufficiently precise or suitable for our needs.

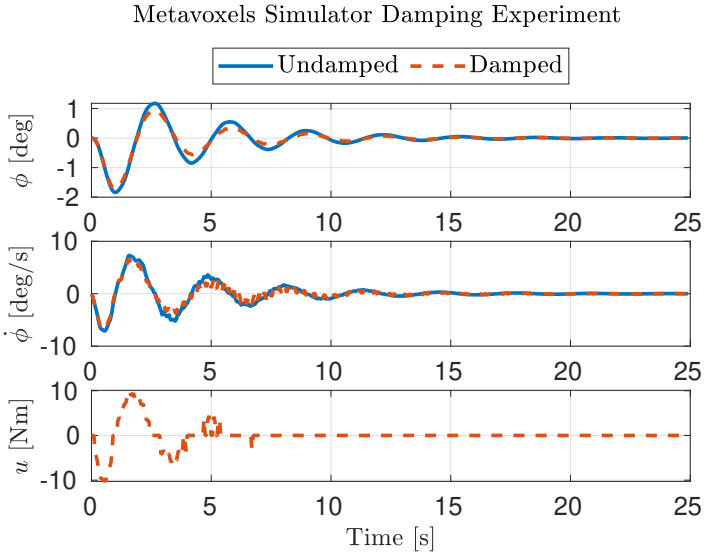
The Rigid Body simulator utilises the RigidBodyDynamics[2] library written in Julia to create models. This simulator was implemented by Dominik Hodan during the Team Project course at CTU as a tool to simulate robot ŽůžO (Chapter 6). The structures are modelled as a smart interconnection of planar and revolute joints with forces acting on each of the joints. The advantages of this approach are again, the possibility of expressing the discrete nature of the system with a lot of degrees of freedom. However, being the same type of simulator as the Metavoxel simulator carries the burden of having similar downsides. Namely, that some of the parameters are difficult to tune right and it can be challenging to model structures for which the library does not have an immediate support. For example, the library is not meant for modelling closed kinematic chains with floating base. To model ŽůžO,

[2] https://juliarobotics.org/RigidBodyDynamics.jl/stable/



(a) : Side view of the created Voxel Tower. Black square in the fourth voxel represents the sensor.

(b) : 3D view of the Voxel Tower model. Orange are compliant and black are rigid sides.



(c) : Response of the damped system to an intial ramp disturbance to the top of the tower.

Figure 4.1: The Metavoxels simulator.

Dominik had to devise a workaround by introducing virtual joints with no mass to enable the library to solve the robot's dynamics. Given these reasons and the fact that we are now working with a new version of the robot, for which the model was not created, we include the description of the Rigid Body simulator just for the sake of completeness to document its existence. For illustration, the visualisation of ŽůžO is in the Figure 4.2.

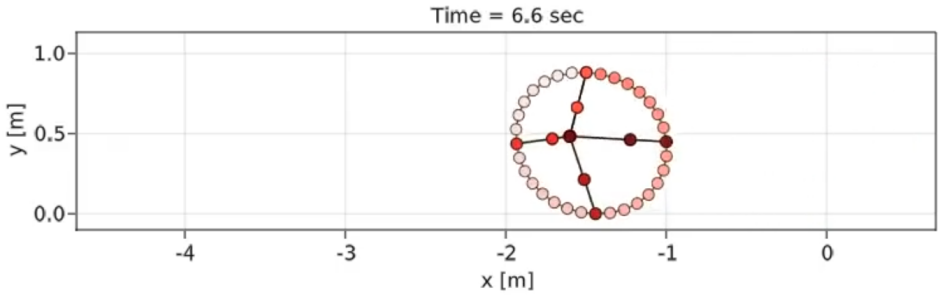


Figure 4.2: Graphical output of the Rigid Body simulator.

4.2 Analytical Approach

The second approach to modelling involves utilising the knowledge of the underlying physics that governs the structure. We demonstrate this analytical modelling procedure on the robot ŽůžO. It is a rolling robot constructed using digital material, achieving rolling motion through the displacement of its heavy central block connected to the outer wheel rim via flexible segments. This way ŽůžO shifts its center of mass, enabling the rolling motion. Figure 1.3 depicts the complete robot.

Due to the discrete nature of ŽůžO's building material, deriving differential equations for each degree of freedom becomes highly challenging, if not impossible. To facilitate the modelling process, we will adopt the following approximations and assumptions. Firstly, we will transition from discrete building blocks to a continuous matter representation. Additionally, we will consider the rim to be perfectly rigid. Furthermore, our focus will be solely on the fundamental aspects of the robot, simplifying it to a point of mass connected to the rigid ring via four springs. These simplifications lead to the creation of a simplified model, as depicted in Figure 4.3.

The mathematical model was developed using Lagrangian formalism. This constitutes describing the system in terms of its kinetic and potential energies, T and V. They are then used to determine a smooth function L known as the *Lagrangian*.

$$L(q, \dot{q}) = T(q, \dot{q}) - V(q, \dot{q}). \tag{4.2}$$

First step in describing the robot is to determine its degrees of freedom and generalised coordinates q that can uniquely describe the state of the robot at

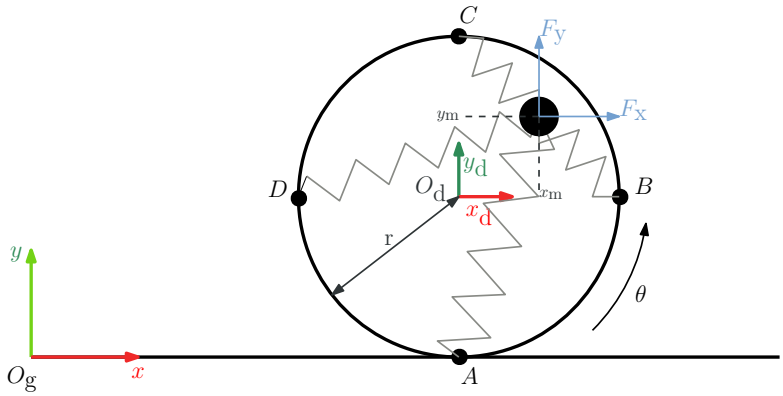


Figure 4.3: Simplified ŽůžO model.

any given time. Our choice of the generalised coordinates is

$$q = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top = \begin{bmatrix} \theta & x_\mathrm{m} & y_\mathrm{m} \end{bmatrix}^\top,$$

where θ [rad] is the current angle of the wheel and x_m [m] and y_m are [m] coordinates of the mass with respect to the center of the disc (expressed in the coordinate system O_d). The transition from the local disc–connected frame O_d and the global frame O_g is possible using the classical 2D rotation matrix $R(\theta) \in \mathcal{SO}(2)$, resulting in the following transformation.

$$x_\mathrm{g}(q) = x + x_\mathrm{m}\cos(\theta) - y_\mathrm{m}\sin(\theta), \tag{4.3}$$
$$y_\mathrm{g}(q) = \mathrm{r} + x_\mathrm{m}\sin(\theta) + y\cos(\theta). \tag{4.4}$$

Furthermore, we can differentiate (4.3) and (4.4) to get expressions for $\dot{x}_\mathrm{g}(q,\dot{q})$ and $\dot{y}_\mathrm{g}(q,\dot{q})$. In the above formulas, we utilise the assumption that the robot is moving without slipping and therefore the relation between the center of the disc $[x\ y]^\top$ and θ is $[x\ y]^\top = [-\theta\mathrm{r}\ \mathrm{r}]^\top$.

In order to evaluate (4.2) we must compute the kinetic and potential energies of the system. Consider the former first. The system is an interconnection of two parts: a rolling disc and a point of mass. Upon denoting individual kinetic energies of disc and mass as T_d and T_m respectively, the total kinetic energy becomes

$$T_\mathrm{d}(q,\dot{q}) = \frac{1}{2}\mathrm{m_d}\dot{x}^2 + \frac{1}{2}\mathrm{J_d}\dot{\theta}^2 = \frac{3}{4}\mathrm{m_d}\mathrm{r}^2\dot{\theta}^2, \tag{4.5}$$
$$T_\mathrm{m}(q,\dot{q}) = \frac{1}{2}\mathrm{m_m}\left(\dot{x}_\mathrm{g}^2(q,\dot{q}) + \dot{y}_\mathrm{g}^2(q,\dot{q})\right), \tag{4.6}$$
$$T(q,\dot{q}) = T_\mathrm{d}(q,\dot{q}) + T_\mathrm{m}(q,\dot{q}), \tag{4.7}$$

where $\mathrm{m_d}$ is the mass of the disc, $\mathrm{m_m}$ is the central mass, $\dot{x}$ is the translational velocity of the rolling disc and $\mathrm{J_d} = \frac{1}{2}\mathrm{m_d}\mathrm{r}^2$ is the disc's moment of inertia.

When considering the potential energy V, it is important to account for the flexible segments that connect the central mass of the robot to its outer rim. We model these segments as generalised springs and as such they contribute to the total potential energy of the system. Upon associating a label $i \in$ {A,B,C,D} to each spring, based on the attachment points on the rim (Fig. 4.3), the total potential energy can be computed as a sum of partial potential energies as

$$\begin{aligned} V(q,\dot{q}) =& V_{\mathrm{d}}(q,\dot{q}) + V_{\mathrm{m}}(q,\dot{q}) + V_{\mathrm{A}}(q,\dot{q}) + \\ & V_{\mathrm{B}}(q,\dot{q}) + V_{\mathrm{C}}(q,\dot{q}) + V_{\mathrm{D}}(q,\dot{q}). \end{aligned} \tag{4.8}$$

Note, that in (4.8) $V_{\mathrm{d}} = 0$ as the disc is only allowed to roll on the ground. $V_{\mathrm{m}}(q,\dot{q}) = \mathrm{m_m\, g\, y_g}(q)$ is the potential energy of the central mass (with g being the gravitational acceleration). The potential energy of a spring i depends on its deformation d_i and the spring constant k_i as $V_i = \frac{1}{2}\mathrm{k_i}\, d_{\mathrm{i}}^2$. Considering the same k for all the springs, we can write (4.8) as

$$V(q,\dot{q}) = \mathrm{m_m\, g\, y_g}(q) + \frac{1}{2}\mathrm{k}\left(d_{\mathrm{A}}^2(q) + d_{\mathrm{B}}^2(q) + d_{\mathrm{C}}^2(q) + d_{\mathrm{D}}^2(q)\right). \tag{4.9}$$

The deflection d_i of each spring can be computed in the rim–connected coordinate frame as the difference between its nominal length r (radius of the wheel) and its current length, based on the position of the mass $[x_{\mathrm{m}}\ y_{\mathrm{m}}]^\top$

$$\begin{aligned} d_{\mathrm{A}}(q) &= \mathrm{r} - \sqrt{(-\mathrm{r} + x_{\mathrm{m}})^2 + y_{\mathrm{m}}^2}, & d_{\mathrm{b}}(q) &= \mathrm{r} - \sqrt{(\mathrm{r} + x_{\mathrm{m}})^2 + y_{\mathrm{m}}^2}, \\ d_{\mathrm{C}}(q) &= \mathrm{r} - \sqrt{x_{\mathrm{m}}^2 + (-\mathrm{r} + y_{\mathrm{m}})^2}, & d_{\mathrm{D}}(q) &= \mathrm{r} - \sqrt{x_{\mathrm{m}}^2 + (\mathrm{r} + y_{\mathrm{m}})^2}. \end{aligned}$$

Using (4.7) and (4.9), equation (4.2) can be written as

$$\begin{aligned} L(q,\dot{q}) = & \frac{3}{4}\mathrm{m_d r^2}\dot{\theta}^2 + \frac{1}{2}\mathrm{m_m}\left(\dot{x}_{\mathrm{g}}^2(q,\dot{q}) + \dot{y}_{\mathrm{g}}^2(q,\dot{q})\right) + \\ & \mathrm{m_m\, g\, y_g}(q) + \frac{1}{2}\mathrm{k}\left(d_{\mathrm{A}}^2(q) + d_{\mathrm{B}}^2(q) + d_{\mathrm{C}}^2(q) + d_{\mathrm{D}}^2(q)\right) \end{aligned} \tag{4.10}$$

giving the Lagrangian of the system.

We also need to account for the dissipative processes in the system. A simple model of rolling friction between the robot and the ground is considered as per

$$F_{\mathrm{r}} = (\mathrm{m_d} + \mathrm{m_m})\, \mathrm{C_{rr}\, g}, \tag{4.11}$$

where $\mathrm{C_{rr}}$ is the rolling resistance coefficient. Another dissipative process in the system is the damping in the springs. It is characterised by the damping constant b and acts only in the direction of the springs. We consider only the spring A first. A simple damper is described by $F_{\mathrm{d}} = \mathrm{b}\dot{\zeta}$, where ζ represents the length of the damper and in this case is equal to the length of

the spring. Our goal is now to express its rate of change. For spring A, the fixed attachment point is represented by the position vector $p_{\mathrm{A}}^{\mathrm{d}} = [0 \ -\mathrm{r}]^{\top}$ (in coordinate system O_{d}), and the difference between the two ends of the spring is given by $\zeta_{\mathrm{A}} = p_{\mathrm{m}}^{\mathrm{d}} - p_{\mathrm{A}}^{\mathrm{d}} = [x_{\mathrm{m}} \ (y_{\mathrm{m}} + \mathrm{r})]^{\top}$. We can now differentiate the norm of ζ_{A} to get the following damper equation

$$F_{\mathrm{dA}} = \frac{1}{2}\mathrm{b}\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{\zeta_{\mathrm{A}}^{\top}\zeta_{\mathrm{A}}}\right)^{2}. \tag{4.12}$$

For the other springs we utilise the same approach. In (4.12), we use squaring and dividing the result with one half for the sake of nicer results.

The position of the mass is driven by four actuators, each associated with one of the flexible segments. Ultimately, the combined action of these actuators comprises the non–conservative forces F_{x} and F_{y} acting on the central mass. In order to determine, how individual actuators contribute to the resulting forces, we need to compute the angle ϕ_i between the disc–connected coordinate system O_{d} and vector $-\zeta_i$, expressing the directions of individual springs i. The angles ϕ_i are then

$$\begin{aligned} \phi_{\mathrm{A}} &= \operatorname{atan}(-x_{\mathrm{m}}, -\mathrm{r} - y_{\mathrm{m}}), & \phi_{\mathrm{B}} &= \operatorname{atan}(-y_{\mathrm{m}}, \mathrm{r} - x_{\mathrm{m}}), \\ \phi_{\mathrm{C}} &= \operatorname{atan}(-x_{\mathrm{m}}, \mathrm{r} - y_{\mathrm{m}}), & \phi_{\mathrm{D}} &= \operatorname{atan}(-y_{\mathrm{m}}, -\mathrm{r} - x_{\mathrm{m}}). \end{aligned}$$

Upon associating a label u_i with the control action of each actuator, we can express the forces F_{x} and F_{y} by projecting u_i onto the coordinate axes of O_{d}

$$F_{\mathrm{x}} = u_{\mathrm{A}}\sin(\phi_{\mathrm{A}}) + u_{\mathrm{B}}\cos(\phi_{\mathrm{B}}) + u_{\mathrm{C}}\sin(\phi_{\mathrm{C}}) + u_{\mathrm{D}}\cos(\phi_{\mathrm{D}}), \tag{4.13}$$

$$F_{\mathrm{y}} = u_{\mathrm{A}}\cos(\phi_{\mathrm{A}}) + u_{\mathrm{B}}\sin(\phi_{\mathrm{B}}) + u_{\mathrm{C}}\cos(\phi_{\mathrm{C}}) + u_{\mathrm{D}}\sin(\phi_{\mathrm{D}}). \tag{4.14}$$

To be concise, we can introduce a vector $u = [u_{\mathrm{A}} \ u_{\mathrm{B}} \ u_{\mathrm{C}} \ u_{\mathrm{D}}]^{\top}$ and write (4.13) and (4.14) shortly as $F_{\mathrm{x}}(u, q)$ and $F_{\mathrm{y}}(u, q)$, respectively.

Finally, a general form of the *Euler–Lagrange equation* is

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L(z, \dot{z})}{\partial \dot{z}} - \frac{\partial L(u, \dot{z})}{\partial z} = -\frac{\partial \mathcal{D}(z, \dot{z})}{\partial \dot{z}} + F, \tag{4.15}$$

where z and $\dot{z}$ are some generalised coordinates, $\mathcal{D}$ represents the dissipative and $F = [F_{\mathrm{x}} \ F_{\mathrm{y}}]^{\top}$ the non–conservative forces, respectively. Because there are three generalised coordinates describing the system there will also be three Euler–Lagrange equations, each associated with one coordinate.

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L(q, \dot{q})}{\partial \dot{\theta}} - \frac{\partial L(q, \dot{q})}{\partial \theta} = -F_{\mathrm{r}}, \tag{4.16}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L(q, \dot{q})}{\partial \dot{x}_{\mathrm{m}}} - \frac{\partial L(q, \dot{q})}{\partial x_{\mathrm{m}}} = -\frac{\partial \left(F_{\mathrm{dA}} + F_{\mathrm{dB}} + F_{\mathrm{dC}} + F_{\mathrm{dD}}\right)}{\partial \dot{x}_{\mathrm{m}}} + F_{\mathrm{x}}(u, q), \tag{4.17}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L(q, \dot{q})}{\partial \dot{y}_{\mathrm{m}}} - \frac{\partial L(q, \dot{q})}{\partial y_{\mathrm{m}}} = -\frac{\partial \left(F_{\mathrm{dA}} + F_{\mathrm{dB}} + F_{\mathrm{dC}} + F_{\mathrm{dD}}\right)}{\partial \dot{y}_{\mathrm{m}}} + F_{\mathrm{y}}(u, q). \tag{4.18}$$

Symbol	Description	Value	Unit
r	Radius of the rim	0.46	m
$\mathrm{m_d}$	Mass of the rim	0.2	kg
$\mathrm{m_m}$	Mass of the central block	0.9	kg
$\mathrm{C_{rr}}$	Coefficient of rolling resistance	0.1	-
b	Damping constant	2	Nm/s
k	Spring constant	30	N/m

Table 4.1: Parameters of the nonlinear ŽůžO model.

This results in a system of three second order differential equations that can be translated into six first order differential equations constituting the model of the robot described with a state $\overline{x} = \left[\theta\ \dot{\theta}\ x_\mathrm{m}\ \dot{x}_\mathrm{m}\ y_\mathrm{m}\ \dot{y}_\mathrm{m}\right]^\top$. We have focused on detailing the crucial steps leading to the derivation of equations (4.16), (4.17), and (4.18). However, due to space limitations and the fact that obtaining the final form of the differential equations is no longer specific to the described approach, we have omitted their inclusion. We do provide a list of physical parameter values used in the model in Table 4.1. It should be noted that not all of these parameters perfectly align with the physical reality. While easily determinable parameters, such as the radius of the rim, were directly measured and used in the model, other, like the spring constant of the flexible segments, were selected iteratively to achieve visually similar behaviour to the real robot. However, no extensive validation against the real robot has been performed and the model currently captures the utilised principle of motion rather than a particular structure. The precise identification of all parameters poses a challenge that can be addressed in future work.

