Diss. ETH No. 23383

Non-Smooth Granular Rigid Body Dynamics with Applications to Chute Flows

A thesis submitted to attain the degree of
Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by

Gabriel Elias Nützi
MSc ETH ME, ETH Zurich
born on 31.10.1986
citizen of Wolfwil SO
Switzerland

accepted on the recommendation of

Prof. Dr.-Ing. Dr.-Ing. habil. Christoph Glocker, examiner
Prof. Dr.-Ing. Jürg Dual, co-examiner
Prof. Dr. ir. habil. Remco I. Leine, co-examiner

2016

```
template<typename My>
void to(My && family);
```

Acknowledgement

The work presented in this thesis has been carried out at the Institute for Mechanical Systems at the ETH Zurich. During these turbulent 4 years, I have been accompanied by many people whom I would like to devote my utmost thanks.

At first place, I would like to thank my supervisor Prof. Dr.-Ing. Dr.-Ing. habil. Ch. Glocker for supporting my research during these four years. He has given me the unconditional opportunity to extend my curiosity and my interest in mechanics as well as to delve deep into the beauty and madness of computer science. I was very much impressed right from the start of my studies about his endeavor for effective teaching and his strong believe in an environment where prosperous research and well-structured and concise education shake hand. As a mentor he taught me that nothing is too simple to be treated as obvious and his minimal non-trivial mechanical examples became the archetypical examples in this matter. I also devote my gratitude to his meticulous style of technical drawings and notation – the literature in engineering science will be so much more understandable if more attention to this matter is given. As an assistant, I was very much pleased to contribute to the teaching of mechanics under his lead.

Further thanks go to Prof. Dr.-Ing. Jürg Dual for his friendly welcome in his group and for his straightforward agreement to be my first co-examinator.

Furthermore, I very much like to thank Prof. Dr. ir. habil. Remco Leine for serving as my second co-examiner even at hardship. He has helped me closing some gaps in understanding essential concepts in non-smooth mechanics and has confronted me with the right questions at the right time and has always lent an ear for technical and non-technical issues.

I also want to thank Adrian Schweizer for sharing my office and for the fruitful technical and non-technical discussions and his patient willingness to teach me crazy stuff at the white board. I am also very thankful to Simon Eugster who helped me a lot in grasping some difficult concepts of modern continuum dynamics and for his continuous support throughout these four years. In this sense, I am also very thankful for Michael Baumann. His sharp-witted mathematical approach and his patience for my sometimes nonsense palaver was very much supportive. Playing chess together was much of a fun I already miss. Heartfelt thanks also go to the following people which supported and tolerated me in various ways: Michael Blösch, Raoul Hopf, George Rempfler, Thomas Heimsch.

I am also thankful to Michael Möller for his profound advice and knowledge in computer science and to Ondrej Papes for the discussion about variational integrators. Further thanks go to Thierry Baasch for his efforts in GPU computing during his master thesis. Many thanks also to all the members of the group of Prof. Jürg Dual who made my work life and launch times very enjoyable.

Another special thank goes to Claudia Bieler, Tom Feistel, Silvio Bamert and Perry Bartelt without whom I would not have been able to carry out the chute flow experiments. I am also thankful to Xylar Asay-Davis who helped me in using his fancy correlation algorithm for my research. Many thanks also go to all the people of the open-source community at `stackoverflow.com` which answered countless questions during my research and to Urban Borštnik and the Euler/Brutus cluster support team at the ETH Zurich.

I owe my deepest gratitude to Nadia Zollinger. I could not be more intimately grateful for her unconditional, selfless support, warmth and love. She always stood to my side even in unbearable times and has always been believing in me. Countless times she has been a patient listener not only to my many technical programming problems during this research. She will always stay my desirable reference when it comes to "treat others as you would wish to be treated".

I also thank Samuel Kilcher, Martin Knobel and Olaf Metzger for unforgettable climbing memories in Switzerland and abroad.

I thank my family and my friends for the always moral support and continuous encouragement. Without them, this work would not have been possible.

Zurich, February 2016 *Gabriel Elias Nützi*

Abstract

The work presented in this monograph is a contribution to the field of granular rigid body dynamics and its related subfields in computer science. The main contribution of this work is an open-source granular rigid body simulation framework (`GRSF`) which incorporates modern formulations and algorithms within the framework of non-smooth rigid body dynamics to efficiently simulate granular materials. This work discusses granular simulations in the context of their mechanical theory, their computer science related challenges and applications to real experiments.

In this work, a granular material is treated on the microscopic space scale where the particle interactions are treated as individual events over time. This modeling paradigm is of special interest because it gives detailed insight into the rheology and impacting behavior of granular materials in contrary to macroscopic models from fluid or continuum dynamics. The increasing trend in parallel computing at the present point in time evermore offers the performance needed to compute granular simulations also for large-scale models. The granular material model adopted in this work consists of a large-scale rigid body assembly in the physical space. The time evolution of a body in space is driven by two fundamental axioms in mechanics: the principle of virtual work and the variational law of interaction. These two axioms are exploited to properly derive the equations of motion of a rigid body starting from a scalable body which is parametrized by a quaternion.

Dissipation phenomena such as friction and impacts and the impenetrability condition between bodies in a granular material are conveniently modeled by set-valued contact laws. Concepts from convex analysis and convex optimization, such as normal cones and projections to convex sets, are directly at hand to describe and solve common set-valued contact laws, such as the unilateral contact with Coulomb friction. The contact laws are complemented with a Newton-type impact law to allow for discontinuities in the velocities of the bodies. The numerical time integration is performed with Moreau's explicit time-stepping scheme which discretizes the continuous and discontinuous motion of the bodies over a time interval and provides a good trade-off between accuracy and efficiency for large-scale multi-body simulations.

The aforementioned modern mechanical modeling aspects are implemented in the `GRS` framework which is able to efficiently and accurately simulate hundred thousands up to millions of rigid bodies. The parallel implementation, which mainly targets high-performance distributed systems, makes use of the mass-splitting method which subdivides the contact problem at each time step of the integration scheme. The contact problem, consisting of millions of contacts, is solved using iterative projection algorithms which are closely related to the methods from convex optimization. To spatially distribute the workload of the simulation, namely the time-stepping of the rigid bodies, to several processes, domain decomposition methods such as the grid or kd-tree decomposition are

implemented in the GRS framework by the help of the open-source library ApproxMVBB which was developed alongside the GRS framework. A simple load balancing strategy is employed to leverage the parallel computing power for simulations where bodies rapidly distribute over time. The Message Passing Interface (MPI) is used to provide the communication interface on distributed systems and to communicate and synchronize the numerical computations during the parallel simulation.

To demonstrate the functional ability of the GRS framework and to validate the numerical implementation, a chute flow experiment is performed at the institute of Snow and Avalanche Research (SLF) in Davos, Switzerland. The chute flow experiment consists of approximately 1 million glass beads which are released from rest in a channel above an inclined slope. A simulation study is performed where the friction coefficient between the glass beads is varied. By the help of an advection-corrected image correlation algorithm, the velocity field of the chute experiment is reconstructed and compared to the simulation. Visualizations ranging from two-dimensional velocity field plots to three-dimensional renderings provide meaningful insight into the results of the simulation and experiments. The comparison between the simulations and the experiment confirm the validity and usefulness of the numerical model implemented in the GRS framework.

Zusammenfassung

Die in der vorliegenden Abhandlung beschriebene Arbeit liefert einen Beitrag an die Forschung im Bereich der granularen Starrkörperdynamik und deren verwandten Unterbereichen der Computerwissenschaft. Der Hauptschwerpunkt der Arbeit liegt in der Umsetzung von modernen mechanischen Formulierungen im Bereich der nichtglatten Dynamik an Hand einer numerischen Softwareumgebung namens Granular Rigid Body Simulation Framework (`GRSF`), welche es erlaubt granulare Medien zu simulieren und zu untersuchen. Die Herausforderungen lagen dabei in der Anwendung der mechanischen Theorie im Bereich der computergestützten Wissenschaften und deren Übertragung auf reale Experimente.

Granulare Materialien werden in dieser Arbeit auf einer mikroskopischen Raumskala betrachtet, in welcher jede Partikelinteraktion im Verlaufe der Zeit als separates Ereignis behandelt wird. Diese mikroskopische Modellierungsart ist von speziellem Interesse, da sie exakten Einblick in das Fliess- und Stossverhalten von granularen Medien gibt, im Gegensatz zu makroskopischen Modellen, zum Beispiel aus der Fluiddynamik oder Kontinuumsmechanik. Die Einführung des Hochleistungsrechnens in das Forschungsgebiet der granularen Medien in der Mehrkörperdynamik eröffnet neue Möglichkeiten und Herangehensweisen. Stiess die Rechenleistung von Simulationen auf einem Rechenkern bei ungefähr 10'000 Starrkörpern an ihre Grenzen, kann heute mit Hilfe eines Rechenclusters ein Vielfaches dieser Menge simuliert werden. Das verwendete granulare Modell in dieser Arbeit besteht aus einer grossen Ansammlung von Starrkörpern im physikalischen Raum. Die Bewegung eines Körpers im Raum ist im wesentlichen beschrieben durch zwei fundamentale Axiome in der Mechanik: dem Prinzip der virtuellen Arbeit und dem variationellen Wechselwirkungsprinzip. Ausschliesslich diese zwei Axiome werden verwendet, um ausgehend vom skalierbaren Körper, welcher über ein Quaternion parametrisiert wird, die Bewegungsgleichungen des Starrkörpers auf konsistente Art herzuleiten.

Dissipationsmechanismen, wie Reibung und Stösse zwischen den Partikeln sowie deren Undurchdringbarkeit, werden mit Hilfe von mengenwertigen Kraftgesetzen modelliert. Konzepte der konvexen Analysis und konvexen Optimierung, wie zum Beispiel der Normalkegel oder Projektionen auf konvexe Mengen, werden verwendet um Kontaktgesetze, wie den unilateralen Kontakt mit Coulombscher Reibung, exakt abzubilden. Die diskutierten Kontaktgesetze werden mit einem Newtonschen Stossgesetz ergänzt, welches Unstetigkeiten in den Geschwindigkeiten der Körper ermöglicht. Die numerische Zeitintegration wird an Hand des expliziten Zeitschrittverfahrens von Moreau ermöglicht, welches die stossfreie und stossbehaftete Bewegung eines Körpers über ein Zeitintervall approximiert und dadurch einen guten Kompromiss zwischen Exaktheit und Effizienz bildet.

Das `GRS` Framework, welches die erwähnten mathematischen Formulierungen implementiert, ermöglicht es, Hunderttausende bis einige Millionen von Körpern effizient und exakt

zu simulieren. Die parallele Implementierung, welche hauptsächlich auf verteilte Hochleistungsarchitekturen fokussiert ist, verwendet das Massenzerteilungsverfahren, um das Kontaktproblem in jedem Zeitschritt zu unterteilen und parallel zu lösen. Das Kontaktproblem, bestehend aus Millionen von gekoppelten Kontakten, wird anhand von iterativen Projektionsalgorithmen gelöst, welche aus Methoden der konvexen Optimierung abgeleitet sind. Um die Ansammlung von Körpern räumlich aufzuteilen, werden zwei Raumaufteilungsverfahren diskutiert: die uniforme Gitteraufteilung und die kd-Baumstruktur. Beide Verfahren werden von der zusätzlich entwickelten Bibliothek `ApproxMVBB` unterstützt, welche Algorithmen zur Berechnung einer Begrenzungsbox mit minimalem Volumen bereitstellt. Ein einfaches Lastverteilungsverfahren im `GRS` Framework sorgt für die sinnvolle Ausnutzung der Computerressourcen für Simulationen mit sich schnell ausbreitenden Körperansammlungen. Das `GRS` Framework verwendet das Message Passing Interface (MPI), um die Kommunikation und die Synchronisation der numerischen Resultate während der parallelen Simulation auf verteilten Rechnerarchitekturen zu gewährleisten.

Um die Funktionalität des `GRS` Framework zu testen und um die numerische Implementierung zu validieren, wurde ein Schüttversuch am Institut für Schnee- und Lawinenforschung (SLF) in Davos durchgeführt. Das Experiment besteht aus ungefähr einer Million Glaskugeln, welche aus der Ruheposition in einem Kanal auf eine darunterliegende schiefe Ebene fliessen. In einer Parametersimulationsstudie wird der Reibungskoeffizient zwischen den Kugeln variiert, um einen Vergleich zwischen dem Experiment und dem numerischen Modell herzustellen. Mit Hilfe eines advektions-korrigierenden Bildkorrelationsalgorithmus wird das Geschwindigkeitsfeld aus den Videodaten rekonstruiert und mit den Simulationen verglichen. Zweidimensionale Visualisierungen von Experiment und Simulation sowie dreidimensionale Renderings der Simulation geben zusätzlich einen qualitativ wichtigen Einblick in den Vergleich zwischen dem Glassgranulat und dem mechanischen Starrkörpermodell. Der Vergleich zwischen Experiment und Simulation bestätigt die Anwendbarkeit des entwickelten numerischen Simulationsmodells.

Contents

Chapter 1

Introduction

In the last few decades, the computing power of high-performance and desktop computers has grown massively. Graphics processing units, which offer a thousandfold of small computing cores compared to a multi-processor desktop computer, are nowadays the state-of-the-art in high-performance computing. This fast progress enables to simulate computationally more expensive physical models which are at the same time simpler, more accurate and have less model parameters. This includes the modeling of granular matter which is the focus of this monograph.

From a fundamental point of view, a granular material is an assembly of particles large enough that thermal agitation can be neglected. Hence, a granular material can be viewed as a zero temperature mechanical system and this definition of granular matter, although applicable in the context of this thesis, is only one of many and highly depends on the perspective and research field.

The research on granular dynamics over the last decades can be summarized by the following apt quote:

> *"Granular materials represent a major object of human activities: as measured in tons, the first material manipulated on earth is water; the second is granular matter … This may show up in very different forms: rice, corn, powders for construction … In our supposedly modern age, we are extraordinarily clumsy with granular systems …"*
>
> — P. G. de Gennes, [43]

Although granular materials are seemingly simple to describe, they exhibit a tremendous amount of complex behavior much of which has not yet been satisfactorily explained, in particular when studying granular matter in three-dimensions. Granular materials behave differently than solids, liquids, and gases which justifies the characterization of granular materials as a new form of matter. The everyday example of refilling a pepper mill (wooden French ones preferred) by using a funnel with a too small outlet aptly demonstrates this: the peppercorns will eventually get stuck and the clogging can only be avoided by constantly shaking the mill[1].

[1] So happened to the author after some rough workdays and if not to spill pepper everywhere in the kitchen, that may not be the suggested way of doing it.

The motivation of this thesis is twofold: First, simulating granular material models described by rigid bodies and Coulomb-frictional interactions within the field of non-smooth dynamics is extremely interesting and provides a way to study quantities which are difficult to obtain from experiments without affecting the dynamics. That includes, for example, velocity field distribution, volumetric mass density distribution as well as pressure and stress distributions. Second, the software implementation of such granular models is fascinating and challenging and the applied algorithms, concepts and procedures for these models reach far into the field of computer science.

This thesis aims at providing an impetus for the experimental, theoretical and numerical study of granular materials.

The following literature review does not aim at giving a complete review on granular materials and their mechanical treatment. As the study of granular matter spans multiple disciplines such as geophysics, continuum mechanics, fluid dynamics, thermodynamics, statistical mechanics, rigid body dynamics and computer science, the reader is referred to literature which summarizes conducted and still ongoing work in agreement with the three parts presented in this thesis: the theory of non-smooth rigid body dynamics, the software implementation and the application of granular rigid body dynamics.

1.1 Literature Review

Since the development of the *contact dynamics method* in the field of non-smooth dynamics, today known as Moreau's time-stepping scheme, by subsequent theoretic and algorithmic contributions of Moreau and Jean in the late eighties and nineties (Moreau [112, 113, 119, 114, 115], Jean & Pratt [79], Jean [80], Jean & Moreau [78], Jean et. al [77] and Jean [76, 81]) and the rapid increase in computer power, many small up to large software frameworks have emerged which offer the ability to simulate non-smooth dynamic multi-body systems with constraints, contacts and friction for applications in different fields such as mechanical engineering, biomechanics, robotics and computer graphics. We will review some of the software frameworks later.

The non-smooth *contact dynamics method* is also said to belong to the class of discrete element methods (DEM). The *discrete element method*, pioneered by Cundall [42] in the seventies, is a procedure to simulate the dynamics of many particles which are treated as small rigid bodies, mostly spheres. In a classical *discrete element method*, the particle interactions are modeled as compliant contacts which leads to viscoelastic motion. Since contacts are modeled with impressed forces which depend only on the displacements and velocities of the particles, their numerical computation is simple except for long-range forces such as gravity, electrostatic and magnetic forces. [41, 189, 72, 187, 190, 188, 98, 101, 104, 57, 59, 159]. Friction between contacting particles is modeled by nonlinear springs and dampers. Resolving the impenetrability condition in a granular material modeled with a *discrete element method* implies a small timescale for the integration of the discretized equations of motion due to numerical instabilities.

The *contact dynamics method*, in contrast to the classical *discrete element method*, models the interactions between contacting rigid bodies with set-valued force laws which can

be determined by additionally considering the non-smooth equations of motion which includes continuous and non-continuous dynamics. The advantage of set-valued force laws is that they allow to formulate the impenetrability condition of rigid bodies in an exact form without introducing artificial penalization parameters or damping, that is, by using unilateral contacts. A lot of constitutive force laws in a mechanical system can be exactly represented in this framework such as isotropic and non-isotropic Coulomb friction or simple bilateral constraints such as translational and rotational joints between bodies. The non-smoothness of the *contact dynamics method* refers to different aspects: first, the velocities are discontinuous due to impacts in the system and second, the set-valued forces can also evoke discontinuities, for example, the stick-slip transitions caused by Coulomb friction. Furthermore, the timescale of the time-stepping procedure used to integrate the discretized non-smooth equations of motion can be chosen typically larger than in the case of a classical *discrete element method*. This is mainly due to the discretized formulation on velocity level which is numerically more stable and which approximates the interactions between rigid bodies during a discrete time step with a single or multiple impulsive forces, called percussions.

Non-smooth dynamics emerged from mechanical problems involving unilateral contacts and its sound mathematical formulation employs powerful and well-studied concepts from convex analysis and measure theory. For a complete history on the development of non-smooth mechanics the reader is referred to [30, 3].

The *contact dynamics method* has been applied to study different aspects of granular materials, such as shear zones [116], indeterminacy of contact forces in granular arrays and packings [149, 104], stress transmission [153], packing and rearrangements [29], velocity fluctuations [152], instabilities in granular piles [177], the influence of different particle shapes [138], shear instabilities and force transmission for two-dimensional discs [182], dynamic loading of ballast modeled with two-dimensional polygons [54], force transmission and packing for pentagonal particles [15], shear zones for three-dimensional spheres [160] and contact and force networks [154, 175]. Recent research with regard to the *contact dynamics method* also includes aspects like cohesive granular materials [164, 44], frictional contact models of soft-particle systems [133], the comparison between the material point and contact dynamics method [88], flows of polyhedral grains down a rough inclined plane [16], non-convex particle rheology in [165] and sedimentation transport [203] or the comparison study of the discrete element method with the contact dynamics method using the example of a granular material in a rotating drum [172]. Most of these studies have been conducted for two-dimensional granular materials. Studies on large-scale three-dimensional granular setups is still an open research field. The three-dimensional case is important especially for studying the rheology of granular flows with regard to the research of ice avalanches [148]. The *contact dynamics method* became one of the fundamental concepts in non-smooth rigid multi-body dynamics. It has been successfully applied for frictional multi-body systems such as masonry [4], tensegrity structures [137], rockfall [170], legged robots [58], bobsleigh models [155, 11] and microrobots [126], just to name a few. The results obtained from applying a contact dynamics approach often prove to be in good agreement with experimental observations as for example reported in [151, 116, 90, 170, 155, 11]. The reader is also referred to [181, 30] for more references on

modeling unilateral contacts and friction and its application.

Lots of effort has also been devoted to granular materials in the field of fluid and continuum mechanics. We refer to [64] for a broad review on this topic and to [200] for an overview of contact problems in continuum mechanics.

From a software perspective, rigid body dynamics became an attractive field in computer science since the early nineties when the computer graphics community became interested in efficient, plausible physics-based animations for computer games [18]. Nowadays, rigid body simulations build an important part of many modern software tools in a wide range of application areas such as virtual prototyping [199], training simulators in sports and medicine [8, 155] and special effects in the film industry [45]. It was mainly the *contact dynamics method*, developed by the effort of Jean and Moreau, which has opened up a new era in computer graphics and is today the state-of-the-art method for physically based interactive rigid body animations in computer games. The method is also better known as *impulse-based dynamics method* and builds the central idea one finds today in interactive rigid body engines such as `Bullet` [39] and `ODE` [176]. At time of writing, the software library `Bullet` encompasses one of the most extensive and most powerful collision detection back-ends available. The strength of `Bullet` lies in its recent developments for parallel computation on graphics processing units (GPUs). The reader is referred to the recent review [22] for an overview of the history of interactive rigid body dynamics in the field of computer graphics over the last 20 years.

Rigid as well as deformable body dynamics are nowadays still large open research fields in computer science since they offer many non-trivial challenges such as how to implement fast collision detection algorithms between different geometries [52], how to efficiently parallelize the numerical algorithms related to rigid body dynamics on different architectures [142, 17, 183, 40, 158] or how to design the different software layers in a general-purpose rigid multi-body dynamics pipeline [73, 127]. Certain software implementations presented in this work where inspired by the `pe` rigid multi-body engine [74, 75] which has set the standards in simulating large-scale multi-body systems on high-performance distributed systems. In [75], a simulation containing 1.1 billion rigid bodies has been conducted on 9120 processors on the HLRB-II supercomputer in Munich [75] with the fast frictional dynamics algorithm [83]. The application of the fast frictional dynamics algorithm mainly targets fast animations for computer graphics and its formulation, despite containing innovative ideas, is very intransparent in comparison to the sound formulation of impacts and friction in non-smooth dynamics. Another software project aiming at simulating granular materials is the open-source engine `Chrono` [103] which has support for distributed systems as well as for graphics processing units (GPUs) [183]. Both frameworks `pe` and `Chrono`, at time of writing, are extending their functionalities towards the interaction between fluid and rigid body dynamics. Another rigid multi-body software framework with the same focus and targeting distributed systems is `Solfec` [85]. The frameworks `pe`, `Chrono` and `Solfec` use spatial domain decomposition for the parallel time-stepping, where `Solfec` relies on the data-management and load-balancing services of the library `Zoltan` [25]. The framework `Chrono` relies on the collision detection capabilities of the library `Bullet` whereas the other mentioned rigid multi-body frameworks implement own collision detection routines. The frameworks `pe`, `Chrono` and `Solfec` all use iterative pro-

cedures for solving the contact problem in each time step. The implementations in `pe` and `Solfec` encompass projected fixed-point iterations (cf. SOR and JOR Prox algorithms in section 8.3, [108, 181]) whereas `Chrono` also contains other related iterative procedures from convex optimization [102].

The software packages `Siconos` [2], `MBSim` [169], `LMGC90` [48], and `DynamY` [181] are other multi-body frameworks with focus on the accurate and consistent simulation of multi-body systems with set-valued force laws and impacts. They are/were primarily not focused on large-scale granular simulations, meaning a multiple of 10^5 bodies, and do not primarily target parallel architectures. The framework `Siconos` also supports switched electrical circuits which can also be modeled with set-valued forces [109, 110]. The software `LMGC90` also supports electrical and thermal couplings [156, 150] and fluid-particle interactions [193].

The parallelization work in [174] for the simulation of granular materials using the *contact dynamics method* is important as it is the dual analogue to the mass-splitting method used in this work and discussed in chapter 13.

1.2 Aim & Scope

This thesis aims at unifying three aspects of the field of granular dynamics, namely the modeling of a granular material within the sound framework of non-smooth rigid body dynamics, the complex software implementation for the parallel simulation of granular matter on high-performance distributed systems and the application of theory and software to compare granular experiments with simulations.

The thesis main focus is the theory on non-smooth dynamics in part I and its related software implementation in part II. The application field is briefly covered in part III by applying the theory and software implementation to the example of a granular chute flow experiment.

Depending on the interest in the space scale, that is, the macroscopic, mesoscopic or microscopic scale, a granular material can be modeled in different ways. At the macroscopic scale, the granular material is considered as continuous which can be best described by methods within the framework of continuum mechanics or fluid dynamics. At the mesoscopic and microscopic scale, the methods mainly rely on modeling the behavior of each particle and the behavior of all interactions of each particle in its neighborhood. One major difficulty for building a macroscopic continuous model is the selection of relevant phenomenological parameters to represent different complex behavior such as elasticity, visco-plasticity or fracture. Because microscopic models are generally simpler and encompass less model parameters compared to macroscopic, continuous models which entail lots of model parameters to be identified, this thesis focuses on modeling a granular material with the *contact dynamics method* at the particle and interaction scale. This is in agreement with the endeavor of obtaining a model of a granular material which is as simple as possible, but not simpler[1]. Hence, in this thesis, the microscopic granular material

[1] with regard to the quote "Everything should be made as simple as possible, but not simpler." attributed to Einstein (1977)

model is described within the sound framework of non-smooth dynamics by modeling the particles as rigid bodies and their interactions with set-valued contact and impact laws. The modeling task therefore reduces to the choice of geometry of the particles and the modeling of the contact and impact laws. The main set-valued contact law discussed in this work is the unilateral contact with Coulomb friction with a Newton-type impact law extension for modeling the impact dynamics. The thesis also aims at giving a proper, self-contained overview into the necessary concepts from non-smooth dynamics to model granular materials such as: rigid body kinematics, rigid body dynamics, quaternions, convex optimization, time discretization and the description and solution methods of a contact problem.

Since the drawback of a microscopic model is mainly its increased computational cost, the thesis focuses on the parallel simulation of granular materials on high-performance distributed systems by using the Message Passing Interface (MPI). The thesis aims at providing a sound software framework, named Granular Rigid Body Simulation Framework (`GRSF`), to simulate granular rigid body dynamics. The main intention of the `GRS` framework is to provide a quality-conscious software implementation for the research community which is consistent with the theory both from a notational and conceptual point of view. The thesis aims at giving a proper discussion on the used parallelization techniques such as the spatial domain decomposition, the mass-splitting method, the communication mechanisms for the parallel simulation and the load balancing strategies using the developed library `ApproxMVBB` [139]. Some brief discussion is also provided for the data extraction and visualization tasks by using the converter tools of the `GRS` framework.

From the application point of view, the thesis provides some small exemplary insight into conducting an experiment of a chute flow of glass beads down a smooth inclined plane. A brief discussion is given on the particle analysis necessary for the performed simulation studies and on the velocity field reconstruction from the obtained video footage of the experiment. The brief comparison between simulation and experimental results at the end of the thesis aims at giving some first insight into the velocity field of the chute flow when the friction coefficient is gradually increased.

1.3 Outline

As already mentioned, this work is separated into three parts in agreement with the three major research fields of granular dynamics: theory, software and application. Part I encompasses the state-of-the-art theory on granular non-smooth rigid body dynamics with some new insights into deriving the equations of motion of a rigid body from a scalable body by only using the principle of virtual work. Part II encompasses the discussion on some of the more important intricacies of the software implementation with regard to the developed framework `GRSF` [141] and the library `ApproxMVBB` [139]. Part III focuses on the evaluation of a particular chute flow experiment conducted at the Institute for Snow and Avalanche Research (SLF) in Davos, Switzerland, by applying the concepts discussed in parts I and II. The reader is kindly referred to the pages 8, 80 and 138 for a detailed outline of each part.

Part I

Granular Rigid Body Dynamics

"'Obvious' is the most dangerous word in mathematics."
— E. T. Bell, *Mathematics: Queen and Servant of Science*, 1951

This part discusses all mechanical aspects relevant for the context of this thesis. As this thesis aims at modeling granular materials as large-scale rigid body systems, chapter 2 introduces the concept of a rigid body and its motion in the Euclidean space. Chapter 3 then discusses the parametrization of rotations using quaternions. Chapter 4 introduces the kinematics of the *scalable* and *rigid* body and preludes the derivation of the equations of motion for both of them. Chapter 5 gives a short overview of the two most fundamental principles in mechanics, namely the *principle of virtual work* and the *variational law of interaction*. The equation of motion of the *scalable* body, parametrized by a non-unit quaternion, is then directly derived from the principle of virtual work. The equation of motion for the *rigid* body follows directly by constraining the scalabilty of the *scalable* body. Chapter 5 is closed with some concepts required later such as the derivation of the equation of motion in generalized coordinates where the internal virtual work is omitted. Chapter 6 familiarizes the reader with some basic notion from convex analysis and convex optimization to describe different contact laws in chapter 7. Contact laws are used to model the internal interaction of a granular material modeled with many rigid bodies. Section 7.2 discusses the basics of the inclusion problem. The inclusion problem is the key component of the numerical algorithm to solve the contact laws together with a discretization of the equation of motion explained in chapter 8 and the impact equation discussed in section 7.3. Chapter 8 concludes with a discussion on the contact problem and its numerical treatise with the iterative projection algorithms.

Chapter 2

Rigid Body Assumption

The following discussion of the concept of a rigid body is based on [51, 170]. As described in [51], a *three-dimensional continuous body* is viewed as a three-dimensional compact differentiable manifold $\mathcal{B}$ with boundary. Each material point of a body $\mathcal{B}$ is placed in the physical space $\mathbb{E}^3$ by an embedding of $\mathcal{B}$ into the space $\mathbb{E}^3$ at any instant of time t. The space $\mathbb{E}^3$ with an orgin O denotes the three-dimensional Euclidean inner product space. An embedding of a body $\mathcal{B}$ at a time instant t is called the *configuration* of body $\mathcal{B}$ at time t. The time evolution of the configuration of a body $\mathcal{B}$ describes the *motion* of body $\mathcal{B}$ in the space $\mathbb{E}^3$ over time. The set of all material points of body $\mathcal{B}$ embedded in $\mathbb{E}^3$ at time t is denoted as $B_t \subset \mathbb{E}^3$. For the initial configuration at time $t = 0$, the subscript is omitted and the set is denoted by B. In contrast to the approach in [51], where a coordinate chart of $\mathcal{B}$ is chosen independently of the initial configuration, we identify this coordinate chart directly with coordinates of $\mathbb{E}^3$. Hence, the motion $\boldsymbol{\xi}$ of the body $\mathcal{B}$ in the physical space $\mathbb{E}^3$ is given as

$$\begin{aligned} \boldsymbol{\xi}(\cdot, t) : B &\rightarrow B_t \subset \mathbb{E}^3 \\ \mathbf{x} &\mapsto \boldsymbol{\xi} = \boldsymbol{\xi}(\mathbf{x}, t) \ , \end{aligned} \tag{2.1}$$

where $\mathbf{x} = x^1\mathbf{e}_1^{\mathrm{I}} + x^2\mathbf{e}_2^{\mathrm{I}} + x^3\mathbf{e}_3^{\mathrm{I}} \in B \subset \mathbb{E}^3$ is the position vector of a material point $P \in B$ at the time instant $t = 0$. The coordinates ${}_{\mathrm{I}}\mathbf{x} = [x^1, x^2, x^3]^\top \in \mathbb{R}^3$ with respect to the inertial coordinate system I with orthonormal basis $\bar{\mathbf{e}}^{\mathrm{I}} := (\mathbf{e}_1^{\mathrm{I}}, \mathbf{e}_2^{\mathrm{I}}, \mathbf{e}_3^{\mathrm{I}})$ can be thought of as the coordinate chart of the body manifold $\mathcal{B}$ (cf.[51, chapter 4]). Figure 2.1 visualizes the above mentioned concepts of a body $B \subset \mathbb{E}^3$.

So far, the motion $\boldsymbol{\xi}$ of a body in $\mathbb{E}^3$ is assumed to be differentiable in both arguments and bijective in the first argument for each time point t, that is, $\mathbf{x} = \boldsymbol{\xi}^{-1}(\boldsymbol{\xi}(\mathbf{x}, t), t)$. The map $\boldsymbol{\xi}$ describes the motion of a deformable body $\mathcal{B}$ embedded in $\mathbb{E}^3$. Since only non-deformable bodies, called *rigid bodies*, are considered in the context of this work, the motion $\boldsymbol{\xi}$ is constrained such that it forms an isometry in the first argument. What that means is explained in the following.