4.3 Dynamic Mode Decomposition with Control (DMDc)

Dynamic Mode Decomposition (DMD) is a versitile matrix decomposition and dimensionality reduction technique that can also be used for obtaining best–fit linear models. It stemms in the field of fluid dynamics and was developed to extract dynamic information from flow fields. It was first defined as an algorithm by Schmid [Sch10] and since then it proved useful in a variety of areas ranging from video processing, epidemology [KBBP16] to control of dynamical systems [PBK16]. Since the control of dynamical systems is almost exclusively coupled with some form of inputs, the DMD algorithm for autonomous systems is ommitted for the sake of brevity. Instead, we describe only its version for systems with inputs – DMD with control (DMDc).

We will start by briefly describing the *Koopman operator* approach to analysing dynamical systems which offers the rationale behind the DMDc. However, its thorough understanding is not crucial in the context of this work. Then, we detail the process of obtaining a linear predictor of nonlinear dynamics using the DMDc algorithm, which is a method that we encounter in several places in the text.

The main appeal of the Koopman operator is that it provides a well grounded approach for describing nonlinear dynamics using linear operator with the ability of describing the nonliearity ideally. However, such Koopman operator can end up being infinitely dimensional and thus we try to approximate it using a classical (finite) linear predictor (4.23). We use DMDc to find the matrices A, B and C defining the linear system predicting the nonlinear dynamics.

The foundations for utilising the Koopman operator for analysing dynamical system were laid by the works of Koopman and von Neumann in the 1930s [Koo31, KN32]. Consider now an autonomous discrete dynamical system

$$x[k+1] = \mathcal{T}\left(x[k]\right), \qquad x[k] \in \mathcal{M}, \tag{4.19}$$

defined on the state space $\mathcal{M}$ governed by a nonlinear transition mapping $\mathcal{T}(x[k])$. For the sake of illustrativeness we are restricting ourselves to the case of autonomous systems only, however all of the Koopman operator theory can be extended to systems with external inputs and we reffer the reader to [KM18a] for more insight. The core idea is to shift the focus from the mapping $\mathcal{T}$ to some (possibly infinitely many) user–defined functions of the states, so–called *observables*. This is motivated by the fact that upon defining observable as a function $\psi : \mathcal{M} \longrightarrow \mathbb{R}, \in \mathcal{F}$, where $\mathcal{F}$ is typically an inifinitely dimensional vector space, the Koopman operator $\mathcal{K} : \mathcal{F} \longrightarrow \mathcal{F}$ defined as

$$\left(\mathcal{K}\psi\right)\left(x[k]\right) = \psi\left(\mathcal{T}(x[k])\right) = \psi\left(x[k+1]\right) \tag{4.20}$$

advances every observable ψ from the time step k to $k+1$. One of the fundamental properties of $\mathcal{K}$ is its linearity since for any two observables ψ_1 and ψ_2 and scalars α_1 and α_2 the following holds

$$\mathcal{K}\left(\alpha_1\psi_1 + \alpha_2\psi_2\right) = \alpha_1\mathcal{K}\left(\psi_1\right) + \alpha_2\mathcal{K}\left(\psi_2\right). \tag{4.21}$$

So, using the Koopman operator one can convert the analysis of finite–dimensional nonlinear system into the analysis of the infinite–dimensional linear operator with possibly infinitely many observables ψ of the original states. Thus we aim to find a finite–dimensional approximation of such operator using (4.23). For more information about the Koopman operator theory, the reader is reffered e.g. to [BBKK21].

Consider now the extension of (4.19) to the case of systems with inputs

$$x[k+1] = \mathcal{T}_{\mathrm{u}}\left(x[k], u[k]\right), \qquad x[k] \in \mathcal{M},\ u[k] \in \mathcal{U}. \tag{4.22}$$

The process of obtaining an accurate linear predictor of the nonlinear dynamics (4.22) includes *lifting* the nonlinear dynamics into a higher dimensional space where the evolution of the lifted state becomes approximately linear. This can be done by selecting appropriate s lifting mappings (observables). For the sake of compactness, we will combine all observables into one vector $\Psi = [\psi_1 \ \cdots \ \psi_s]^\top$ which allows for constructing the predictor's state from the states of the system as $z[k] = \Psi(x[k]) = [\psi_1\left(x[k]\right)\ \psi_2\left(x[k]\right)\ \cdots\ \psi_s\left(x[k]\right)]^\top$. Then the predictor takes a form of a discrete linear system

$$\begin{aligned} z[k+1] &= Az[k] + Bv[k], \\ \hat{y}[k] &= Cz[k], \end{aligned} \tag{4.23}$$

with $\hat{y}[k]$ being the prediction of the ouput $y[k]$ of the nonlinear system at time k, $z[k]$ being the lifted state and $v[k] = \Psi(u[k])$ the lifted input.

We have selected the lifting observable ψ to be the so–called *delay embedding*. This resorts to stacking time–shifted copies of the state variables as the lifted state vector $z[k]$. (4.24) expands the resulting form of $z[k]$ for s delay embeddings. The use of delay embeddings as the observable mapping has been documented and justified e.g. in [TRL+14].

$$z[k] = \begin{bmatrix} x[k-s] \\ \vdots \\ x[k-1] \\ x[k] \end{bmatrix}. \tag{4.24}$$

The next step is to collect p snapshots of measured values y_i and $u_i\ \forall i \in \{1, \ldots, p\}$ (i being the index in the measured sequence of shapshots) on the real nonlinear system. Then, we can arrange the data in the following manner while assuming that the measurements satisfy the relation $y_i^+ = \mathcal{T}(y_i, u_i)$ (here, note that $x = y$)

$$\begin{aligned} X_{\mathrm{lift}} &= \begin{bmatrix} \Psi(y_1) & \Psi(y_2) & \cdots & \Psi(y_p) \end{bmatrix} = \begin{bmatrix} z_1 & z_2 & \cdots & z_p \end{bmatrix}, \\ X'_{\mathrm{lift}} &= \begin{bmatrix} \Psi(y_1^+) & \Psi(y_2^+) & \cdots & \Psi(y_p^+) \end{bmatrix} = \begin{bmatrix} z_1^+ & z_2^+ & \cdots & z_p^+ \end{bmatrix}, \\ \Omega_{\mathrm{lift}} &= \begin{bmatrix} \Psi\left(u_1\right) & \Psi\left(u_2\right) & \cdots & \Psi\left(u_{p-1}\right) \end{bmatrix} = \begin{bmatrix} v_1 & v_2 & \cdots & v_{p-1} \end{bmatrix}. \end{aligned}$$

Here the matrix X'_{lift} is just X_{lift} after time–shifting $k := k+1$. Using the above, we can rewrite the state evolution equation (4.23) for all k as

$$X'_{\mathrm{lift}} \approx AX_{\mathrm{lift}} + B\Omega_{\mathrm{lift}}. \tag{4.25}$$

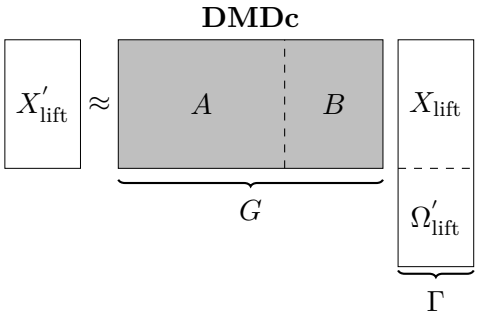


Figure 4.4: Data distribution in the DMDc algorithm.

Utilizing the matrices X_{lift}, X'_{lift} and Ω_{lift}, DMDc finds best–fit approximations of A and B that satisfy (4.25) for all snapshots i. Relation (4.25) can further be rewritten as

$$X'_{\text{lift}} \approx \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} X_{\text{lift}} \\ \Omega_{\text{lift}} \end{bmatrix} = G\Gamma, \tag{4.26}$$

so that $G = [A\ B]$ is the augmented operator matrix and $\Gamma = [X_{\text{lift}}\ \Omega_{\text{lift}}]^\top$ is the augmented data matrix. An illustration showing the data arrangement in (4.26) is in the Figure 4.4.

We find the matrix G from (4.26) to minimise the following error

$$||X'_{\text{lift}} - G\Gamma||_F, \tag{4.27}$$

where $||M||_F = \sqrt{\sum_{j=1}^{n}\sum_{k=1}^{m} M_{jk}^2}$ is the Frobenius norm. G is then given by

$$G = X'_{\text{lift}}\Gamma^\dagger, \tag{4.28}$$

where † denotes the Moore–Penrose pseudoinverse. One method for finding the minimzer of the Frobenius norm is *Singular Value Decomposition* (SVD). Using the SVD of $\Gamma = U\Sigma V^*$, (4.26) can be expressed as

$$G = X'_{\text{lift}} V\Sigma^{-1}U^*. \tag{4.29}$$

The operators A and B are then extracted from G by suitably splitting along columns into two matrices. Effectively, when splitting the left singular vectors U into two separate components $U_1 \in \mathbb{R}^{sn\times sn}$ and $U_2 \in \mathbb{R}^{sn\times sm}$, A and B are found as (m being the number of inputs into the system)

$$\begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} X'_{\text{lift}} V\Sigma^{-1}U_1^* & X'_{\text{lift}} V\Sigma^{-1}U_2^* \end{bmatrix}. \tag{4.30}$$

In this case, where the mapping ψ just rearanges the state variables, the matrix C can be directly constructed so that the output $\hat{y}[k]$ corresponds with the not–delayed current values of $x[k] \in \mathbb{R}^n$ as

$$C = [0_{n\times sn}\ I_{n\times n}]. \tag{4.31}$$

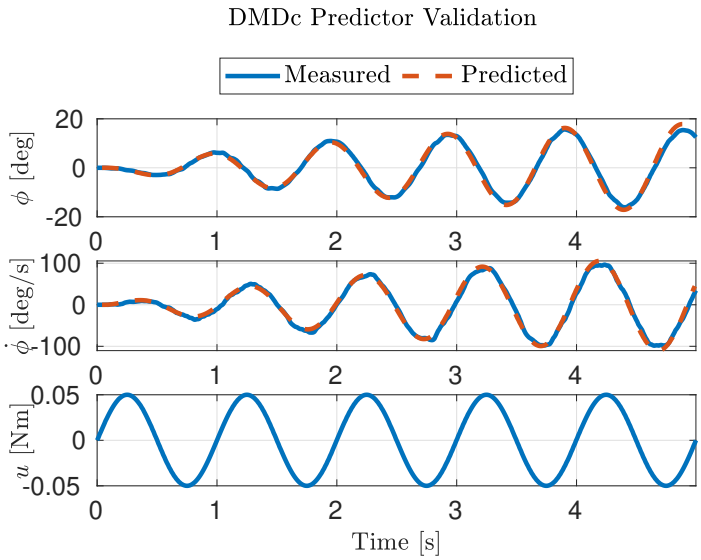


Figure 4.5: Comparison of the nonlinear system and linear predictions.

We showcase the performance of the linear predictor that can be obtained using this method in the Figure 4.5. There, we plot the comparison of a measured response of the Voxel Tower (described in Chapter 5) and the DMDc–based predictions of states ϕ (angle) and $\dot{\phi}$ (angular velocity) when subjected to sine–wave input.

For a more in–depth treatment of this topic, we reffer the reader for example to [AKM18]. The convergence of the DMDc algorithm that utilises observables was then proven in [KM18b]. Another way of constructing predictors exploiting the system dynamics was then proposed in [KM20].

Chapter 5

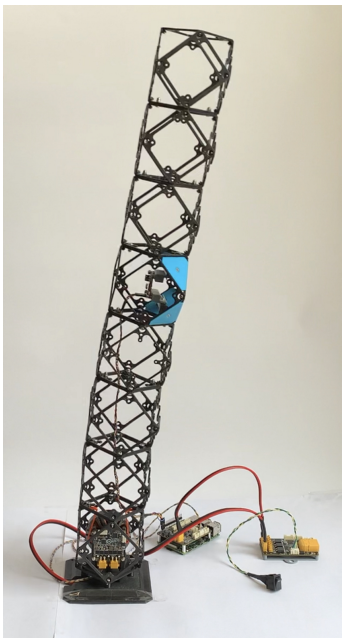
Voxel Tower

Building upon the foundation laid in the preceding chapters, we now shift our focus to the heart of this work: the three digital material structures. In this chapter, we introduce the first of these structures: the Voxel Tower in Figure 5.1a. It comprises 7 vertically connected semi–compliant voxels and one rigid voxel housing an *inertial measurement unit* MPU9250[1] IMU sensor. The whole structure is driven by the actuated voxel at its bottom end. Acting as the central control hub is a Raspberry Pi 4b[2], which runs the control algorithms. The deveolpment of said algorithms forms the main contribution of this chapter and is discussed upon addressing some preceeding tasks, including, but not limited to, the creation of a linear model for the system.

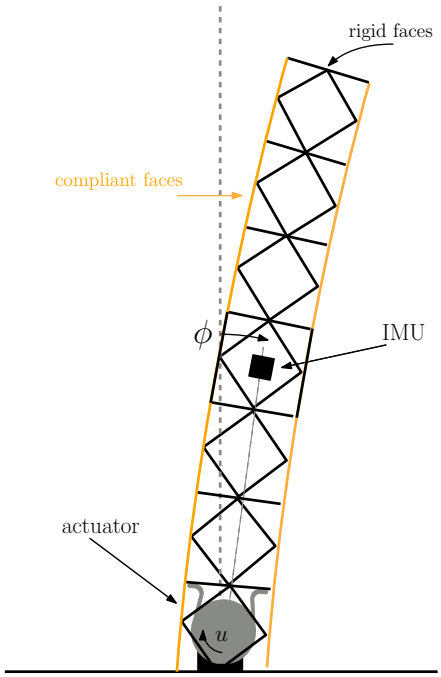
We organise this chapter as follows. Firstly, we establish the mathematical foundation for describing the system and address some subtleties in creating a linear model. We proceed to studying its validity in Section 5.1. Subsequently, in Section 5.2, we address challenges encountered when utilising measured data to develop the model. We then explore two control scenarios for the presented structure: disturbance attenuation and oscillation damping in Section 5.3, and trajectory tracking in Section 5.4. Finally, Section 5.5 concludes the chapter with a discussion on the level of accomplishment of individual tasks and the insights gained from the conducted experiments.

The Voxel Tower represents a *single–input–multiple–output* (SIMO) system that contains one actuator at the bottom and one sensor for measuring the

[1] https://www.sparkfun.com/products/16832
[2] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/



(a) : Photo of the Voxel Tower in motion.



(b) : A graphic of the Voxel Tower system with the meaning of individual variables.

Figure 5.1: The Voxel Tower.

tilt of the structure. Despite its apparent simplicity, the structure presents a variety of interesting and challenging tasks that need to be addressed. One such challenge is the placement of sensors, a topic that has been extensively studied [BBMR87, NT19]. While acknowledging that there is no claim to optimality, we experimentally decided to mount the sensor halfway up the structure. This positioning allows us to capture even higher modes of the structure, adding to the range of movement we can study.

The dynamics of the Voxel Tower can generally be described by a map

$$\dot{x}(t) = f(x, u, t), \tag{5.1}$$

where $x(t) \in \mathbb{R}^{n=2}$ and $u(t) \in \mathbb{R}$ are the state vector and input at time t and $f(\cdot)$ is an unknown function. The state of the system can be expressed as $x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} \phi(t) \\ \dot{\phi}(t) \end{bmatrix}$, where $\phi(t)$ [deg] and $\dot{\phi}(t)$ [deg/s] are the tilt angle and velocity, respectively. $u(t)$ [Nm] is the torque applied to the motor. Furthermore, since both states can be directly measured using the IMU sensor, they are also considered as outputs of the system. To reduce the noise in the measured velocity, we applied a moving mean filter based on 10 measurements. We sample the dynamics (5.1) (taking the filtered velocity) by performing measurements on the system at time instants

$k\mathrm{T_s},\ k \in \mathbb{N} \cup \{0\}$, where $\mathrm{T_s} = 0.005\,\mathrm{s}$ is the sampling period of the system. The obtained snapshots of measured data are then utilised to create a discrete linear predictor approximating the continuous nonlinear dynamics (5.1). We compute the predictor (5.2) using the approach described in Section 4.3.

We developed two models: one for stabilisation and another for tracking. The only difference between the two models is the lifting map ψ, specifically the number of delay embeddings s. The stabilisation model used $s = 2$ delay embeddings, while the tracking predictor yielded better results with $s = 3$. The motivation behind having two models was the observation that the tracking performance was not sufficient using the model for stabilisation. Therefore, we decided to design a separate predictor specifically for tracking. In this part, we use the notation introduced in Section 4.3. To avoid repetition and maintain brevity, we refrain from restating the definitions and notation in this section. In case a symbol's meaning is unclear, we refer the reader back to Section 4.3 for clarification.

As we have already indicated, the chosen lifting map ψ was to introduce delay embeddings. With this choice, we can construct the matrices $X_{\text{lift}},\ X'_{\text{lift}}$ and Ω_{lift} from the measured snapshots. These we then use to compute the linear predictor utilising the lifted state $z[k] = \Psi(x[k])$ and input $v[k] = \Psi(u[k])$

$$\begin{aligned} z[k+1] &= Az[k] + Bv[k], \\ \hat{y}[k] &= Cz[k], \end{aligned} \tag{5.2}$$

utilising the DMDc algorithm as per (4.30) and (4.31). A slight caveat arises when considering $s > 0$ delay embeddings. Namely, that using the lifted inputs Ω_{lift} in (4.30) results in B having $s+1 > 1$ columns. However, this does not correspond with the reality of having a single input to the system. We expand $v[k]$ and rewrite the state evolution equation from (5.2) to illustrate the issue

$$z[k+1] = Az[k] + \underbrace{\begin{bmatrix} | & & | & | \\ b_s & \cdots & b_1 & b_0 \\ | & & | & | \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} u[k-s] \\ \vdots \\ u[k-1] \\ u[k] \end{bmatrix}}_{v[k]}, \tag{5.3}$$

where b_i is a column of B corresponding with ith delayed entry in the vector $v[k]$. We can solve this discrepancy by incorporating the delayed inputs into the state vector, producing a new predictor

$$\begin{aligned} \bar{z}[k+1] &= \overline{A}\bar{z}[k] + \overline{B}u[k], \\ \hat{y}[k] &= \overline{C}\bar{z}[k], \end{aligned} \tag{5.4}$$

with $\bar{z}[k] = [z[k]\ u[k-s]\ \cdots\ u[k-1]]^\top$ being the new *augmented* state vector.

Besides producing the augmented state vector $\bar{z}[k]$, we need to alter the system matrices accordingly as well. Using the labels of individual columns of B as in (5.3), we can write the augmented state matrices as

$$\overline{A} = \begin{bmatrix} \mathrm{A} & b_s & \cdots & b_1 \\ 0_{s\times n} & & \mathrm{I}^1_{s\times s} & \end{bmatrix}, \quad \overline{B} = \begin{bmatrix} b_0 \\ 0_{1\times s-1} \\ 1 \end{bmatrix},$$
$$\overline{C} = \begin{bmatrix} 0_{2\times 2s} & \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} & 0_{2\times s} \end{bmatrix},$$

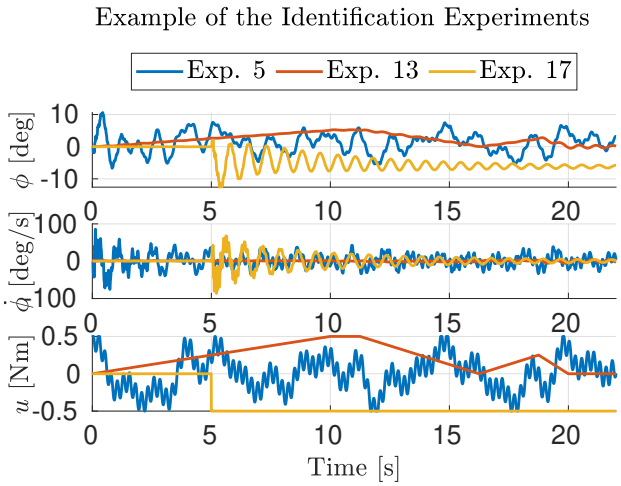
where $I^1_{s\times s}$ is a $s \times s$ matrix with ones on its superdiagonal. This way we have obtained a linear predictor of the nonlinear dynamics of the Voxel Tower with incorporated delay embeddings but just one input.

The process of creating the two aforementioned models then boils down to creating two predictors (5.2) using different number of delay embeddings and then composing two sets of $\overline{A}, \overline{B}$ and $\overline{C}$, each with different dimensions and based on the different matrices A and B.

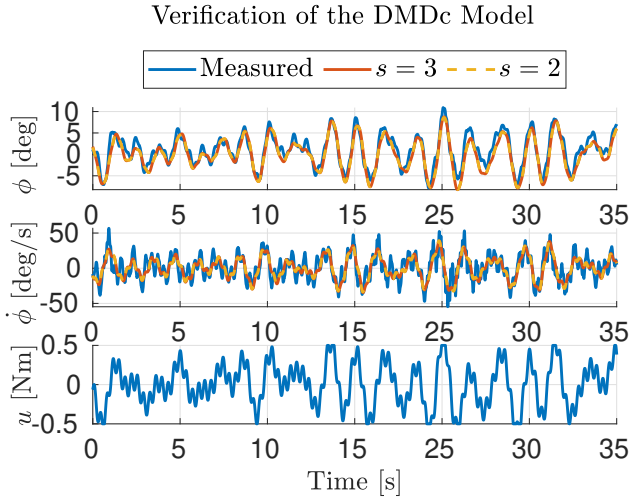
5.1 Linear Model Verification

The above text discusses the derivation and final form of the state matrices describing the linear predictor (5.4). In the following two sections, we discuss the practical aspects of the model acquisition problem.

The DMDc algorithm is a data–based method and as such its performance heavily rests on the supplied identification data. The identification data should be rich enough to contain as much of the system's dynamic modes and characteristics as possible. To ensure this, we created a set of 13 identification experiments, each lasting between 20 and 60 seconds. In these experiments, we applied various input signals such as step changes, ramps, and random composite harmonic signals. Figure 5.2a visualizes some of these signals along with their corresponding measured responses. However, we observed that some of the experiments produced data that negatively affected the performance of the developed controllers, despite reasonable predictions during the open loop verification. These subtleties and their impacts are described in more detail in the following Section 5.2.



(a) : Example of identification data.



(b) : Comparison of the two DMDc models.

Figure 5.2: Identification and verification of the DMDc model of the Voxel Tower.

In light of that observation, we selected only a subset of the measured experiments to be used for fitting the model. The selected responses were then stacked together to form the data matrices $\Omega_{\text{lift}}, X_{\text{lift}}$ and X'_{lift} from (4.26).

We collected a separate set of measured responses for testing purposes, which were not used in model creation. These measurements were used to validate the resulting predictor $\overline{A}$, $\overline{B}$, and $\overline{C}$. We assesed the accuracy of individual models by applying a given input and comparing the model's predictions with the measured responses. Figure 5.2b illustrates models' correct predictions of the system's response. In terms of angle, both models capture the behaviour of the nonlinear system quite well as the predictions closely follow the measured response. Although the models did not fully capture the high–frequency velocity oscillations, they accurately described

the overall trend and average changes. Therefore, we consider the models good enough to be used in controller design.

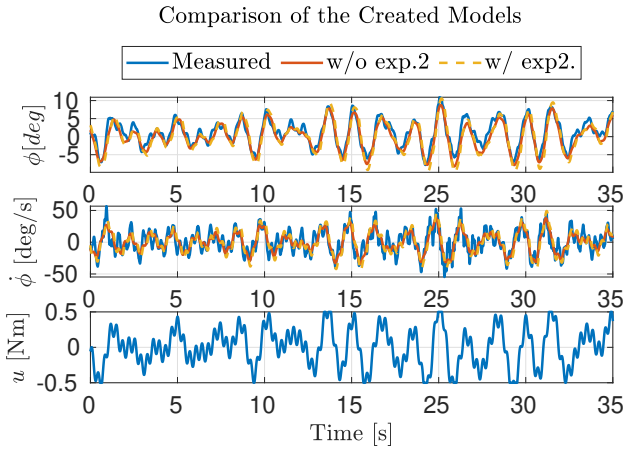
A quick note on the comparisson of the two models with different number of delay embeddings follows. Although their predicted responses are nearly identical, the rationale behind having two models stems from our observation during the design of the tracking controller. We found that the model with three delay embeddings ($s = 3$) exhibited significantly better behavior in this context.

5.2 Examining Sets of Identification Experiments

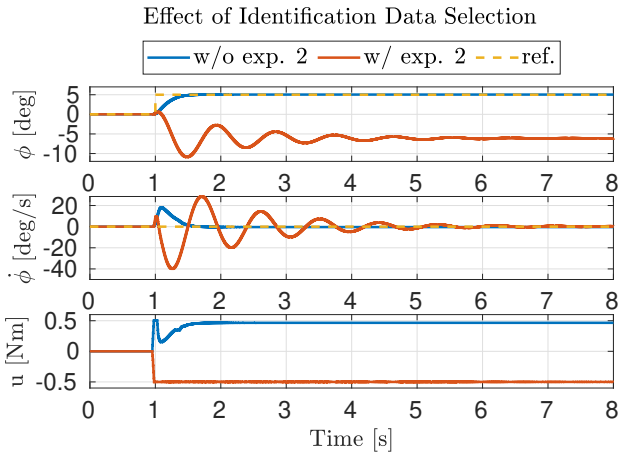
In light of the observation discussed in the previous section, we now turn our attention to the impact of identification data on the designed controller. Surprisingly, some of the measured identification responses had a negative effect on the controller, instead of improving it by providing the model with furhter information about the system's behaviour. To maintain brevity, we assign labels to the individual experiments using their corresponding index in the set ranging from 1 to 13.

To illustrate the issue, we tuned an MPC controller with almost ideal tracking response. Although this controller is too aggressive for driving the real system, it performs well on the linear model, allowing us to effectively demonstrate the problem at hand. The response of the linear model driven by this controller is in Figure 5.3b. We utilised identification experiments $\{1,\ 3,\ 6,\ 11,\ 12,\ 13\}$ to create the model. However, when including e.g. experiment no. 2 into the *training set*, the same controller completely fails in trying to follow the reference signal. This is illustrated in Figure 5.3b. At the same time, the inclusion or exclusion of experiment 2 in the training set does not appear to affect the model's ability to accurately predict the real system's behaviour, as shown in Figure 5.3a. Consequently, it becomes challenging to identify problematic responses in advance solely by evaluating the predictions of the linear model.

To investigate the root cause of the issue, we analysed the predictor matrices A and B. We revealed that the presence or absence of certain data in the identification process has a notable impact on the input matrix B. The specific values of resulting matrices B (without experiment 2) and $B^{'}$ (with experiment 2) are provided below. It is worth noting that we investigated



(a) : Both models predicting the outputs of the real system.



(b) : Both models being controlled by an MPC.

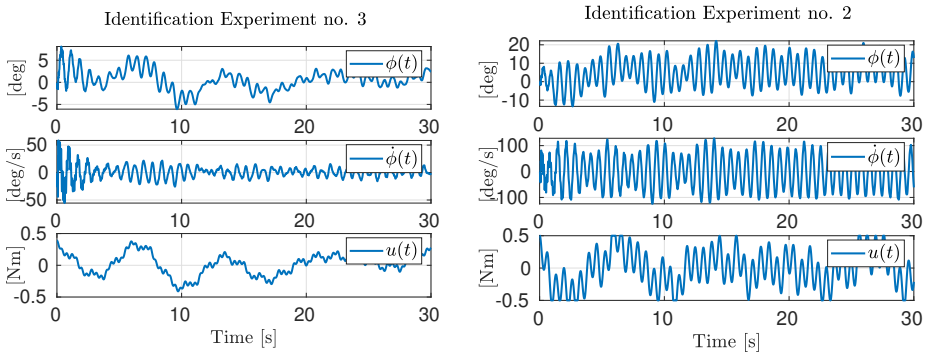
Figure 5.3: Behaviour of predictors created with and without utilising the experiment no. 2.

this for the model with 3 delay embeddings.

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0.1566 & 2.7876 & 0 & 1 \end{bmatrix}^\top,$$
$$B' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0.0728 & -0.7539 & 0 & 1 \end{bmatrix}^\top.$$

A quick remark: A similar pattern, as described below, persisted even when we explored different combinations of experiments. However, to maintain conciseness, we will not provide detailed information about these combinations. Instead, we have selected one specific case involving a pair of *good* and *bad* experiments that effectively demonstrate the issue.

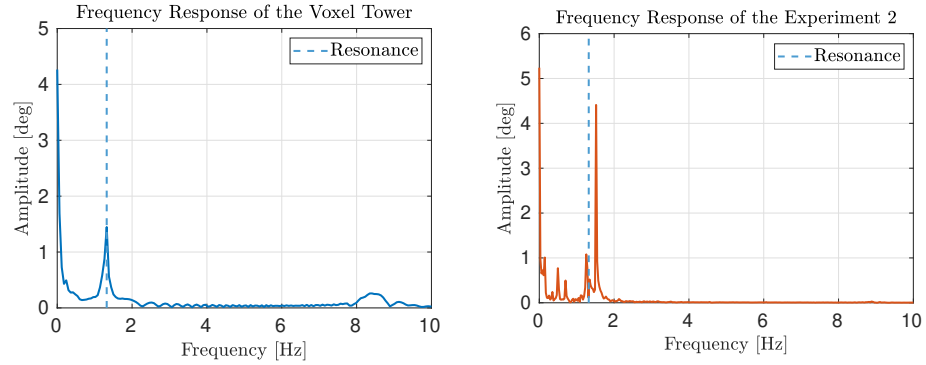
The form of matrix B' is not an issue by itslef but it can give us more insight into the behaviour of the *faulty* model. Individual elemets of B' are noticeably smaller in absolute value than the entries of B, signifying a much



(a) : Identification experiment number 3.

(b) : Identification experiment number 2.

Figure 5.4: The *good* and *bad* identification experiments.



(a) : Close up into the frequency sweep results.

(b) : Close up into the frequencies and amplitudes from the experiment 2.

Figure 5.5: Results of the performed frequency analysis.

lower influence of the input signal on the system. Furthermore, considering this observation in the context of the visualised experiments 2 and 3 in Figure 5.4, we notice one possible explanation. The core of the problem might be that in the experiment 2 (Fig. 5.4b), we excited the system *too much*, resulting in the inertia and own dynamics of the system overcoming the motor as the dominant driving force. As a result, the model fails to *learn* the input–output mapping accurately. To support this assumption, we performed a frequency analysis of the response in experiment no. 2. By comparing the resulting frequency spectrum (Fig. 5.5b), computed using Fast Fourier Transform, with the frequency spectrum of the Voxel Tower (Fig. 5.5a), we observed that the response in Fig. 5.4b exhibited highest amplitudes near the resonance frequency of the system. This confirms our assumption that the input in experiment 2 induced behavior that masks meaningful dynamics and actuator influence, rendering it unsuitable for identification purposes. In contrast, in experiment 3 (Figure 5.4a), despite lower amplitudes and slower changes in the driving signal, the influence of the actuator on the system is more evident.

5.3 Stabilisation

The correct understanding and control of vibrations, oscillations, and the behaviour under external disturbance are crucial to any static, let alone flexible, structure. Especially vertical structures are constantly exposed to various forms of external disturbance and sources of oscillations, such as wind, air turbulences, and, less frequently, earthquakes [TKS15].

In this section, we design and validate a *Linear Quadratic Regulator* (LQR) for oscillation damping and disturbance attenuation. The discrete version of an LQR is a controller that minimises the following classical quadratic cost function.

$$J(u, \bar{z}) = \sum_{k=0}^{\infty} \left(\bar{z}[k]^\top Q \bar{z}[k] + u[k]^\top R u[k] \right). \tag{5.5}$$

Considering the notation introduced earlier, we can formulate the optimisation problem that leads to the LQR as

$$\begin{aligned} \min_{u,\bar{z}} \sum_{k=0}^{\infty} \bar{z}[k]^\top Q \bar{z}[k] + u[k]^\top R u[k], \\ \text{subject to: } \bar{z}[k+1] = \overline{A}\bar{z}[k] + \overline{B}u[k], \\ \text{with parameters: } \bar{z}[0] \text{ is given.} \end{aligned} \tag{5.6}$$

In (5.6), matrices Q and R represent the penalties on the state and input, respectively. The design of an LQR essentially involves an iterative process where we adjust the elements of Q and R until the resulting controller aligns with our design goals. Utilising the `MATLAB` syntax, the tuned penalty matrices are

$$Q = \texttt{blkdiag}(0_{2s\times 1},\ 100,\ 1,\ 0_{s\times 1}), \qquad R = 50000.$$

Matrix Q takes this form because it is multiplied with the state vector $\bar{z}[t]$, which contains s delay embeddings. Specifically, the first $2s$ elements represent delayed states and last s elements represent delayed inputs. The weights 100 and 1 then correspond with $x_1[t]$ (angle) and $x_2[t]$ (angular velocity), respectively.

Due to the way (5.6) is formulated (specifically, because of the first term in the summation), LQR, by default, aims to drive the state vector $\bar{z}$ to zero. This is desirable for our current application, where the *zero* state corresponds with the Voxel Tower standing still in the upright position. However, for applications where the desired state is not *zero*, such as tracking, it is necessary to modify the LQR algorithm, e.g., by introducing integral control.

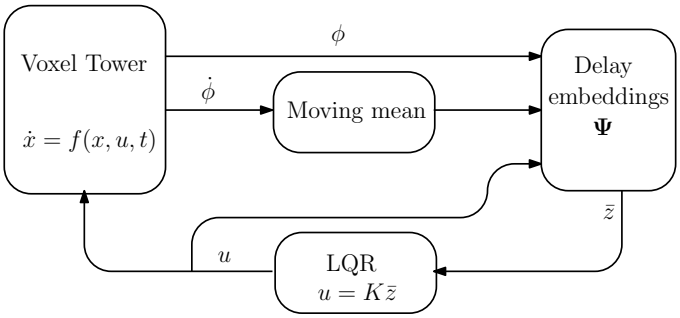


Figure 5.6: Signal diagram of the LQR control scheme.

Returning to the focus of this section – stabilisation, we can use the LQR algorithm without modifications. The optimal control minimising (5.5) results in a state feedback controller

$$u[t] = -K\bar{z}[t]. \tag{5.7}$$

The optimal gain matrix K can be found as

$$K = \left(\overline{B}^\top S\overline{B} + R\right)^{-1} \overline{B}^\top S\overline{A}, \tag{5.8}$$

where S is the solution of the associated discrete–time algebraic Ricatti equation

$$\overline{A}^\top S\overline{A} - S - \overline{A}^\top S\,\overline{B}\left(\overline{B}^\top S\overline{B} + R\right)^{-1} \overline{B}^\top SS\,\overline{A} + Q = 0. \tag{5.9}$$

Using penalties Q and R as evaluated above, the computed state feedback gain K is

$$K = \begin{bmatrix} -1.64 & -0.59 & -2.03 & 0.83 & 2.67 & -0.85 & 1.96 & 1.46 \end{bmatrix} \cdot 10^{-2}.$$

The Figure 5.6 depicts a signal diagram illustrating the feedback interconnection of the entire controlled system, including its individual components.

To validate the designed controller, we conducted three testing scenarios. Two of these scenarios involved studying the response to non–zero initial conditions, while the third scenario focused on evaluating the controller's ability to reject disturbances. We describe the designed experiments below.

1) Non–zero inital tilt: The first testing scenario involved tilting the tower and observing its response after a period of rest ($\dot{\phi} = 0$ deg/s). We

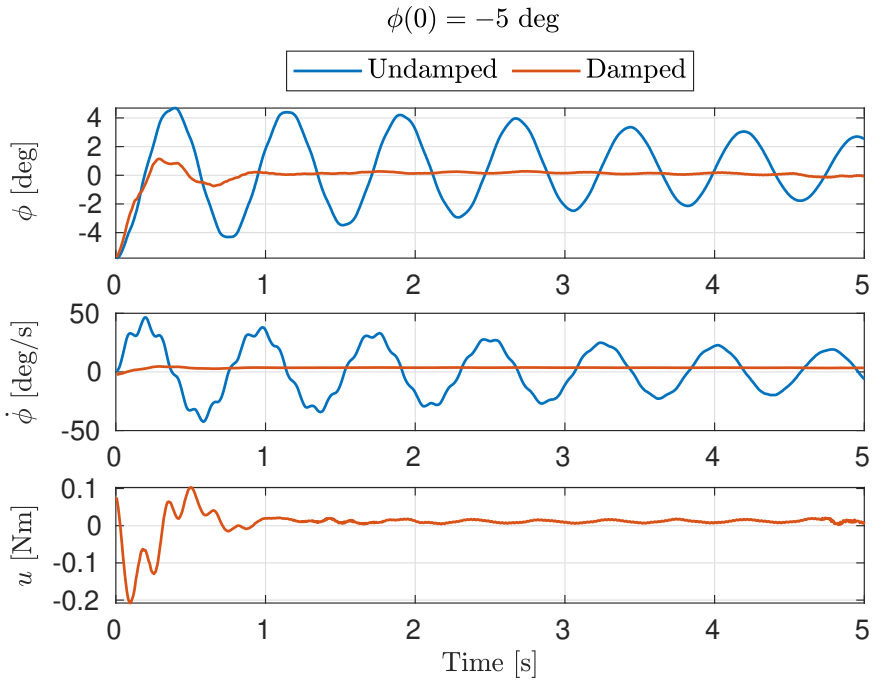


Figure 5.7: Response to non–zero initial deflection of -5 deg.

conducted experiments with two different initial deflections: $\phi(0) = -5$ deg and $\phi(0) = -20$ deg, to account for small as well as large starting angles. For each experiment, we compared the response of an autonomous system to the system controlled by the LQR. The measured results are presented in Figures 5.7 and 5.8.