The map $\boldsymbol{\xi}$ is isometric in the first argument if the distance between any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ is preserved in time. Measuring distances in the physical space $\mathbb{E}^3$ is given by its induced Euclidean norm $\|\mathbf{x}\|_2 = \sqrt{(\mathbf{x} \,|\, \mathbf{x})}$, where $(\mathbf{x} \,|\, \mathbf{y})$ is the standard Euclidean inner product of $\mathbb{E}^3$. This renders the inner product space $\mathbb{E}^3$, called *Hilbert space*, also a

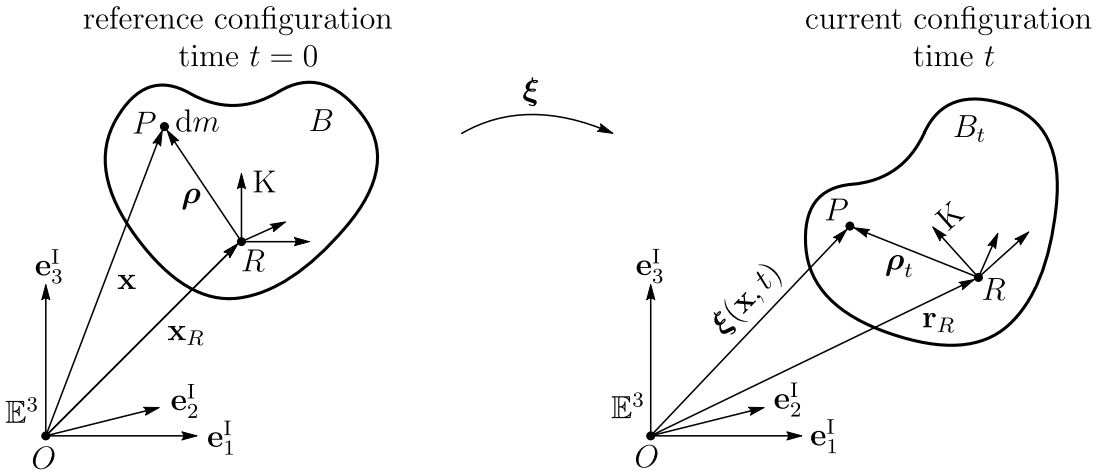


Figure 2.1: The motion $\boldsymbol{\xi}$ applied to a body $B \in \mathbb{E}^3$.

normed space. The requirement on $\boldsymbol{\xi}$ to be isometric for a given $t \in \mathbb{R}$ is

$$\|\boldsymbol{\xi}(\mathbf{x}_1, t) - \boldsymbol{\xi}(\mathbf{x}_2, t)\|_2 = \|\mathbf{x}_1 - \mathbf{x}_2\|_2 \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in B \subset \mathbb{E}^3 \ . \tag{2.2}$$

It is important to note that at an instant of time t, equation (2.2) imposes uncountably infinitely many bilateral constraints of the form $g(\mathbf{x}_1, \mathbf{x}_2) = 0$ for all possible pairs of points $\mathbf{x}_1, \mathbf{x}_2 \in B$. These constraints restrict the motion $\boldsymbol{\xi}$ of a continuous body to be *rigid* and are assumed to be *perfect*. A *perfect* constraint is defined by the principle of d'Alembert-Lagrange, which will be discussed briefly in chapter 5.

From the Mazur-Ulam theorem [135] follows that any surjective isometry between two normed vector spaces over the field $\mathbb{R}$ is an *affine* map. A map $a : V \to W$ between two vector spaces V and W is said to be *affine* if there exists a linear map $\mathcal{T} : V \to W$ and a translation $\mathbf{t} \in W$ such that $a(\mathbf{x}) = \mathcal{T}(\mathbf{x}) + \mathbf{t}$ holds. The translation is uniquely determined by setting $\mathbf{x} = \mathbf{0}$ which results in $a(\mathbf{0}) = \mathbf{t}$ since $\mathcal{T}(\mathbf{0}) = \mathbf{0}$. Therefore, the isometry (2.2) requires $\boldsymbol{\xi}$ to be affine in its first argument.

A decomposition of the motion $\boldsymbol{\xi}$ into a linear function and a translation at each time instant t, that is, $\boldsymbol{\xi}(\mathbf{x}, t) = \mathcal{T}(t)(\mathbf{x}) + \boldsymbol{\xi}(\mathbf{0}, t)$, is not directly possible as the domain B of the motion $\boldsymbol{\xi}$ does not contain $\mathbf{0}$ in general. However, defining the material point vector $\mathbf{x} = \boldsymbol{\rho} + \mathbf{x}_R$ where $\mathbf{x}_R \in B$ is an arbitrary reference point and $\boldsymbol{\xi}(\mathbf{x}, t) = \boldsymbol{\xi}(\boldsymbol{\rho} + \mathbf{x}_R, t) =: \boldsymbol{\xi}_{\mathrm{aff}}(\boldsymbol{\rho}, t)$, the decomposition yields

$$\boldsymbol{\xi}_{\mathrm{aff}}(\boldsymbol{\rho}, t) \coloneqq \mathcal{T}(t)(\boldsymbol{\rho}) + \mathbf{r}_R(t) \quad \underset{\mathcal{K}_{\mathrm{D}}^{-1}}{\overset{\mathcal{K}_{\mathrm{D}}}{\rightleftarrows}} \quad {}_{\mathrm{D}}\boldsymbol{\xi}_{\mathrm{aff}} = {}_{\mathrm{D}}\mathbf{T}(t)\ {}_{\mathrm{D}}\boldsymbol{\rho} + {}_{\mathrm{D}}\mathbf{r}_R(t) \ , \tag{2.3}$$

where $\mathbf{r}_R(t) = \boldsymbol{\xi}_{\mathrm{aff}}(\mathbf{x}_R, t)$ denotes the translation of the reference point R (see figure 2.1). The right-hand side in (2.3) is an example for the matrix-vector notation of representing the motion $\boldsymbol{\xi}_{\mathrm{aff}}$ in an arbitrary, not necessary orthonormal, coordinate system D where the coordinate map is denoted as $\mathcal{K}_{\mathrm{D}}$ and where ${}_{\mathrm{D}}\mathbf{T}(t) = \mathbf{mat}_{\mathrm{D}\leftarrow\mathrm{D}}(\mathcal{T}(t)) \in \mathbb{R}^{3\times 3}$ is the matrix representation of the map $\mathcal{T}(t)$ which maps coordinate tuples of vectors represented in basis D to coordinate tuples in basis D (see appendix D). A basis $\bar{\mathbf{e}}^{\mathrm{D}} = (\mathbf{e}_1^{\mathrm{D}}, \mathbf{e}_2^{\mathrm{D}}, \mathbf{e}_3^{\mathrm{D}})$ is said

to be orthonormal if $(\mathbf{e}_i^{\mathrm{D}} \,|\, \mathbf{e}_j^{\mathrm{D}}) = \delta_{ij}$, where δ_{ij} denotes the Kronecker delta. Furthermore, we abbreviate $\boldsymbol{\rho}_t \coloneqq \mathcal{T}(t)(\boldsymbol{\rho})$ (see figure 2.1).

If the notion of measuring angles given by the inner product of $\mathbb{E}^3$ is exploited together with its induced norm, it can be shown that the isometric map $\mathcal{T}(t)$ needs to preserve the inner product as well and $\mathcal{T}(t)$ is said to be orthogonal with respect to the inner product (cf. [170]), that is,

$$(\mathcal{T}(t)(\mathbf{x}) \,|\, \mathcal{T}(t)(\mathbf{y})) = (\mathbf{x} \,|\, \mathbf{y}) \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{E}^3. \tag{2.4}$$

Condition (2.4), evaluated in an orthonormal basis I, can be written in matrix-vector notation as

$$({}_{\mathrm{I}}\mathbf{T}_{\mathrm{I}}\mathbf{x})^{\top}({}_{\mathrm{I}}\mathbf{T}_{\mathrm{I}}\mathbf{y}) = {}_{\mathrm{I}}\mathbf{x}^{\top}{}_{\mathrm{I}}\mathbf{y} \quad \forall {}_{\mathrm{I}}\mathbf{x}, {}_{\mathrm{I}}\mathbf{y} \in \mathbb{R}^3 \tag{2.5}$$

$$\Rightarrow {}_{\mathrm{I}}\mathbf{T}^{\top}{}_{\mathrm{I}}\mathbf{T} = \mathbf{I}\,, \tag{2.6}$$

where $\mathbf{I} \in \mathbb{R}^{3\times 3}$ denotes the identity matrix. From (2.6) follows that ${}_{\mathrm{I}}\mathbf{R} \coloneqq {}_{\mathrm{I}}\mathbf{T}(t)$ has determinant $\det({}_{\mathrm{I}}\mathbf{R}) \in \{-1, +1\}$. A rotation ${}_{\mathrm{I}}\mathbf{R}$ belongs to the special orthogonal group $SO(3)$ if it obeys equation (2.6) and additionally has determinant $\det({}_{\mathrm{I}}\mathbf{R}) = +1$, which corresponds to the preservation of orientation in $\mathbb{E}^3$ and thus ${}_{\mathrm{I}}\mathbf{R}$ is said to be *proper*. Improper rotations with $\det({}_{\mathrm{I}}\mathbf{R}) = -1$ include additional reflections which are excluded in the context of rigid body dynamics.

By the constraint of isometry (2.2), the *affine* body motion $\boldsymbol{\xi}_{\mathrm{aff}}$ in (2.3) becomes a *rigid* body motion $\boldsymbol{\xi}_{\mathrm{rig}}$ defined as

$$\boldsymbol{\xi}_{\mathrm{rig}}(\boldsymbol{\rho}, t) \coloneqq \mathcal{R}(t)(\boldsymbol{\rho}) + \mathbf{r}_R(t) \quad \underset{\mathcal{K}_{\mathrm{D}}^{-1}}{\overset{\mathcal{K}_{\mathrm{D}}}{\rightleftarrows}} \quad {}_{\mathrm{D}}\boldsymbol{\xi}_{\mathrm{rig}} = {}_{\mathrm{D}}\mathbf{R}(t)\ {}_{\mathrm{D}}\boldsymbol{\rho} + {}_{\mathrm{D}}\mathbf{r}_R(t)\,, \tag{2.7}$$

where the time-dependent linear map of the rotation is denoted by $\mathcal{R}(t)$.

The rigid body motion in (2.7) parametrized by the rotation $\mathcal{R}(t)$ and translation $\mathbf{r}_R(t)$ defines a time-dependent path on the *configuration manifold* of a *rigid* body, which is the special Euclidean group $SE(3)$. Furthermore, the *rigid* body motion (2.7) builds the essential ingredient for the *variational law of interaction* in mechanics. The variational law of interaction presented in [51] is a requirement on the internal forces which asks their virtual work to vanish for all *rigid* virtual displacements induced by (2.7).

The correct treatise of rotations in mechanics plays an essential role in the simulation of rigid body systems. Rotations of $SO(3)$ can be parametrized in a number of ways. An elegant and commonly used parametrization is provided by quaternions which are described in more detail in the next section. Using unit quaternions to parametrize rotations has become the de facto standard in mechanics, robotics and computer graphics. A quaternion does not have a singularity in the mapping between its time derivative and the angular velocity of a rigid body. Three parameter formulations, for example the well-known Euler and Kardan angles, suffer from the aforementioned singularity. Nevertheless, enforcing the unit constraint of a quaternion used for a rigid body motion has the drawback of imposing an additional bilateral constraint on displacement level together with constraint forces in the equation of motion.

Chapter 3

Quaternions as Parametrization for Rotations

The theory on quaternions is well studied and possesses connections to many subfields in mathematics. Quaternions are embedded within the field of differential geometry by the Lie group theory, which is beyond the scope of this thesis. This section introduces quaternions in a slightly different way compared to [108, 170] and is inspired by [55]. The beauty of introducing quaternions as matrices lies in the ease of approaching their calculus and structure if the reader is already familiar with matrix-vector calculus. It is the authors opinion that this approach simplifies the understanding of the connection between rotations and quaternions considerably. The aim of this introduction is to combine the best parts of the excellent mathematical derivations and notation in the thesis of Schweizer [170] and Möller [108] as well as the theory in [55]. For more background on quaternions the reader is referred to [69, 9, 87, 198].

3.1 Properties

The space of quaternions $\mathbb{H}$ is conveniently introduced as a subspace of the space of complex matrices $\mathbb{C}^{2\times 2}$ and is given as

$$\mathbb{H} \coloneqq \{p_0\mathbf{I} + p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k}\ , \quad p_i \in \mathbb{R}\} \subset \mathbb{C}^{2\times 2}\ , \tag{3.1}$$

$$\text{with: } \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{i} = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix}, \quad \mathbf{j} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad \mathbf{k} = \begin{bmatrix} 0 & i \\ i & 0 \end{bmatrix}. \tag{3.2}$$

A quaternion $\mathbf{P} \in \mathbb{H}$ consists of a *scalar* part with coefficient $p_0 \in \mathbb{R}$ and basis vector $\mathbf{I}$, and a *pure* part consisting of three coefficients $p_1, p_2, p_3 \in \mathbb{R}$ with basis vectors $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$.

The quaternion space $\mathbb{H}$ is a 4-dimensional *linear space* (*vector space*), with the canonical basis $(\mathbf{I}, \mathbf{i}, \mathbf{j}, \mathbf{k})$ and the matrix addition $+ : \mathbb{H} \times \mathbb{H} \to \mathbb{H}$ and scalar multiplication $\cdot : \mathbb{R} \times \mathbb{H} \to \mathbb{H}$. Together with the standard matrix multiplication $\mathbb{H} \times \mathbb{H} \to \mathbb{H}$, the quaternion space $\mathbb{H}$ also forms a *non-commutative ring* in mathematical terminology.

As $\mathbb{H}$ is a linear space, we introduce the trivial canonical coordinate map $\mathcal{K}$ which maps

a vector $\mathbf{P} \in \mathbb{H}$ to a coordinate tuple $\mathbf{p} \in \mathbb{R}^4$ in basis $(\mathbf{I}, \mathbf{i}, \mathbf{j}, \mathbf{k})$ as

$$\boxed{\mathcal{K} : \mathbb{H} \to \mathbb{R}^4 \ , \quad \mathbf{p} = \mathcal{K}(\mathbf{P}) = \begin{bmatrix} p_0 \\ \mathbf{p}_r \end{bmatrix} ,} \tag{3.3}$$

where p_0 is the coordinate of the *scalar* part and $\mathbf{p}_r = [p_1, p_2, p_3]^\top \in \mathbb{R}^3$ is the coordinate tuple of the *pure* part. Since the coordinate map is bijective, $\mathbb{H}$ is isomorphic to $\mathbb{R}^4$. We abbreviate the inverse map as $\mathbf{P} = \mathcal{K}^{-1}(p_0, \mathbf{p}_r)$. The identity element with respect to the multiplication in $\mathbb{H}$ is the identity matrix $\mathbf{I}$ or represented in canonical coordinates as $[1, 0, 0, 0]^\top$. For convenience, we abbreviate in the following the coordinate tuple of a vector $\mathbf{P} \in \mathbb{H}$ with a bold lower case letter $\mathbf{p}$. It should be emphasized that the multiplication operator is merely defined on the *non-commutative ring* $\mathbb{H}$ and is the matrix-multiplication of two quaternions $\mathbf{P}$ and $\mathbf{Q}$. The notion of some induced operator $\mathbf{p} \cdot \mathbf{q}$ for the multiplication of their respective coordinate tuples $\mathbf{p}$ and $\mathbf{q}$ is not needed in the further course of this section.

The *conjugate* $\mathbf{P}^*$ of a quaternion $\mathbf{P} = \mathcal{K}^{-1}(\, p_0 \,,\, \mathbf{p}_r)$ is defined as

$$\mathbf{P}^* = \mathcal{K}^{-1}(\, p_0 \,,\, -\mathbf{p}_r) \quad \Leftrightarrow \quad \mathbf{p}^* = \begin{bmatrix} p_0 \\ -\mathbf{p}_r \end{bmatrix} , \tag{3.4}$$

where $\mathbf{P}^*$ is the conjugate transpose of the complex matrix $\mathbf{P}$ and $\mathbf{P}^* = \big(\overline{\mathbf{P}}\big)^\top = \overline{(\mathbf{P}^\top)}$ and $(\mathbf{PQ})^* = \mathbf{Q}^*\mathbf{P}^*$ holds, where $\overline{(\cdot)}$ denotes element-wise complex conjugation.

The *scalar* part $\mathrm{Re}(\mathbf{P})$ and the *pure* part $\mathrm{Im}(\mathbf{P})$ of a quaternion $\mathbf{P}$ is given by

$$\begin{aligned} \mathrm{Re}(\mathbf{P}) &= \frac{1}{2}(\mathbf{P} + \mathbf{P}^*) = p_0\mathbf{I} \ , & \mathrm{Im}(\mathbf{P}) &= \frac{1}{2}(\mathbf{P} - \mathbf{P}^*) = p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k} \ , \\ \mathrm{Re}(\mathbf{p}) &= \frac{1}{2}(\mathbf{p} + \mathbf{p}^*) = [p_0,\ \mathbf{0}^\top]^\top \ , & \mathrm{Im}(\mathbf{p}) &= \frac{1}{2}(\mathbf{p} - \mathbf{p}^*) = [0,\ \mathbf{p}_r{}^\top]^\top \ . \end{aligned} \tag{3.5}$$

The space of *quaternions* with only a *scalar* part and with only a *pure* part are denoted as $\mathbb{R}\mathbf{I}$ and $\mathbb{H}_p$, respectively. Note that

$$\mathbf{P} \in \mathbb{H}_p \quad \Leftrightarrow \quad \mathrm{Re}(\mathbf{P}) = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{P}^* = -\mathbf{P} \quad \Leftrightarrow \quad \mathbf{p}^* = -\mathbf{p} \ . \tag{3.6}$$

The matrix multiplication of two quaternions $\mathbf{PQ}$ yields

$$\mathbf{PQ} = \mathcal{K}^{-1}(\, p_0\, q_0 - \mathbf{p}_r{}^\top \mathbf{q}_r \,,\, p_0\, \mathbf{q}_r + q_0\, \mathbf{p}_r + \tilde{\mathbf{p}}_r \mathbf{q}_r) \ , \tag{3.7}$$

where $\tilde{\mathbf{p}}_r \in \mathbb{R}^{3\times3}$ is the real skew-symmetric matrix associated with the cross product, that is, $\tilde{\mathbf{x}}\mathbf{y} = \mathbf{x} \times \mathbf{y}$ for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$. Due to the non-commutative property of the matrix multiplication, the quaternion multiplication is not commutative in general. To make the vector space $\mathbb{H}$ a *Hilbert space*, the inner product $(\mathbf{P} \,|\, \mathbf{Q})$ of two quaternions $\mathbf{P}$ and $\mathbf{Q}$ is defined as

$$(\mathbf{P} \,|\, \mathbf{Q}) \coloneqq \frac{1}{2}\,\mathrm{Tr}(\mathbf{PQ}^*) = (\mathbf{p} \,|\, \mathbf{q}) = \mathbf{p}^\top \mathbf{q} = p_i q_i \ , \tag{3.8}$$

which is the standard Euclidean inner product of $\mathbb{E}^4$ which is a symmetric, positive definite bilinear form. Hence, the quaternion space $\mathbb{H}$ is said to be isomorphic to $\mathbb{E}^4$. In (3.8), Einstein summation convention is used. By (3.8), the induced norm is then given as $\|\mathbf{P}\| = \|\mathbf{p}\| = \sqrt{(\mathbf{p} \,|\, \mathbf{p})} = \sqrt{p_0^2 + p_1^2 + p_2^2 + p_3^2}$. The definition of the inner product (3.8) yields that the space of *scalar* quaternions $\mathbb{R}\mathbf{I}$ is orthogonal to the space of *pure* quaternions $\mathbb{H}_p$. Furthermore, the inverse $\mathbf{P}^{-1}$ of a quaternion $\mathbf{P}$ is obtained by evaluating

$$\mathbf{P}^{-1} = \begin{bmatrix} p_0 + ip_1 & p_2 + ip_3 \\ -p_2 + ip_3 & p_0 - ip_1 \end{bmatrix}^{-1} = \frac{1}{\det(\mathbf{P})} \begin{bmatrix} p_0 - ip_1 & -p_2 - ip_3 \\ p_2 - ip_3 & p_0 + ip_1 \end{bmatrix} = \frac{1}{\|\mathbf{P}\|^2} \mathbf{P}^* \,, \tag{3.9}$$

where one recognizes that the squared norm $\|\mathbf{P}\|^2 = \det(\mathbf{P})$ and that $\mathbf{P}\mathbf{P}^* = \mathbf{P}^*\mathbf{P} = \|\mathbf{P}\|^2 \mathbf{I}$. By (3.9), computing the inverse of a quaternion involves merely a conjugate transpose. From (3.9) it also follows that $\|\mathbf{P}\mathbf{Q}\| = \|\mathbf{P}\| \, \|\mathbf{Q}\|$ which can be shown by

$$\|\mathbf{P}\mathbf{Q}\|^2 \mathbf{I} = (\mathbf{P}\mathbf{Q})(\mathbf{P}\mathbf{Q})^* = \mathbf{P}\mathbf{Q}\mathbf{Q}^*\mathbf{P}^* = \mathbf{P} \, \|\mathbf{Q}\|^2 \mathbf{I}\mathbf{P}^* = \|\mathbf{P}\|^2 \, \|\mathbf{Q}\|^2 \mathbf{I} \,. \tag{3.10}$$

Due to the existence of the inverse for non-zero quaternions, the *non-commutative ring* $\mathbb{H}$ becomes a *non-commutative field*.

The matrix multiplication $\mathbf{Q} = \mathbf{P}\mathbf{X}$ of two quaternions $\mathbf{P}$ and $\mathbf{X}$ in the *field* $\mathbb{H}$ can also be expressed by a linear map $\varphi_L(\mathbf{P}) : \mathbb{H} \to \mathbb{H}$ in the *vector space* $\mathbb{H}$ such that $\mathbf{Q} = \varphi_L(\mathbf{P})(\mathbf{X})$. In canonical coordinates, this is $\mathbf{q} = \boldsymbol{\varphi}_L(\mathbf{P})\mathbf{x}$, where

$$\begin{aligned} \boldsymbol{\varphi}_L \;:\; \mathbb{H} \quad &\to \quad \mathbb{R}^{4\times 4}, \\ \mathbf{P} \quad &\mapsto \quad \boldsymbol{\varphi}_L(\mathbf{P}) = \begin{pmatrix} p_0 & -\mathbf{p}_r{}^\top \\ \mathbf{p}_r & p_0\mathbf{I} + \tilde{\mathbf{p}}_r \end{pmatrix}, \end{aligned} \tag{3.11}$$

follows from (3.7) and similarly for the reversed order of multiplication $\mathbf{Q} = \mathbf{X}\mathbf{P}$ by $\mathbf{q} = \boldsymbol{\varphi}_R(\mathbf{P})\mathbf{x}$ where

$$\begin{aligned} \boldsymbol{\varphi}_R \;:\; \mathbb{R}^4 \quad &\to \quad \mathbb{R}^{4\times 4}, \\ \mathbf{P} \quad &\mapsto \quad \boldsymbol{\varphi}_R(\mathbf{P}) = \begin{pmatrix} p_0 & -\mathbf{p}_r{}^\top \\ \mathbf{p}_r & p_0\mathbf{I} - \tilde{\mathbf{p}}_r \end{pmatrix}. \end{aligned} \tag{3.12}$$

Writing the matrices $\boldsymbol{\varphi}_L(\mathbf{P})$ and $\boldsymbol{\varphi}_R(\mathbf{P})$ in component form similar to (3.9), one sees that the columns are orthogonal to each other with respect to the defined inner product (3.8) for any quaternion $\mathbf{P}$. Additionally, the maps $\boldsymbol{\varphi}_L(\mathbf{P})$ and $\boldsymbol{\varphi}_R(\mathbf{P})$ fulfill

$$\boldsymbol{\varphi}_L(\mathbf{P}^*) = \boldsymbol{\varphi}_L(\mathbf{P})^\top \,, \qquad\qquad \boldsymbol{\varphi}_R(\mathbf{P}^*) = \boldsymbol{\varphi}_R(\mathbf{P})^\top \tag{3.13}$$

and both maps are *homomorphisms* since they preserve matrix multiplication and matrix addition in the *non-commutative field* $\mathbb{H}$, that is,

$$\boldsymbol{\varphi}_L(\mathbf{P} + \mathbf{Q}) = \boldsymbol{\varphi}_L(\mathbf{P}) + \boldsymbol{\varphi}_L(\mathbf{Q}) \,, \qquad\qquad \boldsymbol{\varphi}_L(\mathbf{P}\mathbf{Q}) = \boldsymbol{\varphi}_L(\mathbf{P})\boldsymbol{\varphi}_L(\mathbf{Q}) \,, \tag{3.14}$$

and analogously for $\boldsymbol{\varphi}_R$. Also note that the associative law of matrix multiplication $\mathbf{P}(\mathbf{X}\mathbf{Q}) = (\mathbf{P}\mathbf{X})\mathbf{Q}$ holds true and therefore

$$\boldsymbol{\varphi}_L(\mathbf{P})\boldsymbol{\varphi}_R(\mathbf{Q}) = \boldsymbol{\varphi}_R(\mathbf{Q})\boldsymbol{\varphi}_L(\mathbf{P}) \,. \tag{3.15}$$

To extract more properties from the maps $\boldsymbol{\varphi}_L(\mathbf{P})$ and $\boldsymbol{\varphi}_R(\mathbf{P})$, we evaluate $\boldsymbol{\varphi}_L(\mathbf{P})^\top\boldsymbol{\varphi}_L(\mathbf{P})$ which yields

$$\begin{aligned}\boldsymbol{\varphi}_L(\mathbf{P})^\top\boldsymbol{\varphi}_L(\mathbf{P}) &\overset{(3.13)}{=} \boldsymbol{\varphi}_L(\mathbf{P}^*)\boldsymbol{\varphi}_L(\mathbf{P}) \overset{(3.14)}{=} \boldsymbol{\varphi}_L(\mathbf{P}^*\mathbf{P}) \\ &= \boldsymbol{\varphi}_L(\|\mathbf{P}\|^2\,\mathbf{I}) = \|\mathbf{P}\|^2\,\mathbf{I}_4 \in \mathbb{R}^{4\times 4} \end{aligned} \tag{3.16}$$

and the same result holds true for $\boldsymbol{\varphi}_R(\mathbf{P})^\top\boldsymbol{\varphi}_R(\mathbf{P})$.
The determinant of $\boldsymbol{\varphi}_L(\mathbf{P})^\top\boldsymbol{\varphi}_L(\mathbf{P})$ follows from (3.16) as

$$\det(\boldsymbol{\varphi}_L(\mathbf{P})^\top\boldsymbol{\varphi}_L(\mathbf{P})) = \det(\boldsymbol{\varphi}_L(\mathbf{P}))^2 \overset{(3.16)}{=} \det(\|\mathbf{P}\|^2\,\mathbf{I}_4) = \|\mathbf{P}\|^8 \tag{3.17}$$

$$\Rightarrow\ \det(\boldsymbol{\varphi}_L(\mathbf{P})) = \|\mathbf{P}\|^4\ , \tag{3.18}$$

and analogue for $\boldsymbol{\varphi}_R$. The positiveness of $\det(\boldsymbol{\varphi}_L(\mathbf{P}))$ in (3.18) follows from evaluating it using (3.11). From the latter equality in (3.18) follows that if $\mathbf{P}$ is a *unit quaternion*, that is, $\|\mathbf{P}\| = 1$, then $\boldsymbol{\varphi}_L(\mathbf{P})$ and $\boldsymbol{\varphi}_R(\mathbf{P})$ are orthogonal matrices with determinant 1 and thus *proper* rotation matrices given by the special orthogonal group $SO(4)$ and preserve length, angles and orientation in $\mathbb{E}^4$ with respect to the inner product in (3.8). Note that an *improper rotation* has determinant -1 and is always expressed as a *proper rotation* in combination with a reflection in a plane and thus does not preserve orientation. The set of *unit quaternions* $\mathbb{H}_U = \{\mathbf{P} \in \mathbb{H} \mid \|\mathbf{P}\| = 1\}$ is exactly identical to the special unitary group of dimension 2 which is defined as $SU(2) \coloneqq \{\mathbf{P} \in \mathbb{C}^{2\times 2} \mid \mathbf{P}\mathbf{P}^* = \mathbf{P}^*\mathbf{P} = \mathbf{I},\ \det(\mathbf{P}) = 1\}$. The properties obtained so far are not yet linked to rotations in $SO(3)$ which preserve angles, length and orientation of vectors in $\mathbb{E}^3$. The following section shows the relation between rotations in $SO(3)$ and the linear maps φ_L and φ_R.

3.2 Rotation and Scaling in $\mathbb{E}^3$

We have seen in the last section that the maps $\boldsymbol{\varphi}_L(\mathbf{P})$ and $\boldsymbol{\varphi}_R(\mathbf{P})$ are proper rotations in $\mathbb{E}^4$ if and only if $\|\mathbf{P}\| = 1$. It is interesting to look at the composition of the two maps, namely

$$\begin{array}{rccl} \varphi(\mathbf{P},\mathbf{Q}) : & \mathbb{H} & \to & \mathbb{H} \\ & \mathbf{X} & \mapsto & \mathbf{P}\mathbf{X}\mathbf{Q} = \varphi_L(\mathbf{P}) \circ \varphi_R(\mathbf{Q})(\mathbf{X})\ , \end{array} \tag{3.19}$$

where $\mathbf{P}, \mathbf{Q} \neq \mathbf{0}$. The linear map $\varphi(\mathbf{P},\mathbf{Q})$ is also a rotation in $SO(4)$, if $\|\mathbf{P}\| = 1$ and $\|\mathbf{Q}\| = 1$ and the fact that the compositions of two rotations is still a rotation in $SO(4)$. In this case, the maps $\varphi_L(\mathbf{P})$ and $\varphi_R(\mathbf{Q})$ are called the *left-* and *right-isoclinic* rotations of $\varphi(\mathbf{P},\mathbf{Q})$, respectively. Actually, it is sufficient to require that $\|\mathbf{P}\|\,\|\mathbf{Q}\| = 1$ for the above to be a rotation. By using $\|\mathbf{P}\|\,\|\mathbf{Q}\| = 1$, the following computation

$$\det(\boldsymbol{\varphi}(\mathbf{P},\mathbf{Q})) = \det(\boldsymbol{\varphi}_L(\mathbf{P})\boldsymbol{\varphi}_R(\mathbf{Q})) = \|\mathbf{P}\|^4\,\|\mathbf{Q}\|^4 = (\|\mathbf{P}\|\,\|\mathbf{Q}\|)^4 = 1 \tag{3.20}$$

$$\|\mathbf{P}\mathbf{X}\mathbf{Q}\| = \|\mathbf{P}\|\,\|\mathbf{Q}\|\,\|\mathbf{X}\| = \|\mathbf{X}\|\,, \tag{3.21}$$

shows that the map $\varphi(\mathbf{P},\mathbf{Q})$ with $\|\mathbf{P}\|\,\|\mathbf{Q}\| = 1$ is an *isometry* which means that the length given by the induced norm is preserved, and, furthermore, its determinant is 1

which means that the orientation is preserved as well. The easiest way to fulfill the constraint $\|\mathbf{P}\|\,\|\mathbf{Q}\| = 1$ is to chose $\mathbf{Q} \coloneqq \mathbf{P}^{-1}$ which leads to the linear map

$$\begin{array}{rccll} \varphi(\mathbf{P}) \coloneqq \varphi(\mathbf{P}, \mathbf{P}^{-1}) & : & \mathbb{H} & \to & \mathbb{H} \qquad \text{with: } s \coloneqq \|\mathbf{P}\|^2 \\ & & \mathbf{X} & \mapsto & \mathbf{P}\mathbf{X}\mathbf{P}^{-1} = \frac{1}{s}\mathbf{P}\mathbf{X}\mathbf{P}^* = \\ & & & & \frac{1}{s}\varphi_L(\mathbf{P}) \circ \varphi_R(\mathbf{P}^*)(\mathbf{X}) \ , \end{array} \tag{3.22}$$

which is always an element of $SO(4)$ for any non-zero quaternion $\mathbf{P}$. The convenient property of $\varphi(\mathbf{P})$ is that if it is restricted to the space of *pure* quaternions $\mathbb{H}_p$, it is isomorph to a rotation in $SO(3)$. This important property, which finally leads to a quaternion representation of rotations in $\mathbb{E}^3$, is shown in the following.

The linear map $\varphi(\mathbf{P})$ constructed from a quaternion $\mathbf{P}$ maps every *scalar* quaternion in $\mathbb{R}\mathbf{I}$ onto itself and thus, it is the identity map for *scalar* quaternions, that is, $\mathbf{P}\mathbb{R}\mathbf{I}\mathbf{P}^{-1} = \mathbb{R}\mathbf{I}$. On the other hand, every quaternion can trivially be decomposed into a *scalar* and *pure* part as $\mathbf{X} = \mathrm{Re}(\mathbf{X}) + \mathrm{Im}(\mathbf{X})$. Since $\varphi(\mathbf{P})$ is linear, it directly follows that

$$\varphi(\mathbf{P})(\mathbf{X}) = \mathbf{P}\,\mathrm{Re}(\mathbf{X})\mathbf{P}^{-1} + \frac{1}{s}\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^* \tag{3.23}$$

$$= \mathrm{Re}(\mathbf{X}) + \frac{1}{s}\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^* \ . \tag{3.24}$$

From (3.24), one can see that only the *pure* part of $\mathbf{P}$ is mapped by the definition of φ in (3.22). It is left to check that $\frac{1}{s}\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^*$ is always *pure* for every $\mathbf{X}$, that is, $\mathrm{Re}(\frac{1}{s}\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^*) = \mathbf{0}\ \forall\mathbf{X}$. Indeed, an evaluation yields

$$\mathrm{Re}(\frac{1}{s}\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^*) = \frac{1}{2s}\big(\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^* + (\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^*)^*\big) \tag{3.25}$$

$$= \frac{1}{2s}\big(\mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^* - \mathbf{P}\,\mathrm{Im}(\mathbf{X})\mathbf{P}^*\big) = \mathbf{0} \ . \tag{3.26}$$

In summary, the space of pure quaternions $\mathbb{H}_p$ is mapped onto itself by the function φ and this directly implies a 3-dimensional rotation in the subspace $\mathbb{H}_p \subset \mathbb{H}$ by $\varphi(\mathbf{P})(\mathbb{H}_p)$.