2) Non–zero initial tilt and velocity: In real-world scenarios, it is rare for a structure to be completely still. Therefore, it is important to consider situations where not only the initial angle, but also the initial velocity is non–zero. This can occur in cases where the controller acts as a backup and is only activated when the system is not behaving as desired or when control goals change. To ensure repeatability, we set the initial state of the system using the actuator at the bottom of the structure. In this testing scenario, we first applied a precomputed sine wave torque excitation to the system (indicated by the red area in Fig. 5.9) to initiate movement. After 6 seconds, when the tower was already in motion, we switched to the LQR algorithm to govern the actuator. The results of these experiments are presented in Fig. 5.9, where we compare the behaviour of the damped system with the undamped response.

3) Disturbance rejection: The ability to surpress the effect of external disturbance is crucial to any structure operating in real–world environments,

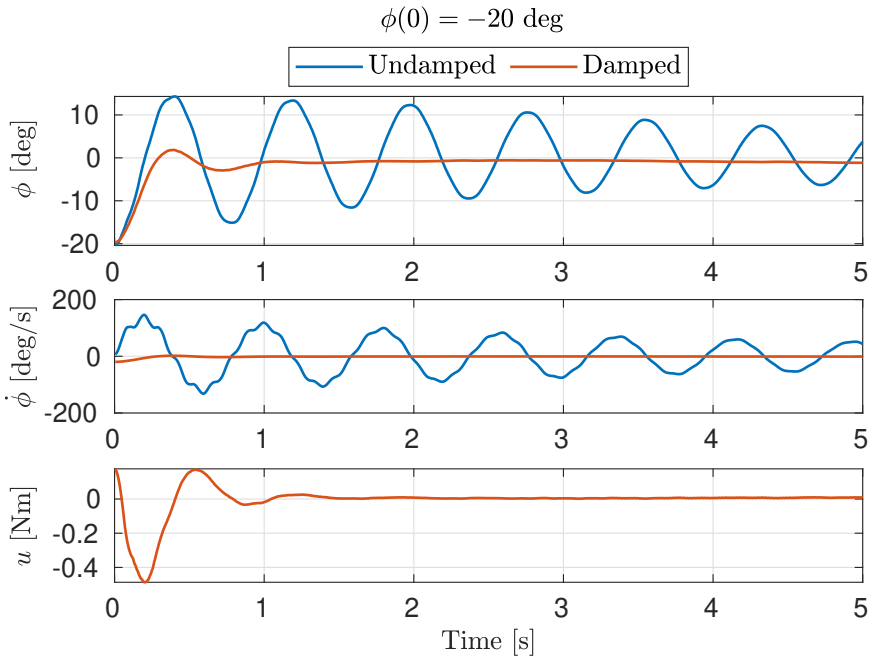


Figure 5.8: Response to non–zero initial deflection of -20 deg.

including digital material structures. In this testing scenario, we focus on the effect of a 0.5 s step changes in the force acting on the structure, which simulate the disturbances. We implement the simulated disturbance utilising the actuated voxel. During the time of the disturbance (indicated by the red areas of plot 5.10), the motor acts as its source, before the LQR takes back over. The disturbance signal is shown in the bottom plot of Fig. 5.10 as a black dashed line. The figure also illustrates the response of the Voxel Tower to the disturbance, comparing the behaviour of the undamped system with the system controlled by the LQR.

5.4 Reference Tracking

The second common problem in control engineering is known as *reference tracking*. This problem revolves around achieving the desired behaviour of the controlled system according to a predefined pattern. For the Voxel Tower, this pattern might consist of predefined values of the tilt angle that the system should mirror. The motivation behind this type of control lies in applications where e.g. an end–effector is attached to a flexible beam or where the structure needs to adjust its orientation based on changing environmental conditions. Reference tracking is a well–studied problem in

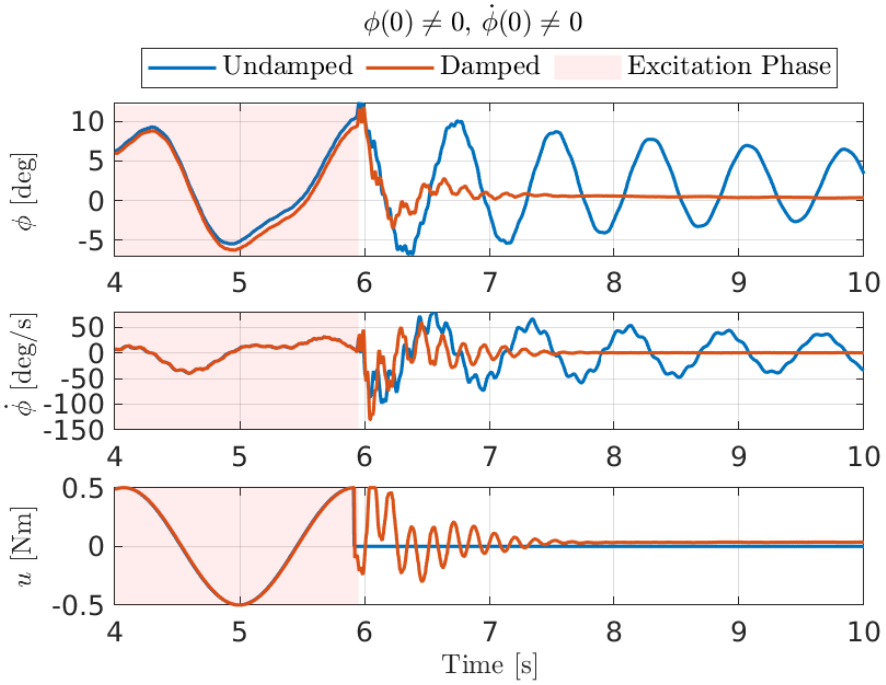


Figure 5.9: Response to non–zero initial angle and velocity.

control theory, and numerous research works have been published on this topic, including those specifically addressing trajectory tracking for flexible structures. In [PSSNCL15] the authors utilise the finite difference method to model a single flexible link robot and employ a PID controller to regulate the position of the beam's tip. Another approach, using iterative learning control (ILC) method for control of an X–Y positioning platform with flexible beams on each axis is described in [SYCX21].

To solve the tracking problem, we opted to implement the *Koopman Model Predictive Control* (KMPC), first introduced in [KM18a] and utilised for controlling complex dynamical systems as demonstrated e.g. in [DKH22]. This approach combines the classical Model Predictive Control (MPC) with a discrete linear predictor describing the system obtained using the Koopman theory. We use the predictor (5.4) derived earlier and created using the DMDc algorithm from Section 4.3. The decision to utilise MPC was motivated by its ability to incorporate input constraints, which is crucial considering the tendency of printed voxel faces to break under excessive stress.

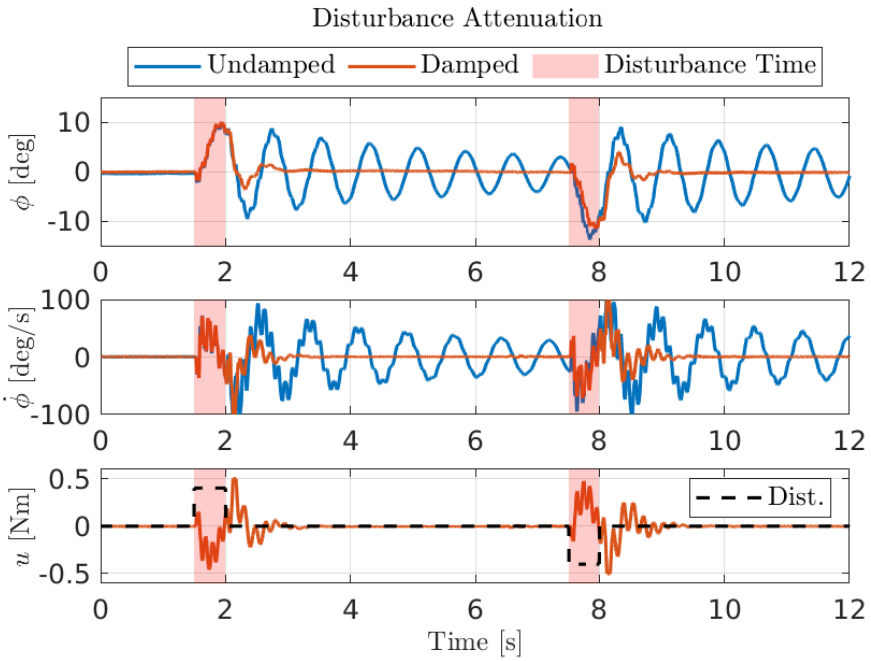


Figure 5.10: Results of the disturbance attenuation experiment.

5.4.1 Koopman Model Predictive Control

The classical Model Predictive Control (MPC) is a discrete–time optimal control algorithm. It works by using optimisation to find input sequences that minimise a given cost function $J(\cdot)$ over some time frame (so–called *prediction horizon*). Upon computing the optimal input sequence, it applies only its first element, thereby producing new measured output of the system, and repeats the whole process in the next time step. Internally, the MPC uses a linear model of the system to predict its behaviour over the prediction horizon and this is where Koopman theory comes into play. The KMPC does not rely on models obtained by e.g. linearising the model around some equilibria but rather uses Koopman theory to approximate the system's behaviour. This section is based on multiple texts. Among our main sources of information for this part were the lecture notes to the *Optimal and Robust Control*[3] course here at CTU. Some further aspects of combinig the MPC equations with the Koopman–based predictor are based on [DKH22, AKM18].

We will now formulate the KMPC optimisation problem for tracking. Let $\hat{y}[t] = \overline{C}\bar{z}[t]$ be the predicted output of the system and $r[t]$ the reference signal at time t. Then, upon introducing the *regulation error* $e[t] := r[t] - \hat{y}[t]$ we can define the tracking problem as finding such input sequence $u_t = \{u[t+k]\}_{k=0}^{N}$

[3] https://moodle.fel.cvut.cz/course/view.php?id=5716

that results in $e_t \longrightarrow 0$ over the prediction horizon of length N. Such sequence can be found by solving the following optimisation problem at every time step t

$$\begin{aligned} \min_{u_t, e_t} \quad & J\left(\{e[t+k]\}_{k=0}^{N}, \{u[t+k]\}_{k=0}^{N}\right), \\ \text{subject to} \quad & \bar{z}[k+1] = \overline{A}\bar{z}[k] + \overline{B}u[k], \quad k = 0, 1, \ldots, N-1, \\ & e[k] = r[k] - \overline{C}\bar{z}[k], \\ & u_{\min} \leq u[k] \leq u_{\max}, \\ \text{with parameters} \quad & \bar{z}[0] = \Psi(y[t]), \\ & r[k] \text{ given}, \ k = 0, 1, \ldots, N, \end{aligned} \tag{5.10}$$

where $y[t]$ is the measured output of the nonlinear system at time t and $u_{\min}$ and $u_{\max}$ are the lower and upper bounds on input, respectively. The loss function $J(\cdot)$ attains the standard quadratic form for an MPC tracking problem [AKM18]

$$\begin{aligned} J\left(e_t, u_t\right) = & \frac{1}{2} e[t+N]^\top S e[t+N] + \\ & + \frac{1}{2} \sum_{k=0}^{N-1} \left(e[t+k]^\top Q e[t+k] + u[t+k]^\top R u[t+k] \right). \end{aligned} \tag{5.11}$$

Matrices $Q \succeq 0$ and $S \succeq 0$ penalise the error inside and the end of the prediction horizon, respectively, while $R \succeq 0$ penalises the input. An important remark is that the problem (5.10) is a convex quadratic program. For such problems, there exist a number of high–performance solvers, and in this work, we have chosen to use the OSQP solver [SBG+20].

We need to address a caveat regarding the previous formulation. There may be cases where nonzero input $u[t] \neq 0$ is required to track a steady reference i.e. to have $e[t] = 0$. However, with (5.11) as above, these are two competing conditions when minimising. This issue can be resolved by adopting the *incremental* formulation of the problem (5.10) and penalising the increments of $u[t]$: $\Delta u[t] = u[t] - u[t-1]$. To achieve the incremental formulation we extend the state vector by introducing a new state variable $z^u[t] := u[t-1]$. This leads to the *augmented* state description as follows

$$\begin{bmatrix} \bar{z}[t+1] \\ z^u[t+1] \end{bmatrix} = \underbrace{\begin{bmatrix} \bar{A} & \bar{B} \\ 0 & I \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} \bar{z}[t] \\ z^u[k] \end{bmatrix}}_{\tilde{z}[t]} + \underbrace{\begin{bmatrix} B \\ I \end{bmatrix}}_{\tilde{B}} \Delta u[t], \tag{5.12}$$

$$\hat{y}[t] = \underbrace{\begin{bmatrix} \bar{C} & 0 \end{bmatrix}}_{\tilde{C}} \begin{bmatrix} \bar{z}[t] \\ z^u[t] \end{bmatrix}. \tag{5.13}$$

Using the above, the cost function from (5.11) can be written in terms of $\tilde{z}$ and Δu as

$$\begin{aligned} J(\tilde{z}, \Delta u) =& \frac{1}{2}\left(r_N - \tilde{C}\tilde{z}_N\right)^\top S \left(r_N - \tilde{C}\tilde{z}_N\right) + \\ &+ \frac{1}{2}\sum_{k=0}^{N-1}\left[\left(r_k - \tilde{C}\tilde{z}_k\right)^\top Q \left(r_k - \tilde{C}\tilde{x}_k\right) + \Delta u_k^\top R \Delta u_k\right]. \end{aligned} \tag{5.14}$$

To address space limitations, we relabeled the time indices in (5.14). Specifically, we introduced the notation $r_i = r[t+i]$ to represent the time steps more concisely. Because constant terms in (5.14) such as $r_N^\top S\, r_N$ play no role in the optimisation, we can ignore them. Furthermore, by creating the following vectors $\mathbf{r} = [r[t] \ \cdots \ r[t+N]]^\top$, $\tilde{\mathbf{z}} = [\tilde{z}[t] \ \cdots \tilde{z}[t+N]]^\top$ and $\Delta\mathbf{u} = [\Delta u[t] \ \cdots \ \Delta u[t+N-1]]^\top$, we can now write the cost function as

$$J(\tilde{\mathbf{z}}, \Delta\mathbf{u}) = \frac{1}{2}\tilde{\mathbf{z}}^\top \overline{\overline{Q}}\tilde{\mathbf{z}} - \mathbf{r}^\top \overline{\overline{T}}\tilde{\mathbf{z}} + \frac{1}{2}\Delta\mathbf{u}^\top \overline{\overline{R}}\Delta\mathbf{u} \tag{5.15}$$

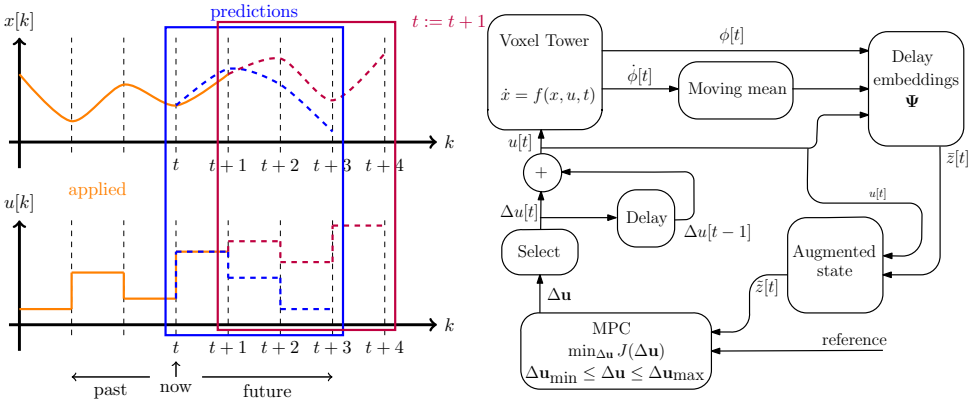
and use that to find the optimal input sequence. The matrices $\overline{\overline{Q}}$, $\overline{\overline{T}}$ and $\overline{\overline{R}}$ are explicitly stated in (A.1).

However, this still minimises simultaneously over the input sequence as well as over the state variables. Thus, the dimensionality of the problem scales with the number of state variables. This is impractical, as using our lifting function ψ would be at price of high computational complexity of the optimisation. To overcome this issue, we can get rid of the dependence on $\tilde{\mathbf{z}}$, by expressing it in terms of inputs and the initial state $\tilde{z}[t]$. This formulation is known as *dense* formulation of the problem and as such, it no longer suffers from the curse of dimensionality. The dense formulation is enabled by writing $\tilde{\mathbf{z}}$ as

$$\tilde{\mathbf{z}} = \underbrace{\begin{bmatrix} \tilde{B} & & & & \\ \tilde{A}\tilde{B} & \tilde{B} & & & \\ \tilde{A}^2\tilde{B} & \tilde{A}\tilde{B} & \tilde{B} & & \\ \vdots & & & \ddots & \\ \tilde{A}^{N-1}\tilde{B} & & \cdots & & \tilde{B} \end{bmatrix}}_{\overline{\overline{C}}} \Delta\mathbf{u} + \underbrace{\begin{bmatrix} \tilde{A} \\ \tilde{A}^2 \\ \vdots \\ \tilde{A}^N \end{bmatrix}}_{\hat{\tilde{A}}} \tilde{z}[t]. \tag{5.16}$$

After substituting the above into (5.15), expanding, and ignoring constant terms, the cost function becomes

$$J(\Delta\mathbf{u}) = \frac{1}{2}\Delta\mathbf{u}^\top \underbrace{\left(\overline{\overline{C}}^\top \overline{\overline{Q}}\,\overline{\overline{C}} + \overline{\overline{R}}\right)}_{\overline{\overline{H}}} \Delta\mathbf{u} + \left[\tilde{z}[t]^\top \mathbf{r}^\top\right] \underbrace{\begin{bmatrix} \hat{\tilde{A}}^\top \overline{\overline{Q}}\,\overline{\overline{C}} \\ -\overline{\overline{T}}\,\overline{\overline{C}} \end{bmatrix}}_{\overline{\overline{F}}^\top} \Delta\mathbf{u}. \tag{5.17}$$



(a) : Illustration of the MPC working principle.

(b) : Signal diagram of the KMPC control scheme for tracking.

Figure 5.11: (K)MPC schematics.

Using this, we can finally write the formulation of the (K)MPC optimisation problem that is solved at each time step. The problem is as follows

$$\begin{aligned} \min_{\Delta\mathbf{u}} \ & \frac{1}{2}\Delta\mathbf{u}^\top \overline{\overline{H}}\Delta\mathbf{u} + \left[\tilde{z}[t]^\top \mathbf{r}^\top\right] \overline{\overline{F}}^\top \Delta\mathbf{u}, \\ \text{subject to: } & \Delta\mathbf{u}_{\min} \leq \Delta\mathbf{u} \leq \Delta\mathbf{u}_{\max}, \\ & \mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max}, \\ \text{with parameters: } & \tilde{z}[t] = \Psi(y[t]), \\ & \mathbf{r} \text{ given}, \end{aligned} \tag{5.18}$$

where $\Delta\mathbf{u} = [\Delta u[t] \ \cdots \ \Delta u[t+N-1]]^\top$ is the input sequence over which we perform the optimisation. The best such sequence is obtained at each time step t and its first element is applied. On top of constraining $\Delta u[t]$, we can incorporate constraints on $u[t]$ as well, by cleverly utilising the augmented state vector $\tilde{z}[t]$ and expressing $u[t]$ as $u[t] = \Delta u[k] + z^u[k]$. Specific values of $\Delta\mathbf{u}_{\min/\max}$, $\mathbf{u}_{,\min/\max}$ and other parameters particular to the optimisation problem's solution are provided in Appendix A. An illustration of the (K)MPC principle is in Figure 5.11a. We also depict the schematics of reference tracking solution using feedback interconnection of a KMPC controller with the system in Figure 5.11b

Tuning the KMPC again involves adjusting the state and input penalty matrices Q and R. However, in comparison to LQR, the number of tunable parameters increases by the prediction horizon length N and the final state penalty matrix S. This makes tuning the controller more challenging, especially considering the sensitivity of the Voxel Tower to these parameter values. To keep things concise, we only provide the submatrices Q_y, S_y, and R_y, corresponding to the output y of the system (first row with ϕ, second with $\dot{\phi}$). Since penalising previous states is not meaningful, elements of the matrices Q, S, and R, associated with the delayed variables are all zero. The

following are the final values that yielded the best performance.

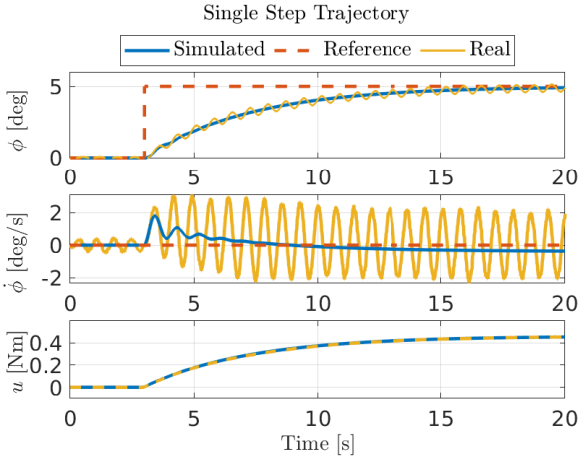
$$Q_y = \begin{bmatrix} 2 & 0 \\ 0 & 1\mathrm{e}^{-4} \end{bmatrix}, \quad S_y = \begin{bmatrix} 3 & 0 \\ 0 & 1\mathrm{e}^{-4} \end{bmatrix}, \quad R_y = 80000, \quad N = 10.$$

Now, having introduced the used control strategy, we can proceed to the description of performed experiments. To test the designed controller, we generated four reference trajectories, which primarily consisted of step or gradual changes of the desired angle. Throughout all the experiments, the reference velocity remained zero. This choice was made considering that the tower was intended to remain stationary or move slowly, and abrupt velocity changes, such as when mirroring a step, would only lead to poorer responses. Additionally, each experiment is accompanied by a simulation using the linear model.

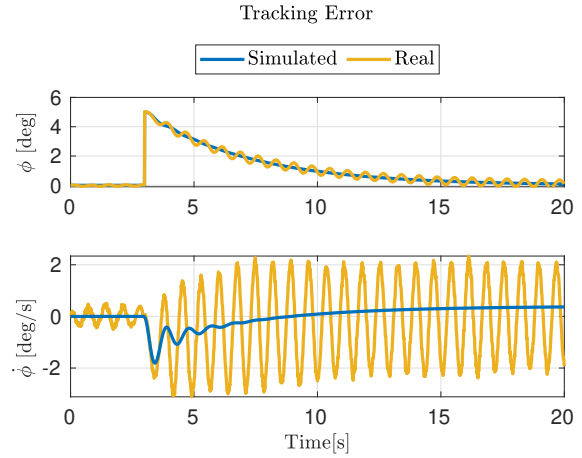
1) Step trajectory: The first type of trajectory involved a single step change in the tilt angle. This simple trajectory allowed us to perform a sanity check on the basic functionalities and assess the capabilities and limitations of the controller. We observed the maximal achievable steady deflection to be around ± 5 deg. This limitation was due to the constraint on the input torque, which in turn affected the maximum steady angle that could be achieved. The results of the experiment are presented in Figure 5.12.

2) Composed step trajectory: In these experiments, we required multiple consecutive step changes in the tilt angle. The key difference from the single step trajectories was that the structure might still be in motion when the next change in the reference arrives. Furthermore, we are more likely to encounter this kind of reference in real–world applications. For example, the system can be asked to periodically translate between multiple predefined positions. We conducted two experiments with trajectories of this type and present the measured responses in Figures 5.13 and 5.14.

3) Ramp trajectory: The third type of tested trajectories differs in that it no longer involves abrupt changes in the angle. Instead, the angle changes gradually. Such motion can be desired when tracking a moving object and tests a different behaviour of the system compared to the previous two types. In this case, we generated only one reference trajectory. The resulting behavior of the system is presented in Figure 5.15.



(a) : Comparison of the model and the real system.

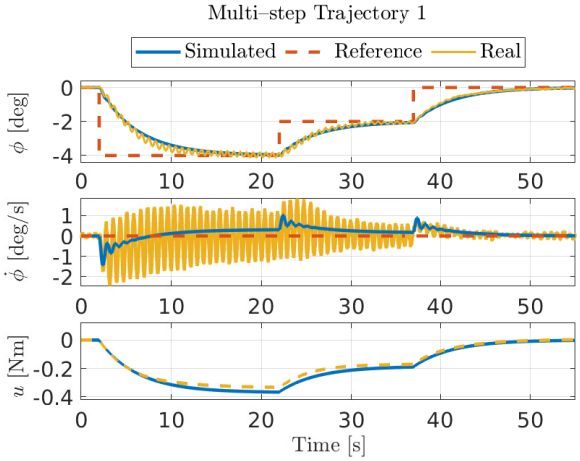


(b) : Tracking error in individual output variables.

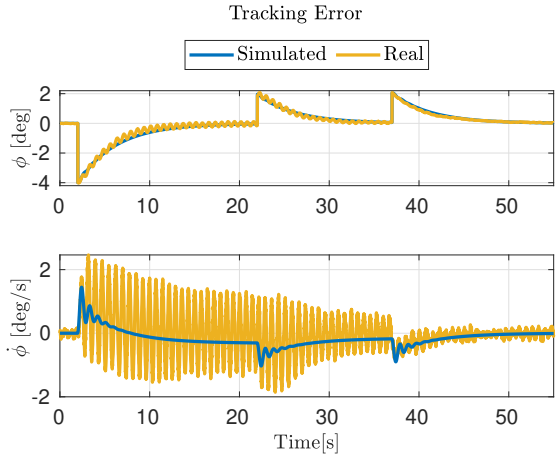
Figure 5.12: Single step reference tracking results.

5.5 Discussion

In this chapter, we addressed two control problems for the Voxel Tower: the problem of stabilisation and trajectory tracking. Firstly, we tested the robustness of the designed control strategy against disturbances and non–zero initial conditions. This is particularly relevant in scenarios where the structure needs to remain stable, such as tall antennas exposed to wind gusts or vibration damping in satellite booms. Secondly, we examined the system's ability to accurately execute motions prescribed in terms of the deflection angle $\phi(t)$ and angular velocity $\dot{\phi}(t)$. This has potential applications again in the field of signal transmission, specifically in antenna positioning rather than stabilisation.



(a) : Comparison of the model and the real system.

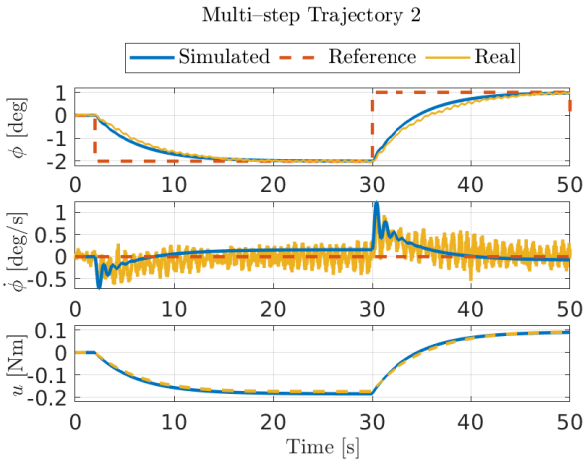


(b) : Tracking error in individual output variables.

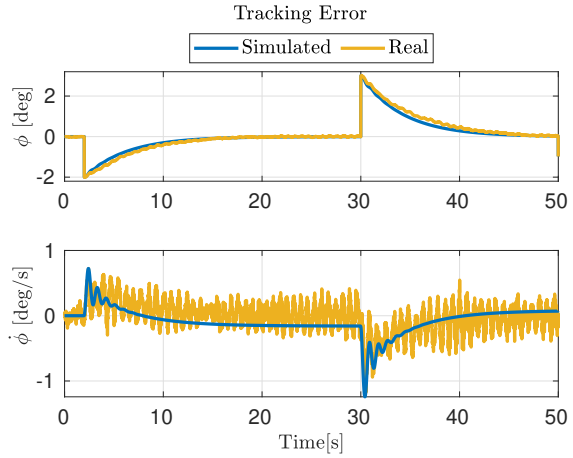
Figure 5.13: Multi–step reference tracking results, first.

1) Stabilisation: The results of all the experiments depicted in Figures 5.7, 5.8, 5.9, and 5.10 confirm the success of the stabilising LQR design. In each experiment, the controlled system consistently outperformed the autonomous (undamped) system in terms of settling time and oscillations damping.

- Focusing on the experiment involving the non–zero initial angle condition, as illustrated in Figures 5.7 and 5.8, we can draw several conclusions. In both experiments, the controlled system achieved a settling time of less than 1 second and maintained stability thereafter. The undamped system, in contrast, continued to oscillate, relying solely on its limited natural damping to eventually come to a stop. Additionally, this experiment highlights effectiveness of the designed controller in rapidly dampening both small and large initial deflections, exhibiting consistent performance varying initial conditions.



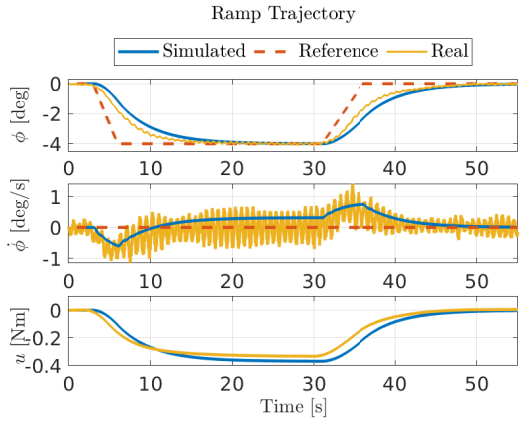
(a) : Comparison of the model and the real system.



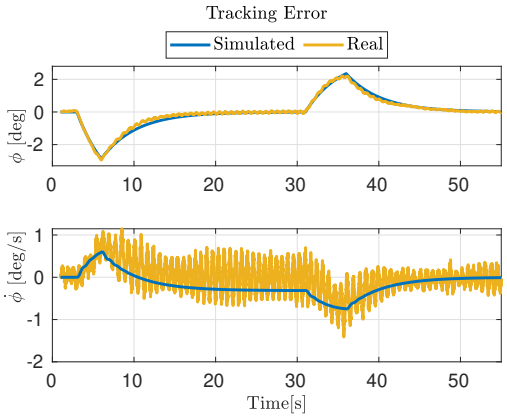
(b) : Tracking error in individual output variables.

Figure 5.14: Multi–step reference tracking results, second.

- The experiment involving non–zero inital conditions for both state variables demonstrates the ability of the LQR to dampen the system even when it is in motion. We can support this claim by referring to Figure 5.9, which compares the results of the controlled system with the autonomous response. The measured results indicate that the controller can bring the structure to a still position within a maximum of two seconds. However, we can note that the damping response is not as smooth as when the initial condition was steady (zero velocity). This may be attributed to the abrupt change in control action when transitioning to the LQR from the excitation signal, which likely triggered higher–frequency oscillations. These oscillations are also visible in Figure 5.10, where the disturbance was also an instantaneous change in the applied torque. Nevertheless, when compared with the autonomous system, the damped system again behaves in a superior manner, as the undamped system continues to oscillate with almost unchanged amplitude.



(a) : Comparison of the model and the real system.



(b) : Tracking error in individual output variables.

Figure 5.15: Ramp tracking results.

- The final stabilisation experiment aimed to assess the ability of the LQR to mitigate the effect of disturbances on the system. The measured responses once again show the successful dampening of oscillations induced by the disturbance. Similar to the experiments studying initial condition responses, the controller required approximately 1 second to recover from the disturbance and bring the system to a stable state. This recovery time roughly corresponds to one period of the resulting oscillations. In contrast, the undamped system was affected by the disturbance quite substantially.

  To improve the experimental setup, a suggestion arises regarding the generation of the disturbance. It is impractical that as of now, the disturbance and the control algorithm cannot act simultaneously. Therefore, in the future the disturbance could be introduced, e.g., through a moving platform on which the Voxel Tower is mounted. This approach would allow for testing how the controller behaves during the disturbance itself, rather than limiting its aftermath.

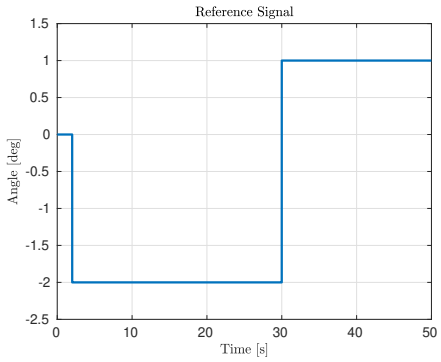
2) Reference Tracking: Having discussed the performance of the designed control algorithm in stabilising the system, we will now shift our focus to the second task: reference tracking. For this purpose, we decided to implement a KMPC controller. One of the reasons behind this choice was its natural incorporation of constraints on the input signals. To assess the performance of the controller in reference tracking, we generated three types of testing references.

- The measured response, depicted in Figure 5.12a, demonstrates the correct reaction of the system to a single step change in reference. The Tower effectively reaches and maintains the desired deflection. The tracking error in terms of $\phi(t)$ in Figure 5.12b reliably decreases to zero, indicating convergence to the reference signal. In contrast, the velocity mirrors the reference only loosely, which is expected since its deviation is not heavily penalised. However, increasing the penalty on velocity would lead to worsened response, as velocity is a noisy signal and the trajectory is primarily characterized by the angle.

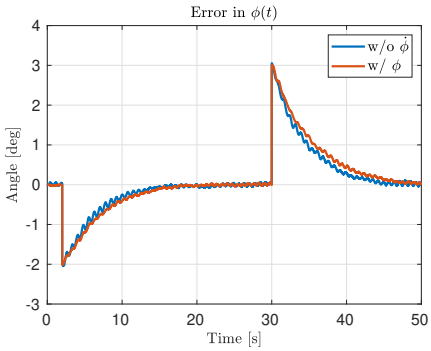
  Additionally, observing the Action Input subplot in Figure 5.12a, we can infer that the system approaches its limit when maintaining a steady deflection of 5 degrees, as indicated by the motor torque reaching close to its upper bound of 0.5 Nm.

- The conclusions drawn from Figures 5.13 and 5.14 build upon the the observations from the first experiment. In this scenario, the deflection angle $\phi(t)$ is expected to follow a reference signal consisting of a sequence of steps changes with varying amplitudes. In both of the currently discussed experiments, the system continues to successfully reach desired values of $\phi(t)$. Furthermore, the Figures 5.13b, and 5.14b also suggest that the pace at which the system reacts to each of the step changes is similar to the case with just one step and does not depend on the amplitude of the change. Notably, the behaviour predicted by the linear model aligns with the measured response of the system, with the exception of velocity. The measured velocity exhibits more oscillatory behaviour, which the model fails to capture. This observation further supports the decision not to heavily penalize velocity, as its fast–changing nature is not adequately represented in the model.

- In the final experiment, we aimed to evaluate the system's response to gradual changes in the reference signal. This is why, instead of step changes, we introduced ramps to the generated trajectory. Despite this change, the observed behaviour is not significantly different from what we observed in the previous experiments. The angle of the system steadily changes until it reaches the desired value. However, we did notice that the velocity oscillations, as depicted in Figure 5.15, have smaller amplitudes compared to, for example, the velocity in Figure 5.13. This can likely be



(a) : Reference angle.



(b) : Individual reference tracking errors.

Figure 5.16: Influence of including velocity on the model's oscillations.

attributed to the slower rate of change in the reference signal, resulting in a slower response of the system.

While the stabilising LQR performed very well and perhaps even exceeded our expectations in some of the tested scenarios, the hopes for the performance of the KMPC controller were higher. Although the system was able to track the desired reference, there are certain aspects of the results that leave room for improvement. A common denominator of all the measured responses is a relatively slow convergence to the reference (around 15 seconds) and an oscillatory response. Even though the amplitude of the oscillations in $\phi(t)$ is not particularly high, their presence showed to be problematic when attempting to achieve a more aggressive and faster response.

The KMPC controller's inability to dampen the observed oscillations may stem from their absence in the linear model. This led us to explore an alternative approach without incorporating velocity. Utilising delay embeddings, we created another model that solely relies on the delayed measurements of $\phi(t)$ and compared its response with the model that considers velocity. The comparison is visualised in Figure 5.16. As per the Subfigure 5.16b, excluding velocity did not bring significant improvement. Although the convergence to the reference value is slightly faster, the system exhibits the same amount, if not more oscillations.

Another possible explanation for the KMPC's inability to dampen these oscillations is the presence of delay in the structure. This delay may be attributed to the underactuated and flexible nature of the system, meaning it requires time for the motion to propagate from the actuator to the rest of the structure.