Let us evaluate $\varphi(\mathbf{P}) \in SO(4)$ restricted to $\mathbb{H}_p$ in coordinates. A pure quaternion coordinate tuple $\mathbf{x} = [0, \mathbf{x}_r{}^\top]^\top$ is transformed to $\mathbf{y} = [0, \mathbf{y}_r{}^\top]^\top$ by (3.22) as

$$\mathbf{y} = \begin{bmatrix} 0 \\ \mathbf{y}_r \end{bmatrix} = \frac{1}{s}\boldsymbol{\varphi}_L(\mathbf{P})\boldsymbol{\varphi}_R(\mathbf{P}^*)\mathbf{x} = \frac{1}{s}\boldsymbol{\varphi}_L(\mathbf{P})\boldsymbol{\varphi}_R(\mathbf{P})^\top\mathbf{x} \ . \tag{3.27}$$

Multiplying (3.27) with $[\mathbf{0}\ ,\ \mathbf{I}] \in \mathbb{R}^{3\times 4}$ from the left and using $\mathbf{x} = [\mathbf{0}\ ,\ \mathbf{I}]^\top\mathbf{x}_r$ yields

$$\mathbf{y}_r = \frac{1}{s}\underbrace{[\mathbf{0}\ ,\ \mathbf{I}]\ \boldsymbol{\varphi}_L(\mathbf{P})}_{\begin{bmatrix} \mathbf{p}_r & p_0\mathbf{I} + \tilde{\mathbf{p}}_r \end{bmatrix}}\underbrace{\boldsymbol{\varphi}_R(\mathbf{P})^\top[\mathbf{0}\ ,\ \mathbf{I}]^\top}_{\begin{bmatrix} \mathbf{p}_r{}^\top \\ p_0\mathbf{I} + \tilde{\mathbf{p}}_r \end{bmatrix}}\mathbf{x}_r = \mathbf{R}\mathbf{x}_r \ . \tag{3.28}$$

The matrix $\mathbf{R} = \frac{1}{s}\hat{\mathbf{R}} \in \mathbb{R}^{3\times 3}$ in (3.28) with

$$\hat{\mathbf{R}} = \begin{bmatrix} \mathbf{p}_r & p_0\mathbf{I} + \tilde{\mathbf{p}}_r \end{bmatrix} \begin{bmatrix} \mathbf{p}_r & p_0\mathbf{I} - \tilde{\mathbf{p}}_r \end{bmatrix}^\top \tag{3.29}$$

and $\mathbf{P} \neq \mathbf{0}$ is always a proper rotation matrix in $\mathbb{E}^3$ whether $\mathbf{P}$ is a unit quaternion or not. Likewise, the matrix $\hat{\mathbf{R}} = s\mathbf{R}$ is always a rotation matrix with an additional scaling factor s. Hence, the coordinate tuple $\mathbf{y}_r \in \mathbb{R}^3$ in (3.28) is the rotated version of $\mathbf{x}_r$. By using the Lagrange identity

$$\mathbf{a}\,\mathbf{a}^\top = \mathbf{a}^\top\mathbf{a}\,\mathbf{I} + \tilde{\mathbf{a}}\tilde{\mathbf{a}} \quad \forall \mathbf{a} \in \mathbb{R}^3 \,, \tag{3.30}$$

a *proper* rotation matrix $\mathbf{R}$ is defined, given a non-zero quaternion $\mathbf{P}$, as

$$\begin{aligned} \mathbf{R} &\coloneqq \mathbf{I} + \frac{2}{s}(p_0\tilde{\mathbf{p}}_r + \tilde{\mathbf{p}}_r^2) \,, \qquad s \coloneqq \|\mathbf{P}\|^2 \,, \qquad \mathbf{P} = \mathcal{K}^{-1}(\,p_0\,,\,\mathbf{p}_r) \\ &= \frac{1}{s}\begin{bmatrix} p_0^2 + p_1^2 - p_2^2 - p_3^2 & 2(p_1p_2 - p_0p_3) & 2(p_0p_2 + p_1p_3) \\ 2(p_1p_2 + p_0p_3) & p_0^2 - p_1^2 + p_2^2 - p_3^2 & 2(p_2p_3 - p_0p_1) \\ 2(p_1p_3 - p_0p_2) & 2(p_0p_1 + p_2p_3) & p_0^2 - p_1^2 - p_2^2 + p_3^2 \end{bmatrix}, \end{aligned} \tag{3.31}$$

and if a unit quaternion is used, (3.31) simplifies to

$$\begin{aligned} \mathbf{R} &\coloneqq \mathbf{I} + 2(p_0\tilde{\mathbf{p}}_r + \tilde{\mathbf{p}}_r^2) \,, \qquad s \coloneqq \|\mathbf{P}\|^2 = 1 \,, \qquad \mathbf{P} = \mathcal{K}^{-1}(\,p_0\,,\,\mathbf{p}_r) \\ &= \begin{bmatrix} 1 - 2p_2^2 - 2p_3^2 & 2(p_1p_2 - p_0p_3) & 2(p_0p_2 + p_1p_3) \\ 2(p_1p_2 + p_0p_3) & 1 - 2p_1^2 - 2p_3^2 & 2(p_2p_3 - p_0p_1) \\ 2(p_1p_3 - p_0p_2) & 2(p_0p_1 + p_2p_3) & 1 - 2p_1^2 - 2p_2^2 \end{bmatrix}. \end{aligned} \tag{3.32}$$

Remark: From a numerical viewpoint, computing the matrix $\mathbf{R}$ by (3.32) with a non-unit quaternion is a bad choice, because the resulting matrix is no more a rotation matrix and transforming a vector $\mathbf{x}$ results in a distortion. This can be circumvented by normalizing the quaternion and using (3.32). However, it is computationally cheaper to use (3.31) with a non-unit quaternion due to the expensive square root operation used for the normalization.

Chapter 4

Body Kinematics

To motivate the equation of motion of a *rigid* body in the next section, a set of *generalized coordinates* $\mathbf{q} \in \mathbb{R}^{n_q}$ are derived from the *affine* motion in (2.3). In a further step, the generalized coordinates $\mathbf{q}$ are complemented with generalized velocities $\mathbf{u} \in \mathbb{R}^{n_u}$. The coordinate representation of $\mathcal{T}(t)$ and $\mathbf{r}_R(t)$ can be chosen as the generalized coordinates $\mathbf{q}$ which parametrize the configuration manifold of an *affine* body $B \subset \mathbb{E}^3$. Therefore, an *affine* body has 9 degrees of freedom for the affine map $\mathcal{T}(t)$ and 3 degrees of freedom for the translation $\mathbf{r}_R(t)$, and thus $n_q = 12$. For a *rigid* body motion where $\mathcal{T}(t) = \mathcal{R}(t) \in SO(3)$, the rotation $\mathcal{R}(t)$ has three degrees of freedom because 6 additional linearly independent constraints are imposed by (2.6). The rotation $\mathcal{R}(t)$ can be parametrized for example by three Euler or Kardan angles. If a quaternion is used to describe the rotation, the unit constraint of the quaternion needs to be taken into account. The parameters for such a parametrization of a coordinate representation of $\mathcal{R}(t)$ together with a coordinate representation of the translation $\mathbf{r}_R(t)$ form the generalized coordinates $\mathbf{q}$ of a *rigid* body. A *scalable* body possesses, compared to the *rigid* body, an additional degree of freedom for a uniform scaling in all directions and has therefore 7 degrees of freedom. This uniform scaling together with the rotation can be parametrized by the four parameters of a non-unit quaternion. If the scaling direction of the *scalable* body is restricted, that is, if the quaternion is enforced to be unit, a *rigid* body is obtained. In this sense, the equation of motion for a *scalable* body derived in the sequel of this chapter, build the foundation for the *rigid* body formulation using quaternions.

In the following, we will elaborate on the kinematics of a *scalable* body in detail to properly prepare the derivation of the equation of motion in chapter 5.

4.1 Scalable Body Kinematics

The motion of a *scalable* body is obtained by setting $\mathcal{T}(t) = s(t)\mathcal{R}(t)$, where $s(t)$ is an additional scaling parameter. The *scalable* body motion (omitting the time dependence

for the coordinate representation) is then given as

$$\boldsymbol{\xi}_{\text{scal}}(\boldsymbol{\rho}, t) = s(t)\mathcal{R}(t)(\boldsymbol{\rho}) + \mathbf{r}_R(t) \quad \underset{\mathcal{K}_{\text{I}}^{-1}}{\overset{\mathcal{K}_{\text{I}}}{\rightleftarrows}} \quad {}_{\text{I}}\boldsymbol{\xi}_{\text{scal}} = s\ {}_{\text{I}}\mathbf{R}\ {}_{\text{I}}\boldsymbol{\rho} + {}_{\text{I}}\mathbf{r}_R\ , \tag{4.1}$$

and can be parametrized using a time-dependent non-unit quaternion $\mathbf{p}(t) = \mathcal{K}(\mathbf{P}(t)) \in \mathbb{R}^4,\ \mathbf{P}(t) \in \mathbb{H}$ as presented in (3.29) with $s(t)\mathcal{R}(t) = \varphi_L(\mathbf{P}) \circ \varphi_R(\mathbf{P}^*)\big|_{\mathbb{H}_p \cong \mathbb{R}^3}$ which yields

$$\boldsymbol{\xi}_{\text{scal}}(\boldsymbol{\rho}, \mathbf{q}(t)) = s(t)\mathcal{R}(\mathbf{p}(t))(\boldsymbol{\rho}) + \mathbf{r}_R(t) \quad \underset{\mathcal{K}_{\text{I}}^{-1}}{\overset{\mathcal{K}_{\text{I}}}{\rightleftarrows}} \quad {}_{\text{I}}\boldsymbol{\xi}_{\text{scal}} = s\ {}_{\text{I}}\mathbf{R}(\mathbf{p})\ {}_{\text{I}}\boldsymbol{\rho} + {}_{\text{I}}\mathbf{r}_R\ , \tag{4.2}$$

with the definitions

$$\mathbf{q}(t) \coloneqq \begin{bmatrix} {}_{\text{I}}\mathbf{r}_R(t) \\ \mathbf{p}(t) \end{bmatrix} \in \mathbb{R}^7\ , \quad s(t) \coloneqq \|\mathbf{p}(t)\|^2\ . \tag{4.3}$$

The *rigid* body motion in (2.7) can be obtain by enforcing the constraint $g(\mathbf{q}(t)) = s(t) = \|\mathbf{p}(t)\|^2 = 1$.

The time-dependent coordinate system K is fixed to the body B and at time $t = 0$ is chosen to be identical to the inertial basis I, see figure 2.1. Therefore, the rotation $\mathcal{R}(t)$ rotates the basis vectors $\bar{\bar{\mathbf{e}}}^{\text{I}} \coloneqq (\mathbf{e}_1^{\text{I}}, \mathbf{e}_2^{\text{I}}, \mathbf{e}_3^{\text{I}})$ of the inertial coordinate system I to the basis vectors $\bar{\bar{\mathbf{e}}}^{\text{K}} \coloneqq (\mathbf{e}_1^{\text{K}}, \mathbf{e}_2^{\text{K}}, \mathbf{e}_3^{\text{K}})$ of the body-fixed coordinate system K which can be expressed by the notation $\mathcal{R}(t) \Leftrightarrow \mathcal{R}_{\text{KI}}(t)$, and in a coordinate system D as ${}_{\text{D}}\mathbf{R}_{\text{KI}}$. Note that the representation of the rotation matrix $\mathcal{R}_{\text{KI}}$ is the same for basis I and K, that is, ${}_{\text{I}}\mathbf{R}_{\text{KI}}(t) = {}_{\text{K}}\mathbf{R}_{\text{KI}}(t)$ (see appendix D.7). The choice of the coordinate system of the scalable body motion in (4.1) and (4.2) and for all further derivations, including the equations of motion, is irrelevant and could be neglected because of the coordinate independence of the underlying mechanical principles that describe the time evolution of a mechanical system. However, the notion of a representation in a basis is important since numerical computations can only be done by using coordinate tuples, but not by vectors of an abstract space. For that reason, we will only omit subscripts for coordinate system when explicitly stated.

To extract the generalized velocities $\mathbf{u}$ of a *scalable* body, we first consider the motion (4.1), where the rotation is not yet parametrized by a quaternion. As a guideline for the sequel of this chapter, the motion (4.2) is regarded as the central equation for the derivation of the equation of motion of the *scalable* body in the next section.

To obtain the absolute velocity of point P in figure 2.1, we differentiate equation (4.1) represented in the inertial coordinate system I with respect to time which yields

$${}_{\text{I}}\dot{\boldsymbol{\xi}}_{\text{scal}} = \dot{s}\ {}_{\text{I}}\mathbf{R}\ {}_{\text{I}}\boldsymbol{\rho} + s\ {}_{\text{I}}\dot{\mathbf{R}}\ {}_{\text{I}}\boldsymbol{\rho} + {}_{\text{I}}\dot{\mathbf{r}}_R\ . \tag{4.4}$$

It is natural to choose the absolute velocity ${}_{\text{I}}\dot{\mathbf{r}}_R$ of the reference point R as the translational part and the scaling velocity $\dot{s}$ as the scaling part of the generalized velocity $\mathbf{u}$. The absolute angular velocity of a body B which is linked to ${}_{\text{I}}\dot{\mathbf{R}}$ will be chosen later as the

rotational part of the generalized velocity $\mathbf{u}$. The next paragraph will introduce the angular velocity of a rotating coordinate system to finally define the angular velocity of the orthonormal body-fixed coordinate system K of the scalable body B with respect to the inertial basis I. The reader should note that the following discussion is rather technical and the end result is given in (4.9).

4.2 Angular Velocity

It is the authors opinion to rather introduce the absolute angular velocity of orthonormal coordinate transformations with the help of the *Euler differentiation* rule (see appendix D.8 and cf. [60]) instead of using time derivatives of rotation matrices. The *Euler differentiation* rule relates the absolute velocity of a vector $\dot{\mathbf{c}}$ expressed in a time-dependent coordinate system D to the time-differentiated coordinate representation $({}_{\mathrm{D}}\mathbf{c})^{\bullet}$ by

$$ {}_{\mathrm{D}}(\dot{\mathbf{c}}) = ({}_{\mathrm{D}}\mathbf{c})^{\bullet} + \mathbf{A}_{\mathrm{ID}}{}^{-1}\dot{\mathbf{A}}_{\mathrm{ID}}\ {}_{\mathrm{D}}\mathbf{c} \ , \tag{4.5} $$

where the inertial coordinate system is denoted by I. The *Euler differentiation* rule (4.5) holds for a general coordinate transformation $\mathbf{A}_{\mathrm{ID}} = \mathbf{mat}_{\mathrm{I}\leftarrow\mathrm{D}}(\mathbb{1}_{\mathbb{E}^3}) = (\mathbf{A}_{\mathrm{DI}})^{-1}$. For the general case, one can differentiate the identity map in both frames which yields

$$ \mathbf{A}_{\mathrm{ID}}{}^{-1}\mathbf{A}_{\mathrm{ID}} = {}_{\mathrm{D}}\mathbf{I} \quad \Rightarrow \quad \underbrace{(\mathbf{A}_{\mathrm{ID}}{}^{-1})^{\bullet}\mathbf{A}_{\mathrm{ID}}}_{=:\ {}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{DI}}} + \underbrace{\mathbf{A}_{\mathrm{ID}}{}^{-1}\dot{\mathbf{A}}_{\mathrm{ID}}}_{=:\ {}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{ID}}} = \mathbf{0} \quad \Rightarrow \quad {}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{DI}} + {}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{ID}} = \mathbf{0} \ , \tag{4.6} $$

$$ \mathbf{A}_{\mathrm{DI}}{}^{-1}\mathbf{A}_{\mathrm{DI}} = {}_{\mathrm{I}}\mathbf{I} \quad \Rightarrow \quad \underbrace{(\mathbf{A}_{\mathrm{DI}}{}^{-1})^{\bullet}\mathbf{A}_{\mathrm{DI}}}_{{}_{\mathrm{I}}\bar{\boldsymbol{\omega}}_{\mathrm{ID}}} + \underbrace{\mathbf{A}_{\mathrm{DI}}{}^{-1}\dot{\mathbf{A}}_{\mathrm{DI})}}_{{}_{\mathrm{I}}\bar{\boldsymbol{\omega}}_{\mathrm{DI}}} = \mathbf{0} \quad \Rightarrow \quad {}_{\mathrm{I}}\bar{\boldsymbol{\omega}}_{\mathrm{ID}} + {}_{\mathrm{I}}\bar{\boldsymbol{\omega}}_{\mathrm{DI}} = \mathbf{0} \ , \tag{4.7} $$

where the matrices ${}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{DI}}$ and ${}_{\mathrm{D}}\bar{\boldsymbol{\omega}}_{\mathrm{ID}}$ transform in the same way as the coordinate representation of a linear map. The transformations are visualized with arrows in (4.6) and (4.7). If only orthonormal coordinate systems are considered, the commutation rule for transposition and derivation holds, that is, $(\mathbf{A}_{\mathrm{ID}}{}^{\top})^{\bullet} = (\dot{\mathbf{A}}_{\mathrm{ID}})^{\top}$, and the term $\mathbf{A}_{\mathrm{ID}}{}^{-1}\dot{\mathbf{A}}_{\mathrm{ID}}$ becomes

$$ {}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}} \coloneqq \mathbf{A}_{\mathrm{ID}}{}^{\top}\dot{\mathbf{A}}_{\mathrm{ID}} \in so(3) \subset \mathbb{R}^{3\times 3} \tag{4.8} $$

which is the angular velocity of a rotating coordinate system D with respect to the coordinate system I represented in basis D. From (4.8) follows that the angular velocity ${}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}}$ forms a skew-symmetric matrix, that is, ${}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}}{}^{\top} = -{}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}}$. Since $-{}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}} = {}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{DI}}$ by (4.6), it follows that the transpose operation and the negation reverses the relative reference, meaning that ${}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{DI}} = {}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{ID}}{}^{\top}$. By the definition in (4.8), the *absolute* angular velocity ${}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}}$ of a *rigid* body B represented in its body-fixed coordinate system K is defines as ${}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}} \coloneqq {}_{\mathrm{K}}\tilde{\boldsymbol{\omega}}_{\mathrm{IK}} = \mathbf{A}_{\mathrm{IK}}{}^{\top}\dot{\mathbf{A}}_{\mathrm{IK}}$.