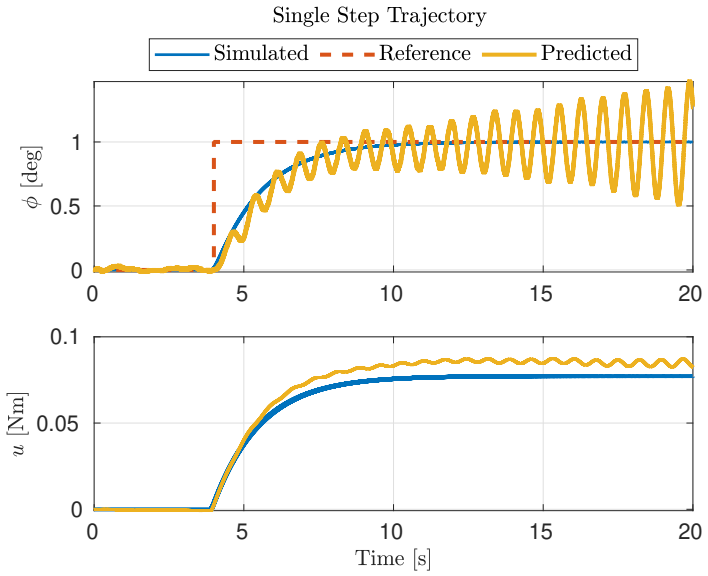


Figure 5.17: Increasing the prediction horizon destabilises the system.

The oscillations proved limitting when attempting to tune a faster response of the system. While we were able to tune almost an ideal simulated response, in reality, faster changes in the deflection angle led to excessive oscillations that rendered the controller ineffective. This is depicted in the Figure 5.17 where we attempted to improve the behaviour by adjusting the prediction horizon to $N = 20$. Despite the expected improvement, in reality, the system became unstable. Tuning the KMPC with such scenarios proved to be a delicate task, as striking a balance between tracking performance and excessive oscillations was challenging.

3) Summary and Possible Improvements: In summary, we have successfully achieved both of our primary objectives. The stabilisation task was effectively addressed using the damping LQR, which demonstrated impressive results in damping oscillations and mitigating disturbances within a time frame of approximately one second. Additionally, in terms of tracking performance, we were able to successfully follow a changing angle trajectory. However, one significant challenge remained unresolved: the presence of small but persistent oscillations in the system. These oscillations limited our ability to tune a faster and more aggressive response.

Regarding stabilisation, there are following potential improvements to consider. First, it would be beneficial to explore a different source of disturbance. As mentioned earlier, mounting the Voxel Tower onto a shaking bed would allow us to observe the system's behavior also during the disturbance, rather than solely after it. Additionally, the experiment in Figure 5.9 cound be

reiterated, this time with a more aggressive input that would generate higher velocities provide further valuable insights.

In connection with tracking, we can suggest a couple of improvements to enhance the behaviour of the system. First, creating a better model could be beneficial. This can involve fitting a new DMDc model with different parameters or collecting new data to better describe the behaviour of the Voxel Tower. Secondly, using a higher–quality sensor with less noise and more stable readings could be helpful. Although the choice for an off-the-shelf MPU9250 IMU sensor was driven by scalability and accessibility, a better sensor would result in smoother and more accurate measurements. Lastly, exploring alternative control strategies is also worth considering. While MPC was chosen initially for its ability to incorporate input constraints, other options such as an LQI controller (LQR enriched with integral control) could yield improved performance, considering the satisfactory results obtained with the stabilizing LQR. These potential improvements and changes can be considered for future work.

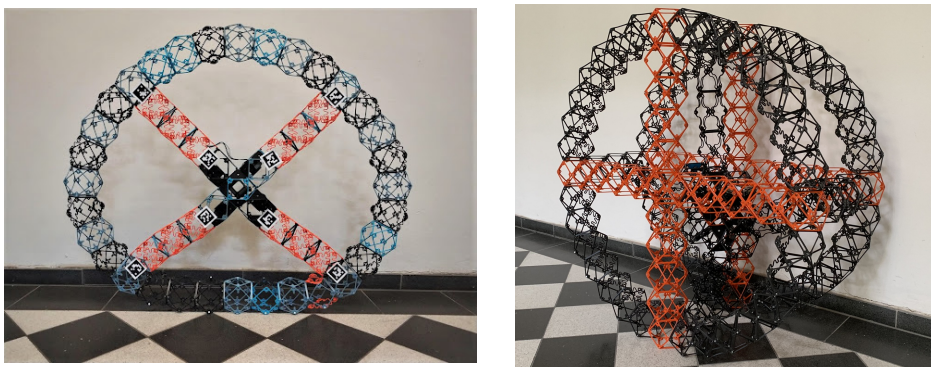
Chapter 6

Robot ŽůžO

The Voxel Tower discussed in the previous chapter was a good platform for testing control algorithms. It has a simple, well defined beam structure that makes the system easy to approach and grasp. Yet it gives rise to a number of challenging and interesting control problems that can be solved. Despite all of that, it does not really test the limits of the building material. This is where we introduce ŽůžO – a mobile, rolling robot built from digital material. Because of the robot's structure, we put the material to the test, subjecting it to constant deformation and varying stress and strain. Furthermore, as ŽůžO rolls, the material frequently makes contact with the ground, resulting in increased wear and potential damage to the material. All the above makes ŽůžO intriguing not only for control engineers but also as a means to asses the suitability of digital materials for constructing intricate mobile structures.

ŽůžO (Fig. 6.1) consists of two main components: an outer rim enabling rolling motion and a central block housing actuators, a battery, and control logic. The central block is attached to the rim through flexible segments. Their flexibility plays a vital role in the robot's chosen motion principle based on [MKB+15]. ŽůžO achieves motion by shifting its *center of mass* through manipulating the position of the central (heavy) block. This is accomplished through four tendon–based actuators, each located within one of the flexible segments. Our motivation to rely on tendons was that it was the easiest way to achieve the necessary compression of the flexible segments.

The first iteration of the robot was constructed by a four–member team as part of the Team Project course at CTU. The team consisted of Dominik Hodan, Konstantin Khoklov, Marina Ionova, and myself. I then continued



(a) : First iteration of the robot.

(b) : Current state of ŽůžO .

Figure 6.1: ŽůžO robot.

working on the robot and rebuilt it into its current state. The first version of the robot (Fig. 6.1a) had several construction weaknesses that severly limited its rolling motion capabilities. The contracting segments were overly rigid, limiting the mobility of the central block. In contrast, the rim was too compliant, leading to insufficient support for the weight of the entire structure. Another significant drawback of the first iteration was the single–rim design, confining all components to a single plane. Consequently, manual suport was necessary to prevent the robot from toppling or losing balance. These issues led to the need for a refined version of the robot that would no longer suffer from these drawbacks. To address the issue of insufficient compliance in the segments, we opted for a flexible filament during printing. This not only resolved the stiffness problem but also prevented the segments from breaking when subjected to greater compressions. Subsequently, we reinforced the rim with rigid beams that effectively bear the weight of the structure (orange crosses in the Fig. 6.1b). These supports resulted in ŽůžO maintaining a round shape, thus improving its rolling capabilities. Last but not least, we added a second rim to enhance the robot's stability and enable independent function without external support. These modifications culminated in an improved version of the robot, possessing enhanced properties. The current state of the robot is shown in the Figure 6.1b.

At present, the robot can be operated remotely using a joystick. The joystick's position serves as a reference for adjusting the position of the central block, taking into account the robot's current orientation measured by the installed IMU. Given a joystick position $[x_\mathrm{j}\ y_\mathrm{j}]^\top_\mathrm{m}$ and orientation θ, we can compute the action inputs for individual actuators $\in$ {A,B,C,D} as

$$u_\mathrm{A} = x_\mathrm{j}\sin(\theta) - y_\mathrm{j}\cos(\theta), \quad u_\mathrm{C} = -u_\mathrm{A},$$
$$u_\mathrm{B} = x_\mathrm{j}\cos(\theta) + y_\mathrm{j}\sin(\theta), \quad u_\mathrm{D} = -u_\mathrm{B},$$

using the labels from Fig. 4.3 and ensuring that motors C and D are acting opposite to their respective counterparts.

Having a physical realisation of the robot that can now roll motivates us to pursue the design of automatic control algorithms. Although the robot can currently be operated manually, it is not yet prepared for the deployment of control algorithms. Hence, during their implementation, we will solely rely on the simplified nonlinear model presented in Section 4.2 for validation purposes. To develop the aforementioned algorithms, we will once again employ a linear model obtained through DMDc, as discussed in Section 4.3.

6.1 From Nonlinear Model to DMDc

Due to not having the option to test the algorithms on the real robot, we will resort to utilising the nonlinear model instead. To recapitulate, the nonlinear description of ŽůžO uses six states

$$\begin{aligned}
\theta \text{ [rad]} &= \text{the rolling angle,} \\
\dot{\theta} \text{ [rad/s]} &= \text{the rolling velocity,} \\
x_\text{m} \text{ [m]} &= x \text{ position of the central block,} \\
\dot{x}_\text{m} \text{ [m/s]} &= x \text{ velocity of the central block,} \\
y_\text{m} \text{ [m]} &= y \text{ position of the central block,} \\
\dot{y}_\text{m} \text{ [m/s]} &= y \text{ velocity of the central block,}
\end{aligned}$$

four inputs and two outputs. The outputs consist of the horizontal position of the center of the robot x [m] (not the position of the block) and its translational velocity $\dot{x}$ [m/s]. For more insight into the meaning of the variables, we refer to Figure 4.3.

First step in obtaining the linear model is to collect identification data. We performed 20 experiments on the nonlinear model, with each experiment lasting 10 seconds. During the experiments random inputs were applied to each motor and we sampled system with period $\text{T}_\text{s} = 0.05$ s. The initial orientation of the robot, always set to $\theta = 0$ rad, holds significant importance. This is because different motors are engaged for achieving the same movement depending on the robot's orientation. For instance, when the robot wants to start rolling to the *right*, it utilises different motors when $\theta = 0$ rad then when $\theta = \pi$ rad. Furthermore, when the robot is rolling, it constantly switches the engaged motors, to keep the central block in the desired position. Initially, we intended to introduce perturbations in the starting angle during the identification experiments to incorporate this behaviour into the linear model. However, this approach led to poor performance. Consequently, we

decided to limit our range of operation to small angles ($|\theta| \lessapprox 0.5$ rad) where the model showed to be sufficiently precise, when tuned around $\theta = 0$ rad.

After converting the identification data into matrices X_{lift}, X'_{lift} and Ω_{lift}, we created a linear model for the system. Surprisingly, we observed that the model achieved reasonable predictions of the nonlinear states, even without the need for delay embeddings (unlike the case with Voxel Tower). In light of that, we can construct the following the following linear predictor

$$\begin{aligned} z[k+1] &= Az[k] + Bv[k], \\ \hat{y}[k] &= Cz[k], \end{aligned} \tag{6.1}$$

where $z = \begin{bmatrix} \theta & \dot{\theta} & x_{\text{m}} & \dot{x}_{\text{m}} & y_{\text{m}} & \dot{y}_{\text{m}} \end{bmatrix}^\top$. The outputs of the system are the horizontal displacement and velocity of the rim. Assuming the robot moves without slipping, the following matrix C enables the determination of these outputs.

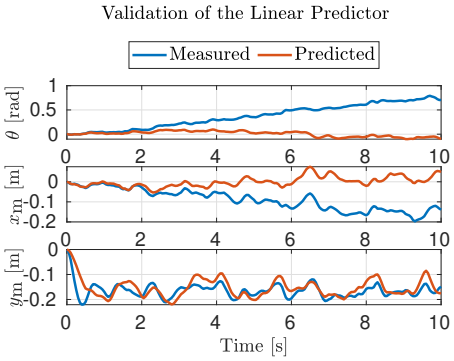
$$C = \begin{bmatrix} -r & 0 & 0 & 0 & 0 & 0 \\ 0 & -r & 0 & 0 & 0 & 0 \end{bmatrix}$$

We utilise the robot's radius $r = 0.46\,\text{m}$ to transform between rotational and translational *positions* and velocities. The comparison of the predicted and nonlinear states is shown in the Figure 6.2a. To keep it concise, we have chosen to display only the states θ, x_{m}, and y_{m}. The excitation input is visualised in the Figure 6.2b. Upon studying the Figure 6.2a, we observe that the predictor slowly diverges from the measured values, particularly in θ and x_{m}. However, this is not a significant concern for us, as we will use the model with KMPC. For KMPC, only the model's performance on a short horizon is crucial. The predictions of the model in Figure 6.2 remain accurate for approximately 2 seconds, which has proven to be sufficient for succesful controller tuning.

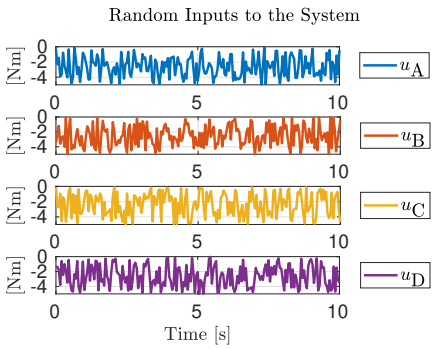
To gain more insight into the behaviour of the model, we implemented a visualisation tool using `MATLAB`. This tool enables us to replay the conducted experiments and facilitates a better understanding of the data by providing a way for interpretting it within the context of the whole robot. The graphical output of this visualiser is in Figure 6.2c.

6.2 Reference Tracking

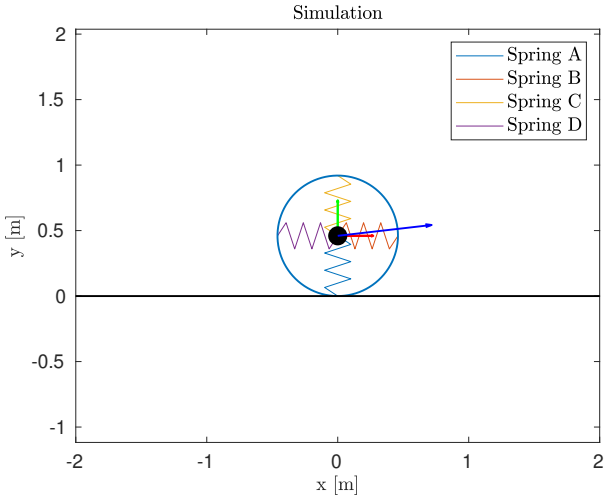
For mobile robots, the ability to track reference is crucial. This is because the task of navigating from point A to point B is typically divided into



(a) : Validation of the model.



(b) : Excitation signals.



(c) : Implemented visualisation.

Figure 6.2: Simulations and visualisations of ŽůžO .

three subproblems: identifying the goal, planning the path, and employing a reference tracking algorithm to accurately follow the planned path towards the goal.

We decided to use the same tools we opted for when we were solving the trajectory reference problem for the Voxel Tower in Section 5.4. That is, the KMPC algorithm. Having discussed KMPC in detail in Section (5.4.1), we will now focus on highlighting the differences resulting from the unique structure of ŽůžO and presenting the results.

The main distinction lies in the computation of the KMPC matrices $\overline{\overline{H}}$ and $\overline{\overline{F}}$ from Equation (5.18), where we employ the predictor (6.1). This leads to matrices with different dimensions and entries, while the rest of the KMPC tuning procedure remains unchanged. Furthermore, unlike the Voxel Tower, ŽůžO has multiple inputs on top of having more than one output.

The KMPC weights and parameters that yield the best observed behaviour of the nonlinear model are

$$Q = \begin{bmatrix} 300 & 0 \\ 0 & 10 \end{bmatrix}, \quad S = \begin{bmatrix} 100 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad R = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}, \quad N = 20.$$

The weights in matrices Q and S correspond with the horizontal displacement $x[k]$ and velocity $\dot{x}[k]$ of the wheel and matrix R penalises changes in individual inputs. N is again the prediction horizon.

We conducted two testing scenarios to verify the controller's performance. Each scenario defined a trajectory for ŽůžO to track, taking into account the requirement to operate near $\theta = 0$ rad. The experimental results, presented in Figures 6.3 and 6.4, include simulations of both, the nonlinear model as well as the linear predictor.

6.3 Discussion

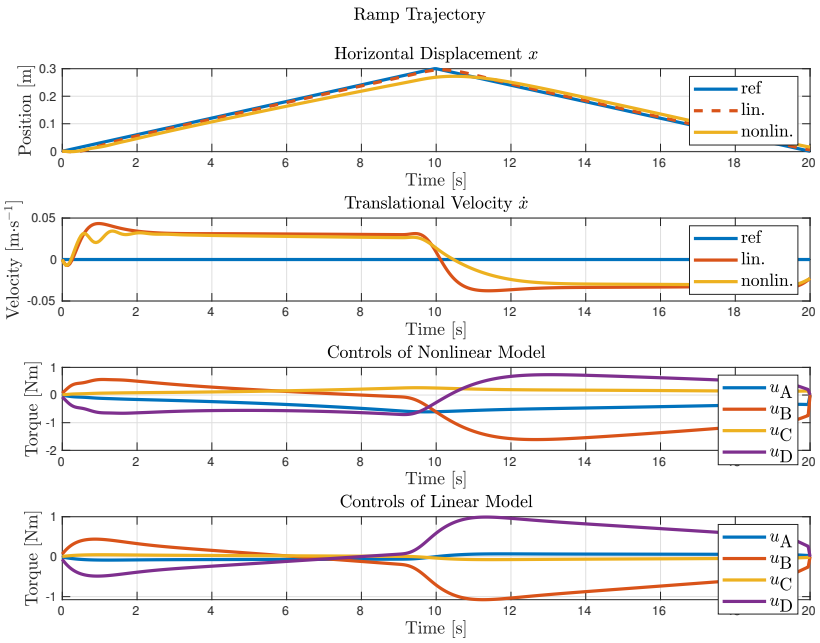
In this chapter, we introduced a rolling robot called ŽůžO, which represents another application of digital materials. Initially, we encountered several challenges with the robot's first iteration. We did not use the optimal filament for printing some of the voxels, and the overall structure required significant refinement. Additionally, during the initial experiments, many voxels broke, often leading to the destruction of almost the entire robot. These difficulties nearly led us to consider abandoning the concept of a rolling voxel-based robot due to doubts about the material's ability to support the structure. But upon implementing the aforedescribed structural improvements, such as utilising different filament for printing and reinforcing the rim of the wheel, the robot started demonstrating promising functionality. We even had the opportunity to showcase the new remote–controlled version at MakerFaire in Prague to promote our faculty.

The main focus of this chapter was the development and subsequent testing of a reference tracking algorithm for ŽůžO. The performance of the controller is demonstrated in Figures 6.3 and 6.4. Since the physical deployment of control algorithms on the robot was not yet feasible, we validated the controller using its simplified nonlinear model. For that, we created two reference trajectories: an increasing and decreasing ramp signal and an oscillating sine wave. To ensure the validity of our linear model, we restricted the range of θ to $|\theta| \lessapprox 0.5$ rad. Both the controlled nonlinear and linear models

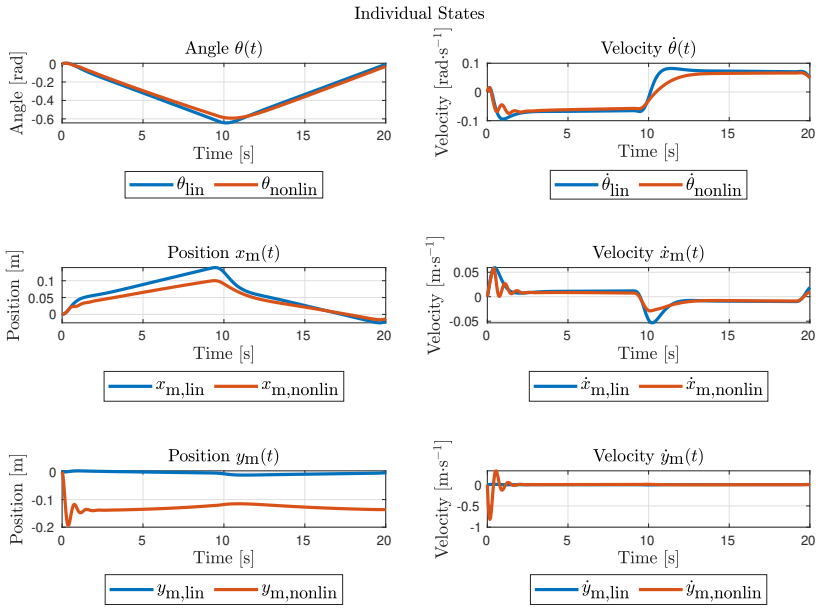
successfully followed the slowly changing ramp as well as the faster sine wave. Additionally, the linear model exhibited precise predictions of the nonlinear outputs x and $\dot{x}$, supporting our claim that it is sufficient for designing control algorithms, despite the deviations observed towards the end of the experiment in Figure 6.2a.

Comparing the predicted and nonlinear states in Figures 6.3a and 6.4a, we can make the following observations. Most of the predicted states show varying levels of agreement with their nonlinear counterparts, but they still follow the same trends. This is unfortunately not the case with predictions of y_{m} as they seem to be completely wrong. This discrepancy is attributed to the influence of gravity acting downward in the y axis. It is surprising that the DMDc model did not account for the influence of gravity, especially considering that gravity was present in the identification data. Furthermore, it correctly predicted its effect on the y_{m} state in the Figure 6.2a. However, even with this discrapancy in predictions of y_{m}, the system was still able to track the reference accurately.

The successful implementation of a KMPC controller for the nonlinear model of ŽůžO paves the way towards further pursuing the automatic control of the real robot. To achieve that, there are still several tasks that need to be addressed. First, regarding the nonlinear model. The model captures the fundamentals of the robot's dynamics, but a more precise identification of individual parameter values is needed for developing control algorithms specific to the real robot. This entails determining the exact values of parameters governing its behaviour, ensuring a stronger alignment between the model and the robot's actual characteristics. The limited validity of the linear model is also an issue. In order to resolve that, we could e.g. use different algorithms from the field of nonlinear control or use control strategies for hybrid systems, such as switching between multiple linear models, based on θ.

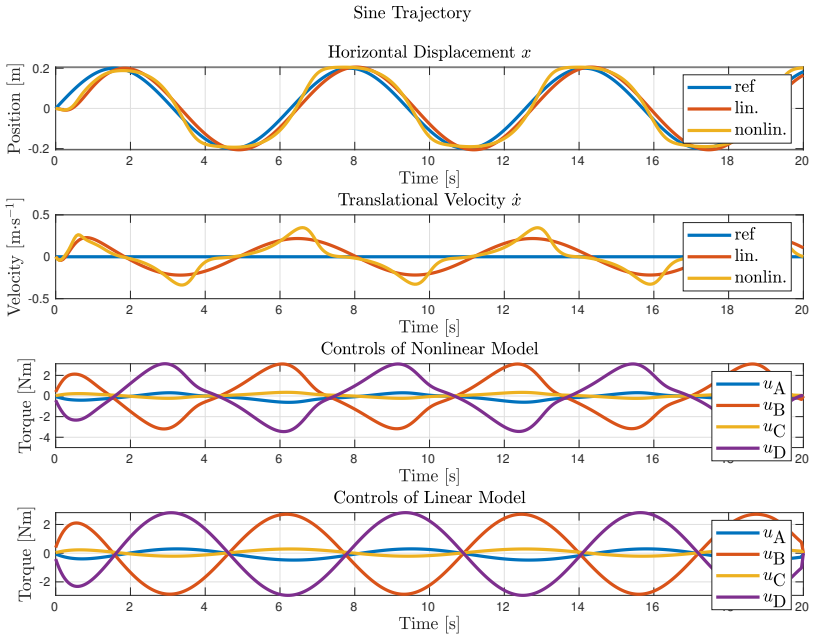


(a) : Outputs and control action.

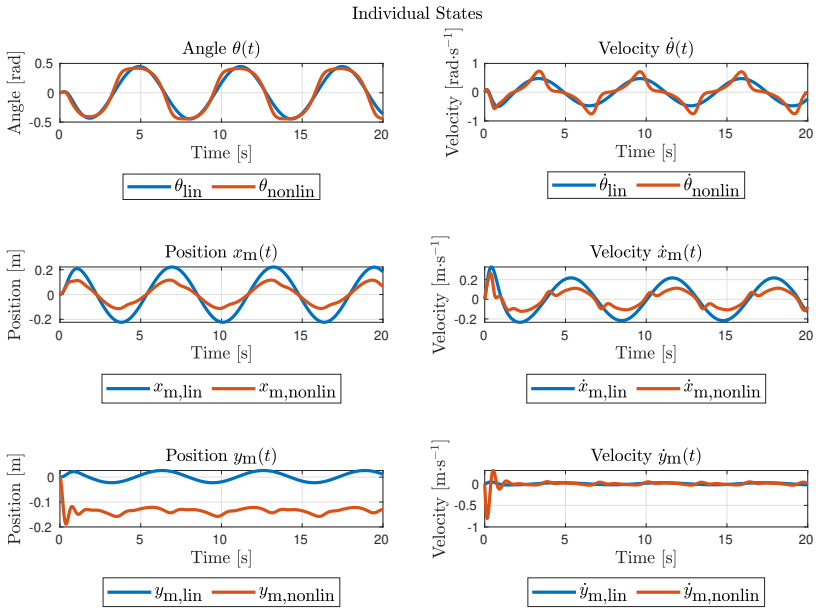


(b) : Comparison of the linear and nonlinear states.

Figure 6.3: Ramp tracking results.



(a) : Outputs and control action.



(b) : Comparison of the linear and nonlinear states.

Figure 6.4: Sine wave tracking results.

Chapter 7

Digicomb

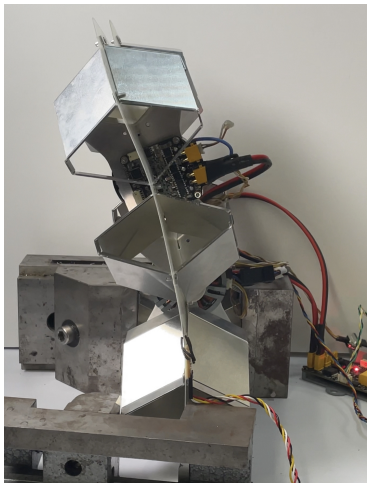
Both the aforedescribed structures were constructed using mechanical metamaterial heavily inspired by the work published by the CBA group at MIT [Jen20, JCT+20, PRFJ+23]. The focus of this work was to introduce automatic control into structures built from this material. The concept of feedback–controlled structures was intriguing to both our group and the CBA group, leading to a collaborative effort during our one-week visit to Boston, Massachusetts, USA. The visiting group consisted of Jiří Zemánek, Krištof Pučejdl, Adam Uchytil, and myself from the CTU. The results presented in this chapter reflect the joint effort of our group mainly in collaboration with Alfonso Parra Rubio from CBA. During our time at MIT, I was mainly occupied with setting up the interface between the sensors and motors as well as performing the experiments and obtaining the linear model of the structure.

Alfonso is currently exploring the potential of using a kirigami–like principle, involving cutting and folding hylite plates, to create structures that offer greater endurance and higher load bearing capacity than the voxel–based structures described in the previous chapters. These folding metamaterials were discussed in Section 3.4, with the promise of utilising them in high–strain applications where other materials may fail. One potential application is the design of a morphing aircraft wing, building upon the research presented in [PRFJ+23], which also dealt with a deformable voxel–based wing.

In the initial stages, our focus was primarily on testing the actuation principle, deformation sensing and gaining better understanding of the properties of the material. Digicomb, depicted in Figure 7.1 is described in Section 7.1.

(a) : Honeycomb. A remote resemblance is there![a]

[a]https://coloradohemphoney.com/



(b) : Motors bending the real Digicomb structure.

Figure 7.1: Honey– and Digicomb.

Subsequently, the performed experiments are outlined, and the measured data is discussed in Section 7.2. Finally, we highlight potential vectors for further research.

In this structure, the interconnectedness of a honeycomb meets digital material. The name 'Digicomg' is derived from two elements: 'honeycomb' and 'digital' and was chosen to expresses the distinct visual resemblance between the structure and a honeycomb, while also highlighting its digital nature.

7.1 Assembled Prototype

The main components of the assembled prototype (Fig. 7.1b) include a flat fiberglass centroid and DFM hylite elements (Section 3.4). The structure consists of three DFM blocks mounted on each side of the centroid, with the middle element thinned to facilitate bending. We drive the structure using two 5208 brushless motors attached directly to the hylite blocks. This direct drive configuration offers advantages such as eliminating parasitic effects and losses from gears or tendons. The actuators are distributed to allow for actuation and bending along the entire length of the centroid. One of the further design goals was for the structure to be self–supporting, meaning it should bear its own weight without relying on the actuators to carry it. While this goal has not been fully achieved, as the structure still bends when held horizontally, the actuators can exert sufficient torque to overcome gravity.

Additionally, when the structure is mounted vertically, this bending issue is not a concern. To measure deformation, we attached two strain gauges (SG) to opposite sides of the centroid, across from the motors. Utilising these strain gauges we can precisely measure even small deformations. However, processing the measured data still presents some unresolved issues. We utilise the HX711 SparkFun[1] amplifier to enable the readings of the strain gauges. For communication with the amplifier we use a Raspberry Pi C++ library[2]. However, the library is primarily intended for building digital scales, so the interpreted values are not directly applicable to our application. As a result, we are currently reading raw numbers without interpretating them as voltage ratios. Despite that, the obtained values adequately reflect changes in the structure, as demonstrated by the graphs in Figure 7.2.

Based on the data gathered during the experiments, which are described in Section 7.2, we created a mathematical model, again utilising the DMDc method (Section 4.3). This resulted in a discrete state description of form

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k], \\ y[k] &= Cx[k] \end{aligned} \tag{7.1}$$

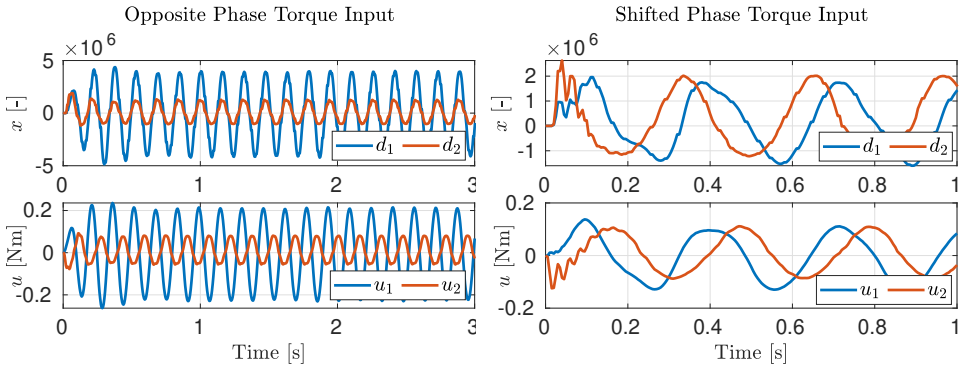
where $x = [d_1\ d_2]^\top$ is the state vector containing the strain gauge displacements d_1 and d_2. $u = [u_1\ u_2]^\top$ is the vector of inputs. This time, no delay embeddings were required to obtain a model that sufficiently captures the dynamics of the system, as is showcased in the Figure 7.3. Because space and the dimensionality of the system permits it, we provide the numeric values of individual state matrices below.

$$A = \begin{bmatrix} 0.7431 & 0.0308 \\ -0.0539 & 0.9114 \end{bmatrix}, \quad B = 1\mathrm{e}^6 \cdot \begin{bmatrix} 3.9500 & -1.1192 \\ 0.9022 & -1.6267 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

7.2 Discussion

Although achieving a morphing wing is still a distant goal, our limited time in Boston allowed us to assemble and test the initial downsized prototype, marking an important first step and serving as a proof of concept. The direct drive configuration of the motors proved suitable for actuation, as they provided sufficient torque to deform the structure (Fig.7.1b). Moreover, this configuration enables rapid propagation of torque changes, as demonstrated

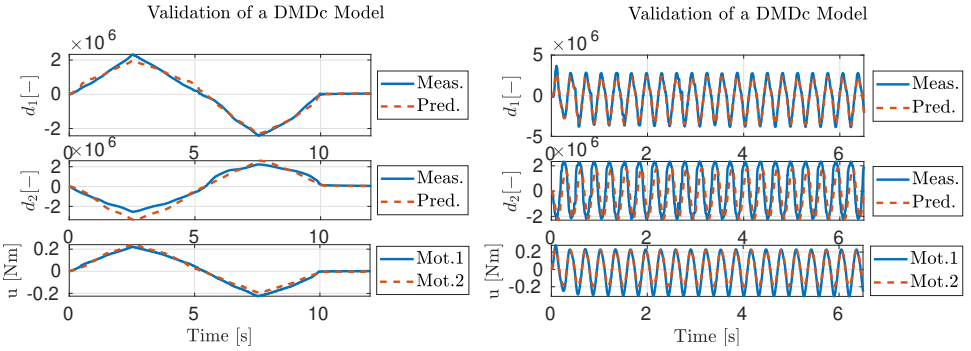
[1] https://www.sparkfun.com/products/ 13879
[2] https://github.com/ endail/hx711



(a) : Response to opposite excitation signals with opposite phases.

(b) : Close up on the response to an abrupt change in torque.

Figure 7.2: Testing of actuation and sensing.



(a) : Predictions with an in–phase ramp input.

(b) : Predictions with an in–phase sine input.

Figure 7.3: Validation of the DMDc model.

in Figure7.2b, where the reference torque of motor 2 transitions from 0 to -0.05 Nm at $t = 0$ s.

Using strain gauges to measure the induced deformation also seems to be a reasonable decision as the gathered data show very clear dependence on the applied torque. It is interesting to note that the *mirror* mounting of the motors means that when the motors act with opposite phases (Fig. 7.2a) the resulting measured values have the same sign. This means that opposite motor torques result in the structure transforming to an 'U' shape (when in horizontal position) and indeed, this shape corresponds with the same direction of deformation for both strain gauges. The sensors even succeed in measuring the fast transients in Figure 7.2b. There is still work left to do in the processing of the signal, but the employment of these sensors is justified.

Regarding the conducted experiments, the majority of them describe a similar setting as the experiment depicted in Figure 7.2. Specifically, they focus on studying the system's behavior in an open–loop setup. We applied a predetermined torque, and observed the resulting response of the system. These experiments primarily aimed to assess whether the applied torque was capable of achieving the desired deformation and to evaluate the reliability of the gathered sensor data.

We utilised the measured data to create a linear model capable of predicting the behaviour of the system. This model exhibits sufficient accuracy in capturing both slow transients (see Figure 7.3a) and faster dynamics (Fig. 7.3b). Although further validation on additional data is necessary, based on the measured responses, the model adequately represents the system's behaviour.

In conclusion, the structure described above is not a fully developed project or a functioning prototype of a morphing wing. However, it shows the successful completion of initial steps within a limited timeframe, providing a foundation for further study and development of the system. Although there are still some structural flaws, such as imperfect motor mounts, these can be easily addressed in future iterations. Unfortunately, due to the limited time at MIT, we were unable to test any feedback control algorithms on the structure. Therefore, the next logical step would be to utilise the mathematical model (Eq. (7.1)) to design a controller that can achieve a predefined shape for the structure. Additionally, exploring larger versions with more actuators and sensors would be a natural progression. Nonetheless, Digicomb represents one of the first dynamically controlled structures with integrated sensing, constructed from the discrete folded digital material. It opens the door to a an interesting project with a lot of future potential.

Chapter 8

Conclusion

In this work, we studied mechanical digital metamaterials as building components for dynamic structures and mobile robots. Namely, we focused on three structures: the Voxel Tower, robot ŽůžO, and Digicomb. To advance the capabilities of the structures, we integrated digital materials with algorithms of optimal control. Specifically, we employed LQR for stabilisation and damping, while KMPC served as our choice for trajectory tracking. The effectiveness of both algorithms was verified through numerous tests, including experimentation with a real prototype of the Voxel Tower. Although trajectory tracking for ŽůžO was limited to simulations, we addressed various aspects of control system development, including the task of model creation. By focusing on each structure individually and extracting general insights, our findings contribute to the broader field and aim to help with future development of one unifying framework for controlling such structures.

We began this work by summarising the current state of the art in the field of controlled metamaterials and flexible structures. Additionally, we outlined the primary objectives of this thesis, which involved studying the feasibility of digital material for constructing dynamical structures and developing control strategies for them. In Chapter 3, we described the two groups of digital materials employed in this thesis, namely 3D printed voxels and folded hylite plates. Notably, our work involved developing an actuated voxel: a means for actuating the structure on the level of a single voxel. This novel method aligns more closely with the philosophy of digital materials compared to the use of tendon–based actuators, which are typically employed in voxel–based structures. Next, we detailed three modelling approaches that we explored throughout this thesis. We established that the most general approach to modelling was the data–driven DMDc which has been successfully applied to

all of the studied structures. On the other hand, we have not had much success in modelling the systems using existing simulation software, primarily due to the numerous unknown parameters involved. We also explored the process of abandoning the discrete nature of the system and using a conventional modelling approach of Lagrangian formalism, resulting in a simplified model of ŽůžO. This approach proved useful when complemented with the DMDc model, providing a basic means of validation for the designed controller. We dedicated the Chapters 5, 6 and 7 to describing the individual structures that we studied.