The Lie algebra $so(3)$, to which the angular velocity belongs, is a vector space and is the tangent space at the identity element of the space of rotions $SO(3)$. The skew-symmetric angular velocity ${}_{\mathrm{D}}\tilde{\boldsymbol{\omega}}_{\mathrm{IK}}$ can be identified in $\mathbb{E}^3$ with a vector ${}_{\mathrm{D}}\boldsymbol{\Omega}$ and the linear map it describes, by the cross product in $\mathbb{E}^3$ such that ${}_{\mathrm{D}}\boldsymbol{\Omega} \times \mathbf{x} = {}_{\mathrm{D}}\tilde{\boldsymbol{\Omega}}\mathbf{x}$. The map $\tilde{(\cdot)}$ from ${}_{\mathrm{D}}\boldsymbol{\Omega}$

to ${}_{\mathrm{D}}\tilde{\boldsymbol{\Omega}}$ is linear and ${}_{\mathrm{D}}\tilde{\boldsymbol{\Omega}}$ transforms in the same way as the representation of a linear map under a change of basis (see appendix D.7).

The angular velocity can be related to the rotation $\mathcal{R}_{\mathrm{KI}}$ by the relationship $\mathbf{A}_{\mathrm{IK}} = {}_{\mathrm{K}}\mathbf{R}_{\mathrm{KI}} = {}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}}$ (see appendix D.7) and from (4.8) then follows

$$ {}_{\mathrm{I}}\tilde{\boldsymbol{\Omega}} = \dot{\mathbf{A}}_{\mathrm{IK}}\mathbf{A}_{\mathrm{IK}}{}^{\top} = {}_{\mathrm{I}}\dot{\mathbf{R}}\ {}_{\mathrm{I}}\mathbf{R}^{\top} , \qquad {}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}} = \mathbf{A}_{\mathrm{IK}}{}^{\top}\dot{\mathbf{A}}_{\mathrm{IK}} = {}_{\mathrm{K}}\mathbf{R}^{\top}{}_{\mathrm{K}}\dot{\mathbf{R}} , \tag{4.9} $$

where the subscripts $(\cdot)_{\mathrm{KI}}$ of the rotation matrix are omitted. Substituting the time derivative of the rotation matrix in (4.4) with the relations in (4.9) yields the following two representations

$$ \begin{aligned} {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}} &= \dot{s}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} + s\ {}_{\mathrm{I}}\dot{\mathbf{R}}\ {}_{\mathrm{I}}\boldsymbol{\rho} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \dot{s}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} + s\ {}_{\mathrm{I}}\tilde{\boldsymbol{\Omega}}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \dot{s}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} - s\ ({}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho})\tilde{}\ {}_{\mathrm{I}}\boldsymbol{\Omega} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \begin{bmatrix} \mathbf{I} & {}_{\mathrm{I}}\bar{\boldsymbol{\rho}}(t) & -s\ {}_{\mathrm{I}}\tilde{\bar{\boldsymbol{\rho}}}(t) \end{bmatrix} \mathbf{u} \\ \text{with: } \mathbf{u} &\coloneqq \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ \dot{s} \\ {}_{\mathrm{I}}\boldsymbol{\Omega} \end{bmatrix} \in \mathbb{R}^6 , \end{aligned} \qquad \begin{aligned} {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}} &= \dot{s}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} + s\ {}_{\mathrm{I}}\dot{\mathbf{R}}\ {}_{\mathrm{I}}\boldsymbol{\rho} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \dot{s}\ {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} + s\ {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}}\ {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \dot{s}\ {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} - s\ {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\ {}_{\mathrm{K}}\boldsymbol{\Omega} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ &= \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -s\ {}_{\mathrm{K}}\mathbf{R}\ {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \mathbf{u} \\ \text{with: } \mathbf{u} &\coloneqq \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_R \\ \dot{s} \\ {}_{\mathrm{K}}\boldsymbol{\Omega} \end{bmatrix} \in \mathbb{R}^6 , \end{aligned} \tag{4.10} $$

where we used

$$ \boldsymbol{\rho}_t \coloneqq s(t)\mathcal{R}(t)(\boldsymbol{\rho}) = s(t)\bar{\boldsymbol{\rho}} , \qquad {}_{\mathrm{I}}\mathbf{R} = {}_{\mathrm{K}}\mathbf{R} , \tag{4.11} $$

$$ {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} = \mathbf{A}_{\mathrm{KI}}\ {}_{\mathrm{I}}\bar{\boldsymbol{\rho}} = \mathbf{A}_{\mathrm{KI}}\ {}_{\mathrm{I}}\mathbf{R}\ {}_{\mathrm{I}}\boldsymbol{\rho} = {}_{\mathrm{I}}\boldsymbol{\rho} , \qquad \tilde{\mathbf{x}}\mathbf{y} = -\tilde{\mathbf{y}}\mathbf{x} . \tag{4.12} $$

The representation of the angular velocity $\boldsymbol{\Omega}$ either in basis K or in basis I is chosen as the rotational part of the generalized velocity $\mathbf{u}$ in (4.10). The choice of $\mathbf{u}$ on the right-hand side in (4.10) with the angular velocity represented in the body-fixed basis K is preferred because it leads to a time-independent inertia tensor when deriving the equations of motion with the *principle of virtual work* in the next section.

For a mechanical system the relation between $\dot{\mathbf{q}}$ and $\mathbf{u}$ can always be written (cf. [60]) as

$$ \dot{\mathbf{q}} = \mathbf{F}(\mathbf{q}, t)\mathbf{u} + \boldsymbol{\beta}(\mathbf{q}, t) \tag{4.13} $$

and is called the *kinematic part* of the equation of motion.

In the following, we derive expression (4.13) for the motion of the *scalable* body in (4.2) and relate the time derivative of (4.2) to (4.10).

4.3 Minimal Velocities for the Scalable Body

As it will prove useful in the next chapter, we introduce minimal generalized velocities $\mathbf{u}_p$ for $\mathbf{q}$ in (4.2). The choice of $\mathbf{u}_p$ is obtained by differentiating the definition of $s(t)$ which

yields

$$s(t) \coloneqq \|\mathbf{p}\|^2 \;\Leftrightarrow\; s(t)\mathbf{I} = \|\mathbf{P}\|^2\,\mathbf{I} = \mathbf{P}^*\mathbf{P} \quad\Rightarrow\quad \dot{s}(t)\mathbf{I} = \dot{\mathbf{P}}^*\mathbf{P} + \mathbf{P}^*\dot{\mathbf{P}} \tag{4.14}$$

$$= (\mathbf{P}^*\dot{\mathbf{P}})^* + \mathbf{P}^*\dot{\mathbf{P}} \tag{4.15}$$

$$= \mathrm{Re}(\underbrace{2\mathbf{P}^*\dot{\mathbf{P}}}_{\mathbf{W}})\ . \tag{4.16}$$

As can be seen from (4.16), it makes sense to represent the term $2\mathbf{P}^*\dot{\mathbf{P}}$ by a time-dependent quaternion

$$\mathbf{W}(t) \coloneqq 2\mathbf{P}^*\dot{\mathbf{P}} \in \mathbb{H} \qquad \Leftrightarrow \qquad \mathbf{w}(t) = \mathcal{K}(\mathbf{W}) \coloneqq [\nu(t)\ ,\ \boldsymbol{\alpha}(t)^\top]^\top \in \mathbb{R}^4\ , \tag{4.17}$$

such that $\mathrm{Re}(\mathbf{w}) = \nu(t) = \dot{s}(t)$. From (4.17) and (3.9) follows

$$\dot{\mathbf{P}} = \frac{1}{2s}\mathbf{P}\mathbf{W} \in \mathbb{H} \qquad \Leftrightarrow \qquad \dot{\mathbf{p}} = \frac{1}{2s}\boldsymbol{\varphi}_L(\mathbf{P})\mathbf{w} \in \mathbb{R}^4\ . \tag{4.18}$$

The kinematic relation (4.13) for the *scalable* body between $\dot{\mathbf{q}}$ and $\mathbf{u}_p \coloneqq [{}_{\mathrm{I}}\dot{\mathbf{r}}_R(t)^\top, \mathbf{w}(t)^\top]^\top$ can be written as

$$\boxed{\begin{gathered}\dot{\mathbf{q}} = \mathbf{F}(\mathbf{q})\mathbf{u}_p,\\ \begin{bmatrix} {}_{\mathrm{I}}\mathbf{r}_R(t) \\ \mathbf{p}(t) \end{bmatrix}^{\cdot} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2s(t)}\boldsymbol{\varphi}_L(\mathbf{P}) \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_R(t) \\ \mathbf{w}(t) \end{bmatrix}\\ \text{with: } \mathbf{F}(\mathbf{q}) \in \mathbb{R}^{7\times 7}, \quad \mathbf{u}_p \coloneqq [{}_{\mathrm{I}}\dot{\mathbf{r}}_R(t)^\top, \mathbf{w}(t)^\top]^\top \in \mathbb{R}^7\ .\end{gathered}} \tag{4.19}$$

All relations are available now to evaluate the time derivative of (4.2). To do so, we rewrite (4.2) in the space $\mathbb{H}$ and differentiate with respect to time, that is,

$$\mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}}) = \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{P}\underbrace{\{0, {}_{\mathrm{I}}\boldsymbol{\rho}\}}_{\mathbf{X}\in\mathbb{H}_p}\mathbf{P}^*) + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.20}$$

$$= \dot{\mathbf{P}}\mathbf{X}\mathbf{P}^* + \mathbf{P}\mathbf{X}\dot{\mathbf{P}}^* + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.21}$$

$$\overset{(3.6)}{=} \dot{\mathbf{P}}\mathbf{X}\mathbf{P}^* - (\dot{\mathbf{P}}\mathbf{X}\mathbf{P}^*)^* + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.22}$$

$$\overset{(3.5)}{=} 2\,\mathrm{Im}(\dot{\mathbf{P}}\mathbf{X}\mathbf{P}^*) + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.23}$$

$$\overset{(4.18)}{=} 2\,\mathrm{Im}(\frac{1}{2s}\mathbf{P}\mathbf{W}\mathbf{X}\mathbf{P}^*) + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.24}$$

$$\overset{(3.22)}{=} \mathrm{Im}(\varphi(\mathbf{P})(\mathbf{W}\mathbf{X})) + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.25}$$

$$\overset{(3.24)}{=} \varphi(\mathbf{P})(\mathrm{Im}(\mathbf{W}\mathbf{X})) + \mathcal{K}^{-1}(0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R) \tag{4.26}$$

and converting back to coordinates yields

$$
\begin{aligned}
[0, {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}}{}^{\top}]^{\top} &= \boldsymbol{\varphi}(\mathbf{P})\,\mathrm{Im}(\boldsymbol{\varphi}_R(\mathbf{X})\mathbf{w}) + [0, {}_{\mathrm{I}}\dot{\mathbf{r}}_R{}^{\top}]^{\top} && (4.27)\\
\Rightarrow {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}} &= [\mathbf{0},\mathbf{I}]\,\boldsymbol{\varphi}(\mathbf{P})\ \mathbf{Im}\ \boldsymbol{\varphi}_R(\mathbf{X})\mathbf{w} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R(t) && (4.28)\\
&= \underbrace{[\mathbf{0},\mathbf{I}]\,\boldsymbol{\varphi}(\mathbf{P})\ [\mathbf{0},\mathbf{I}]^{\top}}_{{}_{\mathrm{I}}\mathbf{R}}\underbrace{[\mathbf{0},\mathbf{I}]\,\boldsymbol{\varphi}_R(\mathbf{X})}_{[{}_{\mathrm{I}}\boldsymbol{\rho},\ -{}_{\mathrm{I}}\tilde{\boldsymbol{\rho}}]}\mathbf{w} + {}_{\mathrm{I}}\dot{\mathbf{r}}_R(t) && (4.29)\\
&= \left[\ \mathbf{I}\ \ {}_{\mathrm{K}}\mathbf{R}_{\mathrm{K}}\bar{\boldsymbol{\rho}}\ \ -{}_{\mathrm{K}}\mathbf{R}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\ \right]\mathbf{u}_p\ . && (4.30)
\end{aligned}
$$

Comparing (4.30) with (4.10) gives the following important relation between the two minimal velocities $\mathbf{u}_p \coloneqq \left[{}_{\mathrm{I}}\dot{\mathbf{r}}_R{}^{\top},\ \nu,\ \boldsymbol{\alpha}^{\top}\right]^{\top}$ and $\mathbf{u} = \left[{}_{\mathrm{I}}\dot{\mathbf{r}}_R{}^{\top},\ \dot{s},\ {}_{\mathrm{K}}\boldsymbol{\Omega}\right]^{\top}$ as

$$
\nu(t) = \dot{s}(t), \quad \boldsymbol{\alpha} = s(t)\,{}_{\mathrm{K}}\boldsymbol{\Omega}, \quad \Leftrightarrow \quad \mathbf{u}_p = \mathrm{diag}\,(\mathbf{I}, 1, s\mathbf{I})\,\mathbf{u}\ . \tag{4.31}
$$

The kinematic relation for the *rigid* body can now be written in terms of $\mathbf{u}$ by setting the scaling degree of freedom in (4.19) to $s(t) = \|\mathbf{p}\|^2 = 1\ \forall t$ from which follows $\dot{s}(t) = \nu(t) = 0$. Using (4.31) then directly yields the kinematic equation (4.13) for the *rigid* body as

$$
\boxed{\begin{bmatrix} {}_{\mathrm{I}}\mathbf{r}_R(t) \\ \mathbf{p}(t) \end{bmatrix}^{\cdot} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\boldsymbol{\varphi}_L(\mathbf{P})\,[\mathbf{0},\mathbf{I}]^{\top} \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_R(t) \\ {}_{\mathrm{K}}\boldsymbol{\Omega}(t) \end{bmatrix}, \quad \|\mathbf{p}\|^2 = 1\ .} \tag{4.32}
$$

The most important property of the *kinematic* differential equation (4.19) is that the matrix $\mathbf{F}(\mathbf{q})$ is never singular as long as $\|\mathbf{P}\| = s(t) \neq 0$, because the inverse of $\boldsymbol{\varphi}_L(\mathbf{P})$ always exists since $\det(\boldsymbol{\varphi}_L(\mathbf{P})) = s(t) \neq 0$ by (3.18). The inverse $\mathbf{F}^{-1}(\mathbf{q})$ is obtained as

$$
\mathbf{F}^{-1}(\mathbf{q}) = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & 2\boldsymbol{\varphi}_L(\mathbf{P}^*) \end{bmatrix} \in \mathbb{R}^{7\times 7}\ . \tag{4.33}
$$

The property $\det(\mathbf{F}(\mathbf{q})) \neq 0$ is the reason why a quaternion parametrization of the rotation for the simulation of *rigid* bodies is very convenient. For other parametrizations, such as Euler or Kardan angles $\boldsymbol{\gamma} \in \mathbb{R}^3$, the map $\dot{\boldsymbol{\gamma}} = \mathbf{H}(\boldsymbol{\gamma})\,{}_{\mathrm{K}}\boldsymbol{\Omega}(t)$ will always become singular at certain angles $\boldsymbol{\gamma}$, regardless in which order the elementary rotations are multiplied. When numerically integrating equation (4.32), the constraint $\|\mathbf{p}\| = 1$ has to be enforced explicitly if the integration scheme does not preserve it implicitly. This can be achieved by an additional normalization or by including the additional bilateral constraint $\|\mathbf{p}\| = 1$ in the equation of motion. Interesting research on this issue has been conducted in [184, 108].