The biggest spotlight was on the Voxel Tower, described in Chapter 5. We explored the influence of identification data on the resulting linear predictor and tackled two control problems: stabilisation and reference tracking. For stabilisation, we used LQR tuned on the linear model obtained with the DMDc method. This controller effectively dampened oscillations and attenuated disturbances within a short timeframe of 2 seconds, hence proving the design of a stabilising algorithm successful. To achieve reference tracking, we adopted the KMPC due to its natural ability to integrate our linear model and constraints. Although the system's response was not as fast as desired and exhibited small, yet persistent oscillations, the controller successfully tracked the reference signals. The amplitude of the reference signals was further limited by the bounds on the driving torque of the motor. Future improvements could involve generating better identification data and employing alternative control strategies. Additionally, for stabilisation, mounting the Tower on a shaking bed could be explored as a better means of generating disturbances.

In Chapter 6, we focused on the rolling robot ŽůžO, which (not intentionally) served also as a platform for testing the suitability of individual voxels for constructing complex dynamical structures. We discussed the process of obtaining a DMDc linear predictor by utilising a nonlinear simplified model from Section 4.2. This allowed us to design a KMPC controller for the simplified nonlinear model and to validate it through simulations, since the physical ŽůžO robot was not equipped for deploying control algorithms. The KMPC controller successfully tracked the reference trajectories, when staying in the range of θ where the linear predictor provided an accurate approximation of the nonlinear system. Future work includes rigorous parameter identification of the nonlinear model, verification against the real robot, and preparing the real robot for implementing the designed control algorithms.

Finally, in Chapter 7 we introduced the third structure which was created in collaboration with the CBA group at MIT. However, we did not go much beyond the introduction. We presented some experiments performed on the real structure and validated its linear model. However, due to the limited time

during our visit at CBA, we were unable to try any feedback control algorithms. Therefore, this chapter served as a foundation for future collaborative work.

Despite the distinct characteristics of each structure, whether in terms of material or type, we identified relevant observations that were consistent across all the studied structures. These findings should be taken into account when seeking a unified approach to controlling lattice–based structures. The first observation we made relates to the model creation process. It is crucial to employ a versatile framework that can accommodate various types of measurements and structures with minimal reconfiguration. In this work, we utilised the DMDc algorithm and based on our gathered experience we see data–driven modelling as a promising direction to explore in the context of modelling general lattice–based structures. DMDc proved capable of modelling diverse structures: static and mobile, incorporating data from various sensors (IMU, strain gauge, etc.), just with different identification data. Concerning the control algorithms, we employed KMPC to reference tracking for two completely different structures. However, the selection of the controller was primarily driven by the task rather than the underlying structural similarities. In conclusion, finding a unifying approach is a challenge that demands substantial investment of time and effort. However, the initial steps taken in this work, including exploring various approaches to specific subproblems and gaining familiarity with the building material, have contributed to making this challenge more approachable.

To summarise and recapitulate, among the main contributions of this work was the development of feedback control algorithms for lattice–based structures and the subsequent validation of said algorithms. The implementation of these control strategies has successfully expanded the capabilities of mechanical metamaterial structures beyond open–loop control and relying on precomputed input sequences. While there is still room for improvement in certain aspects, the presented work opens the door to pursuing applications that demand more advanced control techniques. Throughout this thesis, we gained valuable insights regarding the nature of mechanical metamaterials and their potential for constructing dynamic structures. Collectively, these observations serve as a stepping stone towards the development of a unified framework for modelling and controlling lattice–based structures, offering a promising path for future advancements in this field.

Bibliography

[AKM18] Hassan Arbabi, Milan Korda, and Igor Mezic, *A data-driven Koopman model predictive control framework for nonlinear flows* (en).

[Ash05] M.f Ashby, *The properties of foams and lattices*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **364** (2005), no. 1838, 15–30, Publisher: Royal Society.

[AWP20] Marc Artola, Andrew Wynn, and Rafael Palacios, *Modal-based Model Predictive Control of Multibody Very Flexible Structures*, IFAC-PapersOnLine **53** (2020), no. 2, 7472–7478.

[Bal78] M. J. Balas, *Active control of flexible systems*, Journal of Optimization Theory and Applications **25** (1978), no. 3, 415–436 (en).

[BBA+11] Silvestro Barbarino, Onur Bilgen, Rafic Ajaj, Michael Friswell, and Daniel Inman, *A Review of Morphing Aircraft*, Journal of Intelligent Material Systems and Structures - J INTEL MAT SYST STRUCT **22** (2011).

[BBKK21] Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz, *Modern Koopman Theory for Dynamical Systems*, October 2021, arXiv:2102.12086 [cs, eess, math].

[BBMR87] M. Basseville, A. Benveniste, G. Moustakides, and A. Rougee, *Optimal sensor location for detecting changes in dynamical behavior*, IEEE Transactions on Automatic Control **32** (1987),

no. 12, 1067–1075, Conference Name: IEEE Transactions on Automatic Control.

[CDM17] Jordan Cheer, Stephen Daley, and Cameron McCormick, *Feedforward control of sound transmission using an active acoustic metamaterial*, Smart Materials and Structures **26** (2017), no. 2, 025032 (en), Publisher: IOP Publishing.

[CG13] Kenneth C Cheung and Neil Gershenfeld, *Reversibly Assembled Cellular Composite Materials*, 2013.

[CTdR+16] Corentin Coulais, Eial Teomy, Koen de Reus, Yair Shokef, and Martin van Hecke, *Combinatorial Design of Textured Mechanical Metamaterials*, Nature **535** (2016), no. 7613, 529–532 (en), arXiv:1608.00625 [cond-mat].

[CZL20] Zhenyu Chen, Weijian Zhou, and C.W. Lim, *Active control for acoustic wave propagation in nonlinear diatomic acoustic metamaterials*, International Journal of Non-Linear Mechanics **125** (2020), 103535 (en).

[DKH22] Loi Do, Milan Korda, and Zdeněk Hurák, *Controlled Synchronization of Coupled Pendulums by Koopman Model Predictive Control*, December 2022, arXiv:2212.13783 [math].

[GHC+21] Genhua Guan, Hui Hu, Zhongtao Cheng, Lei Liu, Bo Wang, Huijin Fan, and Jufeng Li, *A Feedback Linearization Control of Morphing Missiles*, 2021 40th Chinese Control Conference (CCC), July 2021, ISSN: 1934-1768, pp. 7616–7621.

[GN81] J. S. Gibson and M. Navid, *Optimal control of flexible structures*, 1981 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, December 1981, pp. 700–701.

[Hoo65] Robert Hooke, *Micrographia : or, Some physiological descriptions of minute bodies made by magnifying glasses. With observations and inquiries thereupon.*, 1665.

[HTD06] S.Z. Mohd Hashim, M.O. Tokhi, and I.Z. Mat Darus, *Active Vibration Control of Flexible Structures Using Genetic Optimisation*, Journal of Low Frequency Noise, Vibration and Active Control **25** (2006), no. 3, 195–207 (en).

[JARCG19] Benjamin Jenett, Amira Abdel-Rahman, Kenneth Cheung, and Neil Gershenfeld, *Material–Robot System for Assembly of Discrete Cellular Structures*, IEEE Robotics and Automation Letters **4** (2019), no. 4, 4019–4026 (en).

[JC17] Ben Jenett and Kenneth Cheung, *BILL-E: Robotic Platform for Locomotion and Manipulation of Lightweight Space Structures*, 25th AIAA/AHS Adaptive Structures Conference, American Institute of Aeronautics and Astronautics, 2017, _eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2017-1876.

[JCGC16] Benjamin Jenett, Daniel Cellucci, Christine Gregg, and Kenneth Cheung, *Meso-Scale Digital Materials: Modular, Reconfigurable, Lattice-Based Structures*, Volume 2: Materials; Biomanufacturing; Properties, Applications and Systems; Sustainable Manufacturing (Blacksburg, Virginia, USA), American Society of Mechanical Engineers, June 2016, p. V002T01A018 (en).

[JCT+20] Benjamin Jennet, Christopher Cameron, Filippos Tourlomousis, Alfonso Parra Rubio, Megan Ochalek, and Neil Gershenfeld, *Discretely assembled mechanical metamaterials*, 2020.

[Jen20] Benjamin Eric Jenett, *Discrete Mechanical Metamaterials* (en).

[JKID90] S.C. Jacobsen, H. Ko, E.K. Iversen, and C.C. Davis, *Control strategies for tendon-driven manipulators*, IEEE Control Systems Magazine **10** (1990), no. 2, 23–28, Conference Name: IEEE Control Systems Magazine.

[KBBP16] J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, November 2016 (en).

[KM18a] Milan Korda and Igor Mezić, *Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control*, Automatica **93** (2018), 149–160 (en).

[KM18b] ———, *On Convergence of Extended Dynamic Mode Decomposition to the Koopman Operator*, Journal of Nonlinear Science **28** (2018), no. 2, 687–710 (en).

[KM20] ———, *Optimal Construction of Koopman Eigenfunctions for Prediction and Control*, IEEE Transactions on Automatic Control **65** (2020), no. 12, 5114–5129, Conference Name: IEEE Transactions on Automatic Control.

[KN32] B. O. Koopman and J. v. Neumann, *Dynamical Systems of Continuous Spectra*, Proceedings of the National Academy of Sciences **18** (1932), no. 3, 255–263, Publisher: Proceedings of the National Academy of Sciences.

[Koo31] B. O. Koopman, *Hamiltonian Systems and Transformation in Hilbert Space*, Proceedings of the National Academy of Sciences **17** (1931), no. 5, 315–318, Publisher: Proceedings of the National Academy of Sciences.

[LCCY94] Young-Tae Lee, Hyouk-Ryeol Choi, Wan-Kyun Chung, and Youngil Youm, *Stiffness control of a coupled tendon-driven robot hand*, IEEE Control Systems Magazine **14** (1994), no. 5, 10–19, Conference Name: IEEE Control Systems Magazine.

[LGG16] Will Langford, Amanda Ghassaei, and Neil Gershenfeld, *Automated Assembly of Electronic Digital Materials*, Volume 2: Materials; Biomanufacturing; Properties, Applications and Systems; Sustainable Manufacturing (Blacksburg, Virginia, USA), American Society of Mechanical Engineers, June 2016, p. V002T01A013 (en).

[LMWW20] Guan-Hua Li, Tian-Xue Ma, Yi-Ze Wang, and Yue-Sheng Wang, *Active control on topological immunity of elastic wave metamaterials*, Scientific Reports **10** (2020), no. 1, 9376 (en), Number: 1 Publisher: Nature Publishing Group.

[Mei13] C. Mei, *Wave Vibration Control of L-Shaped and Portal Planar Frames*, Journal of Vibration and Acoustics **135** (2013), no. 5.

[MJS+18] Mohammad J. Mirzaali, Shahram Janbaz, Matteo Strano, L. Vergani, and Amir Zadpoor, *Shape-matching soft mechanical metamaterials*, Scientific Reports **8** (2018).

[MK07] Satoshi Murata and Haruhsa Kurokawa, *Self-reconfigurable robots*, IEEE Robotics & Automation Magazine **14** (2007), no. 1, 71–78, Conference Name: IEEE Robotics & Automation Magazine.

[MKB+15] Indrek Must, Toomas Kaasik, Inna Baranova, Urmas Johanson, Andres Punning, and Alvo Aabloo, *A power-autonomous self-rolling wheel using ionic and capacitive actuators*, Electroactive Polymer Actuators and Devices (EAPAD) 2015, vol. 9430, SPIE, April 2015, pp. 172–180.

[NT19] Sebastian A. Nugroho and Ahmad F. Taha, *Sensor Placement Strategies for Some Classes of Nonlinear Dynamic Systems via Lyapunov Theory*, 2019 IEEE 58th Conference on Decision and Control (CDC), December 2019, ISSN: 2576-2370, pp. 4551–4556.

[OKH14] Ryuta Ozawa, Hiroaki Kobayashi, and Kazunori Hashirii, *Analysis, Classification, and Design of Tendon-Driven Mechanisms*, IEEE Transactions on Robotics **30** (2014), no. 2, 396–410, Conference Name: IEEE Transactions on Robotics.

[PBK16] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz, *Dynamic Mode Decomposition with Control*, SIAM Journal on Applied Dynamical Systems **15** (2016), no. 1, 142–161, Publisher: Society for Industrial and Applied Mathematics.

[PF90] L.Y. Pao and G.F. Franklin, *Time-optimal control of flexible structures*, 29th IEEE Conference on Decision and Control, December 1990, pp. 2580–2581 vol.5.

[PG07] Paulo José Paupitz Goncalves, *Dynamic analysis and active control of lattice structures*, phd, University of Southampton, October 2007.

[Pre18] André Preumont, *Vibration Control of Active Structures: An Introduction*, Springer, February 2018 (en), Google-Books-ID: oHNLDwAAQBAJ.

[PRFJ+23] Alfonso Parra Rubio, Dixia Fan, Benjamin Jenett, José del Águila Ferrandis, Filippos Tourlomousis, Amira Abdel-Rahman, David Preiss, Michael Triantafyllou, and Neil Gershenfeld, *Modular Morphing Lattices for Large-Scale Underwater Continuum Robotic Structures*, Soft Robotics (2023), soro.2022.0117 (en).

[PSSNCL15] J. F. Peza-Solís, G. Silva-Navarro, and N. R. Castro-Linares, *Trajectory Tracking Control in a Single Flexible-Link Robot using Finite Differences and Sliding Modes*, Journal of Applied Research and Technology. JART **13** (2015), no. 1, 70–78 (en), Publisher: Elsevier.

[SBG+20] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd, *OSQP: an operator splitting solver for quadratic programs*, Mathematical Programming Computation **12** (2020), no. 4, 637–672 (en).

[Sch10] Peter J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, Journal of Fluid Mechanics **656** (2010), 5–28 (en).

[SKM89] G. Singh, P. T. Kabamba, and N. H. McClamroch, *Planar, time-optimal, rest-to-rest slewing maneuvers of flexible spacecraft*, Journal of Guidance, Control, and Dynamics **12** (1989), no. 1, 71–81 (en).

[SMS06] Behnam Salemi, Mark Moll, and Wei-min Shen, *SUPERBOT: A Deployable, Multi-Functional, and Modular Self-Reconfigurable Robotic System*, 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 2006, ISSN: 2153-0866, pp. 3636–3641.

[SSP+19] Lucas A. Shaw, Frederick Sun, Carlos M. Portela, Rodolfo I. Barranco, Julia R. Greer, and Jonathan B. Hopkins, *Computationally efficient design of directionally compliant metamaterials*, Nature Communications **10** (2019), no. 1, 291 (en), Number: 1 Publisher: Nature Publishing Group.

[SSS90] J. Suhardjo, B. F. Spencer, and M. K. Sain, *Feedback-feedforward control of structures under seismic excitation*, Structural Safety **8** (1990), no. 1, 69–89 (en).

[SYCX21] Yongping Sun, Ming Yang, Yangyang Chen, and Dianguo Xu, *Trajectory Tracking and Vibration Control of Flexible Beams in Multi-Axis System*, IEEE Access **9** (2021), 156717–156728, Conference Name: IEEE Access.

[TDZK22] Marek Tyburec, Marek Doškář, Jan Zeman, and Martin Kružík, *Modular-topology optimization of structures and mechanisms with free material design and clustering*, 2022.

[TKS15] Mustafa Tinkir, Mete Kalyoncu, and Yusuf Sahin, *Modelling and Controller Design for a Flexible Structure System against Disturbance Effects*, Low Frequency Noise, Vibration and Active Control **34** (2015), 525–548.

[TM08] Yuzuru Terada and Satoshi Murata, *Automatic Modular Assembly System and its Distributed Control*, 2008.

[TRL+14] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz, *On dynamic mode decomposition: Theory and applications*, Journal of Computational Dynamics **1** (2014), no. 2, 391–421 (en), Publisher: Journal of Computational Dynamics.

[VCG+16] Vishesh Vikas, Eliad Cohen, Rob Grassi, Canberk Sözer, and Barry Trimmer, *Design and Locomotion Control of a Soft Robot Using Friction Manipulation and Motor–Tendon Actuation*, IEEE Transactions on Robotics **32** (2016), no. 4, 949–959, Conference Name: IEEE Transactions on Robotics.

[vMGM+23] Teunis van Manen, Mahya Ganjian, Khashayar Modaresifar, Lidy E. Fratila-Apachitei, and Amir A. Zadpoor, *Automated Folding of Origami Lattices: From Nanopatterned Sheets to Stiff Meta-Biomaterials*, Small **19** (2023), no. 3, 2203603 (en), _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/smll.202203603.

[WPN14] Justin Werfel, Kirstin Petersen, and Radhika Nagpal, *Designing Collective Behavior in a Termite-Inspired Robot Construction Team*, 2014.

[YDR00] M. Yim, D.G. Duff, and K.D. Roufas, *PolyBot: a modular reconfigurable robot*, Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), vol. 1, April 2000, ISSN: 1050-4729, pp. 514–520 vol.1.

Appendix A

KMPC Parameters

In this appendix, we provide formulas for computing matrices $\overline{\overline{Q}}$, $\overline{\overline{T}}$, and $\overline{\overline{R}}$, which we left out in the main text to maintain brevity.

$$\begin{aligned}
\overline{\overline{Q}} &= \begin{bmatrix} \tilde{C}^\top Q \tilde{C} & & & \\ & \ddots & & \\ & & \tilde{C}^\top Q \tilde{C} & \\ & & & \tilde{C}^\top S \tilde{C} \end{bmatrix} \in \mathbb{R}^{nN \times nN}, \\
\overline{\overline{T}} &= \begin{bmatrix} Q\tilde{C} & & & \\ & \ddots & & \\ & & Q\tilde{C} & \\ & & & S\tilde{C} \end{bmatrix} \in \mathbb{R}^{pN \times nN}, \\
\overline{\overline{R}} &= \begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix} \in \mathbb{R}^{mN \times mN}.
\end{aligned} \tag{A.1}$$

In the given context, N represents the length of the prediction horizon, while m and p denote the number of inputs and outputs of the augmented system, respectively. Additionally, n refers to the dimension of the augmented state vector $\tilde{z}$. Furthermore, we provide specific values for the bounds on the input increments $\Delta u[t]$ and the inputs $u[t]$ themselves.

$$\Delta \mathbf{u}_{\min} = \begin{bmatrix} -0.1 \\ \vdots \\ -0.1 \end{bmatrix} = -\Delta \mathbf{u}_{\max} \in \mathbb{R}^N, \qquad \mathbf{u}_{\min} = \begin{bmatrix} -0.5 \\ \vdots \\ -0.5 \end{bmatrix} = -\mathbf{u}_{\max} \in \mathbb{R}^N.$$

Appendix B

Contents of the Attachement

Together with this thesis, we submit the following files containing source code and other files and material important to the presented work. The contents are listed below in Table B.1.

Folder	Contents
`MATLAB` scripts	`MATLAB` files supporting the design and validation of control algorithms
`Simulink` schemes	`Simulink` models with implemented control for the Voxel Tower
LaTeX	LaTeXsource files generating this text

Table B.1: Contents of the attached folders.