This side note concludes the preparations for the derivation of the equation of motion of the *scalable* body in the next section.

Chapter 5

Body Dynamics

A mechanical system $\mathcal{S}$ is a set of points which is embedded in the physical space $\mathbb{E}^3$. The mechanical system $\mathcal{S}$ described in this work is a continuous body $\mathcal{B}$ embedded in the physical space $\mathbb{E}^3$, denoted as $B \subset \mathbb{E}^3$. Each material point $\mathbf{x} \in B$ of the body is able to interact with other points in the set B and with its environment by *forces*. Defining the term *force* is one of the most fundamental yet difficult parts in the field of mechanics. From the effort of a lot of famous mathematicians, physicians and philosophers. over the last 4 centuries, one representation has emerged to be extremely useful. A *force* $\mathbf{F}$ can be seen as the dual quantity to a primal element $\delta\boldsymbol{\xi}$ from the space of virtual displacement fields. The virtual displacement field is induced by the kinematic parametrization of the mechanical system. The definition of a force is solidified by the duality pairing $\delta W = \langle \mathbf{F} \,|\, \delta\boldsymbol{\xi} \rangle \in \mathbb{R}$ (see appendix D.9), which is called *virtual work*. For the case of embedding the body in $\mathbb{E}^3$, a virtual displacement $\delta\boldsymbol{\xi}(\mathbf{x}) \in T_{\boldsymbol{\xi}}\mathbb{E}^3 \cong \mathbb{E}^3$ at a material point $\mathbf{x} \in B$ is an element of tangent space $T_{\boldsymbol{\xi}}\mathbb{E}^3$ located at the displaced point $\boldsymbol{\xi}(\mathbf{x})$. A force $\mathbf{F}$ is an element of the dual space, namely $\mathbf{F} \in T^*_{\boldsymbol{\xi}}\mathbb{E}^3 \cong \mathbb{E}^{3*}$. The following translated quote is helpful in understanding the nature of a *force*:

> *"Forces cannot be seen or visualized in the physical space. Forces are sensed point-wise by imposing virtual displacements on them to fully characterize them by size and direction."*
>
> — Ch. Glocker, *Lecture Notes on Non-Smooth Dynamics*, 2015

One can evaluate the duality pairing in a basis I as $\langle \mathbf{F} \,|\, \delta\boldsymbol{\xi}(\mathbf{x}) \rangle = {}_{\mathrm{I}}\mathbf{F}^{\top}{}_{\mathrm{I}}\delta\boldsymbol{\xi}(\mathbf{x})$, and since we are given the standard Euclidean inner product in $\mathbb{E}^3$, the space $\mathbb{E}^3$ is isomorphic to its dual $\mathbb{E}^{3*}$, see section D.9.2. Note that the duality pairing, by the definition of the dual basis, is independent of any coordinate system and left subscripts are neglected in the following. The dynamic equilibrium of a mechanical system S is given by the *principle of virtual work* (cf. [51]) as an axiom.

Axiom 5.1 *(Principle of Virtual Work)*:
At any instant of time t, the virtual work δW of a body $B \in \mathbb{E}^3$ vanishes for all virtual displacements $\delta\boldsymbol{\xi}$, that is,

$$\delta W(\delta\boldsymbol{\xi}) = \int_B \langle \mathrm{d}\mathbf{F} \,|\, \delta\boldsymbol{\xi}(\mathbf{x}) \rangle = \delta W^{\mathrm{dyn}}(\delta\boldsymbol{\xi}) + \delta W^{\mathrm{int}}(\delta\boldsymbol{\xi}) + \delta W^{\mathrm{ext}}(\delta\boldsymbol{\xi}) = 0 \quad \forall \delta\boldsymbol{\xi} \; . \tag{5.1}$$

The principle of virtual work is independent of the coordinate representation by definition and the variation $\delta\mathbf{c}$ of a vector-valued function $\mathbf{c}(t)$ is understood as

$$\delta\mathbf{c}(t) = \frac{\partial\hat{\mathbf{c}}}{\partial\varepsilon}(t,\varepsilon_0)\delta\epsilon \ , \quad \mathbf{c}(t) = \hat{\mathbf{c}}(t,\varepsilon_0) \ , \tag{5.2}$$

where $\hat{\mathbf{c}}(t,\varepsilon)$ denotes the variational family parametrized by ε.

The main difficulty of applying the above integration over the body B is to find the *constitutive* force laws, namely modeling the contributions $\delta W^{\text{dyn}}, \delta W^{\text{int}}$ and δW^{ext}. The term δW^{dyn} is the virtual work contribution of the inertia of the body. The terms δW^{int} and δW^{ext} are the contributions of *internal* interactions, that is, interactions among points of B, and *external* interactions between points of B and its environment.

The inertia contribution δW^{dyn} for a body B is modeled in correspondence to Newtons second law as

$$\delta W^{\text{dyn}}(\delta\boldsymbol{\xi}) \coloneqq \int_B \delta\boldsymbol{\xi}(\mathbf{x})^\top \mathrm{d}\mathbf{F}^{\text{dyn}}(\mathbf{x}) \ , \quad \text{(virtual work contribution of the inertia)} \tag{5.3}$$

$$\mathrm{d}\mathbf{F}^{\text{dyn}}(\mathbf{x}) \coloneqq -\ddot{\boldsymbol{\xi}}(\mathbf{x})\mathrm{d}m(\mathbf{x}) \ , \qquad \text{(constitutive force law)} \tag{5.4}$$

where $\mathrm{d}m(\mathbf{x})$ is the measure of the mass distribution of body B. The external contribution is modeled as

$$\delta W^{\text{ext}}(\delta\boldsymbol{\xi}) \coloneqq \int_B \delta\boldsymbol{\xi}(\mathbf{x})^\top \mathrm{d}\mathbf{F}^{\text{e}}(\mathbf{x}) \ , \qquad \text{(external virtual work contribution)} \tag{5.5}$$

where $\mathrm{d}\mathbf{F}^{\text{e}}(\mathbf{x})$ denotes the measure for the force distribution on body B which may contain Dirac-type contributions as well.

The *variational law of interaction* in [51] is the second axiom in mechanics and is a requirement for the *internal* virtual work contribution δW^{int}.

Axiom 5.2 *(Variational Law of Interaction)*:
At any instant of time t, the *internal* virtual work $\delta W^{\text{int}}_{B'}$ of any subsystem $B' \subseteq B$ vanishes for all *rigid* virtual displacements $\delta\boldsymbol{\xi}$, that is,

$$\delta W^{\text{int}}_{B'}(\delta\boldsymbol{\xi}) = 0 \quad \forall\ \delta\boldsymbol{\xi} \text{ rigidifying}, \ B' \subseteq B \ . \tag{5.6}$$

Rigidifying virtual displacements of some motion $\boldsymbol{\xi}$, denoted by $\delta\boldsymbol{\xi}^{\text{rfy}}$, can be constructed from the virtual *rigid* body displacements $\delta\boldsymbol{\xi}_{\text{rig}}$. Virtual *rigid* body displacements $\delta\boldsymbol{\xi}_{\text{rig}}$ are derived from varying the rigid body motion (2.7) which yields

$$\delta\boldsymbol{\xi}_{\text{rig}}(\boldsymbol{\rho},t) \coloneqq \delta\mathcal{R}(t)(\boldsymbol{\rho})\ \boldsymbol{\rho} + \delta\mathbf{r}_R(t) \quad \underset{\mathcal{K}_{\text{I}}^{-1}}{\overset{\mathcal{K}_{\text{I}}}{\rightleftarrows}} \quad {}_{\text{I}}\delta\boldsymbol{\xi}_{\text{rig}} = {}_{\text{I}}\delta\mathbf{r}_R + {}_{\text{I}}\delta\tilde{\boldsymbol{\phi}}\ {}_{\text{I}}\boldsymbol{\rho} \ , \tag{5.7}$$

$$\begin{aligned} {}_{\text{I}}\delta\tilde{\boldsymbol{\phi}} &\coloneqq \delta\mathbf{A}_{\text{IK}}\mathbf{A}_{\text{IK}}{}^\top = \delta({}_{\text{I}}\mathbf{R}_{\text{KI}})\ {}_{\text{I}}\mathbf{R}_{\text{KI}}{}^\top \ , \\ {}_{\text{K}}\delta\tilde{\boldsymbol{\phi}} &\coloneqq \mathbf{A}_{\text{IK}}{}^\top\delta\mathbf{A}_{\text{IK}} = {}_{\text{K}}\mathbf{R}_{\text{KI}}{}^\top\ \delta({}_{\text{K}}\mathbf{R}_{\text{KI}}) \ , \end{aligned} \tag{5.8}$$

for some given inertial basis I and a given body-fixed basis K. The vector $\delta\boldsymbol{\phi}$ in (5.7) represents virtual quasi-angles and is in direct correspondence with the angular velocity $\boldsymbol{\Omega}$,

that is, confer (4.9) and replace the time derivative with the ε-derivative given in (5.2). Omitting the coordinate system in (5.7) and replacing $\boldsymbol{\rho}$ with $\boldsymbol{\xi} - \mathbf{r}_R$ yields the *rigidifying* virtual displacment $\delta\boldsymbol{\xi}^{\mathrm{rfy}}$ of a motion $\boldsymbol{\xi}$ as

$$\delta\boldsymbol{\xi}^{\mathrm{rfy}} \coloneqq \delta\mathbf{r}_R + \delta\tilde{\boldsymbol{\phi}}\ (\boldsymbol{\xi} - \mathbf{r}_R)\ , \tag{5.9}$$

where the point R can be chosen arbitrarily. Equation (5.9) defines a rigid virtual displacement field over all material points of B at its current configuration B_t.

The law of interaction can be understood in the way that *internal* forces formulated by an *internal* virtual work contribution δW^{int} must not produce any contribution to the total virtual work δW among virtual displacements of material points which preserve the isometry (2.2). In other words, internal forces are required to produce no virtual work contribution when body B is not virtually deformed. To better understand this concept, we consider a continuous body B with a subsystem B' consisting only of two points $\boldsymbol{\rho}_1 \in B$ and $\boldsymbol{\rho}_2 \in B$ as shown in figure 5.1a. The internal virtual work takes the form

$$\delta W^{\mathrm{int}}_{B'}(\delta\boldsymbol{\xi}) \coloneqq \delta\boldsymbol{\xi}(\boldsymbol{\rho}_1, t)^{\top}\mathbf{F}_1 + \delta\boldsymbol{\xi}(\boldsymbol{\rho}_2, t)^{\top}\mathbf{F}_2\ , \tag{5.10}$$

where $\mathbf{F}_1$ and $\mathbf{F}_2$ are Dirac-like forces as dual quantities to the virtual displacements $\delta\boldsymbol{\xi}(\boldsymbol{\rho}_1, t)$ and $\delta\boldsymbol{\xi}(\boldsymbol{\rho}_2, t)$. Applying the law of interaction for $\delta W^{\mathrm{int}}_{B'}$, with basis subscripts omitted, yields

$$\begin{aligned} 0 = \delta W^{\mathrm{int}}_{B'}(\delta\boldsymbol{\xi}) &= \delta\boldsymbol{\xi}(\boldsymbol{\rho}_1, t)^{\top}\mathbf{F}_1 + \delta\boldsymbol{\xi}(\boldsymbol{\rho}_2, t)^{\top}\mathbf{F}_2 && \forall\delta\boldsymbol{\xi} = \delta\boldsymbol{\xi}^{\mathrm{rfy}} && (5.11)\\ &= \delta\boldsymbol{\phi}^{\top}(\tilde{\boldsymbol{\rho}}_1\mathbf{F}_1 + \tilde{\boldsymbol{\rho}}_2\mathbf{F}_2) + \delta\mathbf{r}_R(\mathbf{F}_1 + \mathbf{F}_2) && \forall\delta\boldsymbol{\phi}, \forall\delta\mathbf{r}_R\ , && (5.12) \end{aligned}$$

from which follows

$$\mathbf{F}_1 = -\mathbf{F}_2\ , \qquad \tilde{\boldsymbol{\rho}}_1\mathbf{F}_1 = \tilde{\boldsymbol{\rho}}_2\mathbf{F}_2 \quad \Leftrightarrow \quad \boldsymbol{\rho}_1 \times \mathbf{F}_1 = \boldsymbol{\rho}_2 \times \mathbf{F}_2\ , \tag{5.13}$$

which requires the forces $\mathbf{F}_1$ and $\mathbf{F}_2$ to be anti-parallel and to have the same magnitude. The requirement (5.13) on the forces $\mathbf{F}_1$ and $\mathbf{F}_2$ is a special case of the *law of interaction*, namely for two Dirac-like forces at the points $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$ and expresses the simple *actio-reactio* principle by Newton's third law.

In the following, we will model an *internal* virtual work contribution of a continuous body B with which we will derive the equation of motion of the *scalable* body in the next section. Therfore, we split the internal virtual work into an *impressed* internal virtual work contribution $\delta W^{\mathrm{int}}_I$ and a *constraint* internal virtual work contribution $\delta W^{\mathrm{int}}_C$ as

$$\delta W^{\mathrm{int}}(\delta\boldsymbol{\xi}) = \delta W^{\mathrm{int}}_C(\delta\boldsymbol{\xi}) + \delta W^{\mathrm{int}}_I(\delta\boldsymbol{\xi})\ . \tag{5.14}$$

The *constraint* internal virtual work contribution defines *internal* forces which constrain the continuous body B with infinite degrees of freedom to a *scalable* body with 7 degrees of freedom given by the constraint motion in (4.2). The *constitutive law* for the *constraint* internal virtual work contribution $W^{\mathrm{int}}_C(\delta\boldsymbol{\xi})$ is modeled to be *perfect*, which means it obeys the principle of d'Alembert-Lagrange.

Principle 5.1 *(Principle of d'Alembert-Lagrange)*:
A constitutive force law formulated by an *internal* virtual work contribution δW_C^{int} which constrains a motion $\boldsymbol{\xi}$ to some submanifold $\mathcal{C} \subset \mathbb{E}^3$ is said to be *perfect* if δW_C^{int} vanishes for all admissible displacements, that is

$$\delta W_C^{\text{int}}(\delta\boldsymbol{\xi}) = 0 \quad \forall \delta\boldsymbol{\xi} \text{ admissible} . \tag{5.15}$$

An admissible displacements $\delta\boldsymbol{\xi}$ lives in the tangent space to the submanifold $\mathcal{C}$ at $\boldsymbol{\xi}$, that is, $\delta\boldsymbol{\xi} \in T_{\boldsymbol{\xi}}\mathcal{C}$.

Hence, the *constraint* internal virtual work W_C^{int} vanishes whenever the virtual displacement field $\delta\boldsymbol{\xi}$ acting on the continuous body is restricted to $\delta\boldsymbol{\xi}_{\text{scal}}$.

The *impressed* internal work contribution for the *scalable* body is modeled only as a contribution in the direction $s(t)$, which is the degree of freedom for the scalability and is the only degree of freedom able to deform the body B or, in other words, able to violate the isometry condition (2.2). The contribution is given by

$$\delta W_I^{\text{int}} = \delta s \lambda_s , \qquad \text{(impressed internal virtual work contribution)} \tag{5.16}$$

$$s, \lambda_s \in \mathcal{D} , \qquad \text{(constitutive force law)} \tag{5.17}$$

where $\mathcal{D}$ denotes some modeled relation between the scaling parameter s and its impressed force λ_s.

The motion under consideration is the motion of a *scalable* body $\boldsymbol{\xi}_{\text{scal}}$ given in (4.1). The law of interaction requires the total *internal* virtual work contribution for any subsystem $B' \subseteq B$ in (5.14) to vanish for all *rigidifying* virtual displacements $\delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}}$ given as

$$\delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}} = \delta\mathbf{r}_R + \delta\tilde{\boldsymbol{\phi}}\ s\mathbf{R}\boldsymbol{\rho} , \tag{5.18}$$

which is obtained by definition (5.9) and (4.1). In the following, we choose $B' = B$. As will be seen in the next section, the *rigidifying* virtual displacements $\delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}}$ are a subset of $\delta\boldsymbol{\xi}_{\text{scal}}$, that is, $\delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}} = \delta\boldsymbol{\xi}_{\text{scal}}|_{\delta s=0} \subset \delta\boldsymbol{\xi}_{\text{scal}}$. Thus, since δW_C^{int} was defined to be *perfect* by the principle of d'Alembert-Lagrange, δW_C^{int} also vanishes for all *rigidifying* displacements $\delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}}$. Hence, the only remaining term still required to vanish by the law of interaction is δW_I^{int}. Applying the law of interaction to (5.16) yields

$$\delta s \lambda_s = 0 \quad \forall \delta\boldsymbol{\xi}_{\text{scal}}^{\text{rfy}} = \delta\boldsymbol{\xi}_{\text{scal}}|_{\delta s=0} , \tag{5.19}$$

which is fulfilled for any λ_s. Thus, (5.16) directly fulfills the variational law of interaction[1] for any subsystem since (5.16) is independent of any subsystem of B.

The approach to split the *internal* virtual work contribution into two parts as in (5.14) is in analogy to the procedure demonstrated in chapter 4.2 in [51], where the motion of a

[1] Precisely, the internal work contribution can be formulated with an impressed stress field which can be split into a force in the direction of the scaling parameter $s(t)$ and a generalized stress tensor $\boldsymbol{\sigma}_s$ as dual quantity to the virtual rotations, that is, $\delta W_I^{\text{int}} = -\delta s\lambda_s - \delta\tilde{\boldsymbol{\phi}} : \boldsymbol{\sigma}_s$. In order to fulfill the *variational law of interaction* which states $\delta\tilde{\boldsymbol{\phi}} : \boldsymbol{\sigma}_s = 0\ \forall \delta\tilde{\boldsymbol{\phi}}, \delta s = 0$, the stress tensor $\boldsymbol{\sigma}_s$ turns out to be symmetric. Furthermore, it will also vanish in the total virtual work of the *scalable* body and is therefore neglected.

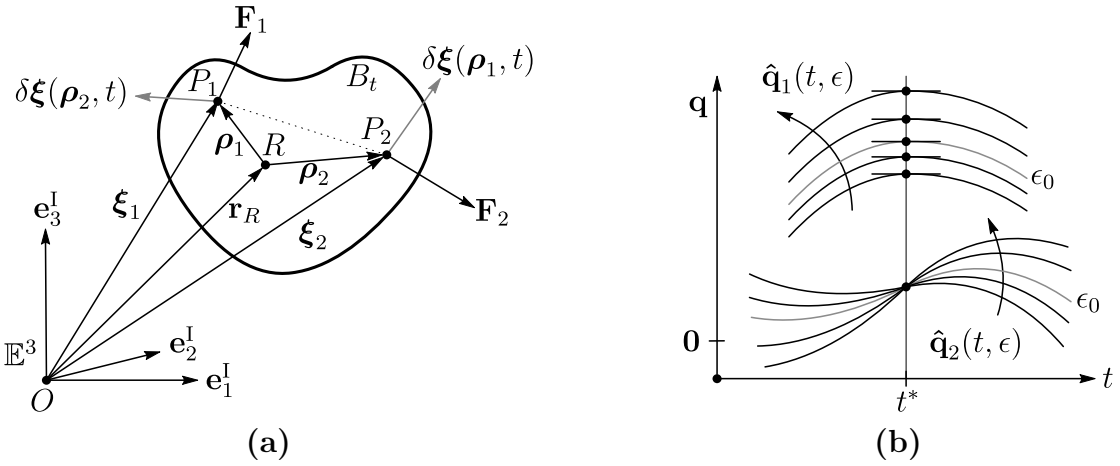


Figure 5.1: (a): Visualization of a virtual displacement field of a subsystem B' consisting of only two points $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$ in (a). The forces $\mathbf{F}_1$ and $\mathbf{F}_2$ are indentified as vectors in $\mathbb{E}^3$. (b): Visualization of two families of varied generalized coordinates $\mathbf{q} \in \mathbb{R}$. Family $\hat{\mathbf{q}}_1(t, \varepsilon)$ has a vanishing time gradient at t^* and family $\hat{\mathbf{q}}_2(t, \varepsilon)$ has a vanishing variation at time t^*.

continuous body is restricted to the kinematics of a deformable beam-like body. The law of interaction, in this case, requires the impressed stress tensor of the impressed internal virtual work to be symmetric.

The above brief discussion on the two fundamental axioms in mechanics, the *principle of virtual work* and *the variational law of interaction*, lays the foundation for the following derivation of the equation of motion of the *scalable* body.

5.1 Scalable and Rigid Body Dynamics

The derivations in this section is based on the work [108] and [170]. This derivation is different as we derive the equation of motion for the *scalable* body directly by the virtual work expression instead of using the Lagrange II formulation.

The virtual work δW of the *scalable* body is assembled from the contributions (5.4), (5.5) and (5.14). The virtual work evaluated with the virtual displacements ${}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}$ induced by the constraint motion ${}_{\mathrm{I}}\boldsymbol{\xi}_{\mathrm{scal}}$ of the *scalable* body represented in the inertial coordinate basis I yields

$$0 = \delta W({}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}) = \delta W^{\mathrm{dyn}} + \delta W_I^{\mathrm{int}} + \underbrace{\delta W_C^{\mathrm{int}}}_{=0 \text{ by principle 5.1}} + \delta W^{\mathrm{ext}} \qquad \forall {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}},\, t \tag{5.20}$$

$$= \int_B {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}{}^{\top}\, {}_{\mathrm{I}}\ddot{\boldsymbol{\xi}}_{\mathrm{scal}} \mathrm{d}m - \delta s \lambda_s - {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}{}^{\top}\, \mathrm{d}\mathbf{F}^{\mathrm{e}} \qquad \forall {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}},\, t\ . \tag{5.21}$$

To evaluate the virtual work (5.21), we directly use the time derivative ${}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{scal}}$ of the motion ${}_{\mathrm{I}}\boldsymbol{\xi}_{\mathrm{scal}}$ in (4.30) with $\mathbf{u}_p := \left[{}_{\mathrm{I}}\dot{\mathbf{r}}_R{}^{\top},\ \nu,\ \boldsymbol{\alpha}^{\top}\right]^{\top}$. The variation ${}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}$ can be deduced from (4.16) by replacing the time derivative with the ε-derivative, that is, $\delta s\mathbf{I} =$

$\mathrm{Re}(2\mathbf{P}^*\delta\mathbf{P})$ where we abbreviate $2\mathbf{P}^*\delta\mathbf{P}$ with the virtual quasi-quaternion $\mathcal{K}(\delta s, \delta\mathbf{a})$. The virtual displacement ${}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}$, in analogy to (4.30), then becomes

$$ {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}} = \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -{}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_R \\ \delta s \\ \delta\mathbf{a} \end{bmatrix} . \tag{5.22} $$

Replacing the time derivative with the ε-derivative in the derivation of (4.10) and using the definition of the virtual quasi-angles ${}_{\mathrm{K}}\delta\boldsymbol{\phi}$ in (5.8) then yields

$$ {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}} = \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -s\,{}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_R \\ \delta s \\ {}_{\mathrm{K}}\delta\boldsymbol{\phi} \end{bmatrix} . \tag{5.23} $$

From comparing (5.22) with (5.23) follows the correspondance between the pure virtual quasi-quaternion parameters $\delta\mathbf{a}$ and the virtual quasi-angles ${}_{\mathrm{K}}\delta\boldsymbol{\phi}$ in analogy to (4.31) as

$$ \delta\mathbf{a} = s\ {}_{\mathrm{K}}\delta\boldsymbol{\phi} . \tag{5.24} $$

Furthermore, $\delta\boldsymbol{\xi}^{\mathrm{rfy}}_{\mathrm{scal}} = \delta\boldsymbol{\xi}_{\mathrm{scal}}|_{\delta s=0}$ in (5.19) is verified by (5.23). The second time derivative ${}_{\mathrm{I}}\ddot{\boldsymbol{\xi}}_{\mathrm{scal}}$ of (4.30) evaluates to

$$ \begin{aligned} {}_{\mathrm{I}}\ddot{\boldsymbol{\xi}}_{\mathrm{scal}} &= \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -{}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \dot{\mathbf{u}}_p + \begin{bmatrix} \mathbf{0} & {}_{\mathrm{K}}\dot{\mathbf{R}}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -{}_{\mathrm{K}}\dot{\mathbf{R}}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \mathbf{u}_p && (5.25) \\ &= \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -{}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \dot{\mathbf{u}}_p + \begin{bmatrix} \mathbf{0} & \frac{1}{s}{}_{\mathrm{K}}\mathbf{R}\,\tilde{\boldsymbol{\alpha}}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -\frac{1}{s}{}_{\mathrm{K}}\mathbf{R}\,\tilde{\boldsymbol{\alpha}}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \mathbf{u}_p && (5.26) \\ &= \begin{bmatrix} \mathbf{I} & {}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}} & -{}_{\mathrm{K}}\mathbf{R}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \dot{\mathbf{u}}_p + \left(\frac{1}{s}{}_{\mathrm{K}}\mathbf{R}\,\tilde{\boldsymbol{\alpha}}\,{}_{\mathrm{K}}\bar{\boldsymbol{\rho}}\nu - \frac{1}{s}{}_{\mathrm{K}}\mathbf{R}\,\tilde{\boldsymbol{\alpha}}\,{}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \right) , && (5.27) \end{aligned} $$

where we have used (4.9) and (4.31) to replace ${}_{\mathrm{K}}\dot{\mathbf{R}}$ with $\frac{1}{s}{}_{\mathrm{K}}\mathbf{R}\tilde{\boldsymbol{\alpha}}$.

Substituting the definitions (5.22) and (5.27) into the *inertia* contribution of the virtual work (5.21) and omitting the subscript K yields

$$ \begin{aligned} -\delta W^{\mathrm{dyn}} = \int_B {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}{}^{\top}\,{}_{\mathrm{I}}\ddot{\boldsymbol{\xi}}_{\mathrm{scal}}\mathrm{d}m = \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_R \\ \delta s \\ \delta\mathbf{a} \end{bmatrix}^{\top} \Bigg(& \int_B \begin{bmatrix} \mathbf{I} & \mathbf{R}\bar{\boldsymbol{\rho}} & -\mathbf{R}\tilde{\bar{\boldsymbol{\rho}}} \\ \times & \bar{\boldsymbol{\rho}}^{\top}\bar{\boldsymbol{\rho}} & -\bar{\boldsymbol{\rho}}^{\top}\tilde{\bar{\boldsymbol{\rho}}} \\ \mathrm{sym.} & \times & \tilde{\bar{\boldsymbol{\rho}}}^{\top}\tilde{\bar{\boldsymbol{\rho}}} \end{bmatrix} \mathrm{d}m\ \dot{\mathbf{u}} \\ & + \int_B \frac{1}{s} \begin{bmatrix} \mathbf{R}\tilde{\boldsymbol{\alpha}}\bar{\boldsymbol{\rho}}\nu + \mathbf{R}\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \\ \bar{\boldsymbol{\rho}}^{\top}\tilde{\boldsymbol{\alpha}}\bar{\boldsymbol{\rho}}\nu - \bar{\boldsymbol{\rho}}^{\top}\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \\ -\tilde{\bar{\boldsymbol{\rho}}}^{\top}\tilde{\boldsymbol{\alpha}}\bar{\boldsymbol{\rho}}\nu + \tilde{\bar{\boldsymbol{\rho}}}^{\top}\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \end{bmatrix} \mathrm{d}m \Bigg) . \end{aligned} \tag{5.28} $$

The expression for the *inertia* contribution (5.28) drastically simplifies if the reference point is chosen to be the center of gravity S of the body, that is, $R = S$. Then, the following integrals vanish:

$$ \int_B {}_{\mathrm{K}}\mathbf{R}_{\mathrm{K}}\bar{\boldsymbol{\rho}}\ \mathrm{d}m = {}_{\mathrm{K}}\mathbf{R}\int_B {}_{\mathrm{K}}\bar{\boldsymbol{\rho}}\ \mathrm{d}m = \mathbf{0} , \qquad \int_B {}_{\mathrm{K}}\mathbf{R}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\ \mathrm{d}m = {}_{\mathrm{K}}\mathbf{R}\int_B {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\ \mathrm{d}m = \mathbf{0} . \tag{5.29} $$

The classical inertia tensor is given as

$$ {}_{\mathrm{K}}\boldsymbol{\Theta}_S \coloneqq \int_B {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}^\top {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \, \mathrm{d}m \;, \qquad \mathrm{Tr}({}_{\mathrm{K}}\boldsymbol{\Theta}_S) = \int_B {}_{\mathrm{K}}\bar{\boldsymbol{\rho}}^\top {}_{\mathrm{K}}\bar{\boldsymbol{\rho}} \, \mathrm{d}m \;, \tag{5.30} $$

where the latter expression can be derived from Lagrange's identity $\tilde{\boldsymbol{\rho}}^\top\tilde{\boldsymbol{\rho}} = \boldsymbol{\rho}^\top\boldsymbol{\rho}\mathbf{I} - \boldsymbol{\rho}\boldsymbol{\rho}^\top$. Furthermore, the term $\boldsymbol{\rho}^\top\tilde{\boldsymbol{\rho}} = (\tilde{\boldsymbol{\rho}}^\top\boldsymbol{\rho})^\top = \mathbf{0}$ vanishes and, with $\tilde{\mathbf{x}}\mathbf{a} = -\tilde{\mathbf{a}}\mathbf{x}$, the terms $\bar{\boldsymbol{\rho}}^\top\tilde{\bar{\boldsymbol{\rho}}}$ and $\bar{\boldsymbol{\rho}}^\top\tilde{\boldsymbol{\alpha}}\bar{\boldsymbol{\rho}}$ in (5.28) vanish as well. With the above definitions, the *inertia* contribution simplifies to

$$ \begin{aligned} -\delta W^{\mathrm{dyn}} = \int_B {}_{\mathrm{I}}\delta\boldsymbol{\xi}_{\mathrm{scal}}{}^\top {}_{\mathrm{I}}\ddot{\boldsymbol{\xi}}_{\mathrm{scal}} \, \mathrm{d}m = \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_S \\ \delta s \\ \delta\mathbf{a} \end{bmatrix}^\top \Bigg(\begin{bmatrix} m\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\,\mathrm{Tr}({}_{\mathrm{K}}\boldsymbol{\Theta}_S) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix} \dot{\mathbf{u}}_p \\ + \frac{1}{s}\int_B \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\alpha}^\top\tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \\ \tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha}\nu + \tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \end{bmatrix} \mathrm{d}m \Bigg) . \end{aligned} \tag{5.31} $$

Evaluating the integral of the second component in (5.31) yields

$$ \int_B -\boldsymbol{\alpha}^\top {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}^\top {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha}\mathrm{d}m = -\boldsymbol{\alpha}^\top {}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \;. \tag{5.32} $$

The term $\tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha}$ in (5.31) can be rewritten as $-\tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\boldsymbol{\alpha}}\tilde{\boldsymbol{\alpha}}\bar{\boldsymbol{\rho}}$ and, by the property of the triple cross product $\tilde{\mathbf{a}}\tilde{\mathbf{b}}\tilde{\mathbf{b}}\mathbf{a} = -\tilde{\mathbf{b}}\tilde{\mathbf{a}}\tilde{\mathbf{a}}\mathbf{b}$, as $\tilde{\boldsymbol{\alpha}}\tilde{\bar{\boldsymbol{\rho}}}^\top\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha}$. Evaluating the integral of the third component in (5.31) yields

$$ \int_B {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}^\top {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha}\nu + \tilde{\boldsymbol{\alpha}}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}^\top {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}}\boldsymbol{\alpha} \, \mathrm{d}m = (\nu\mathbf{I} + \tilde{\boldsymbol{\alpha}})_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \;. \tag{5.33} $$

Substituting (5.32) and (5.33) in (5.31) yields the simplified virtual work contribution of the *inertia* as

$$ -\delta W^{\mathrm{dyn}} = \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_S \\ \delta s \\ \delta\mathbf{a} \end{bmatrix}^\top \Bigg(\begin{bmatrix} m\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\,\mathrm{Tr}({}_{\mathrm{K}}\boldsymbol{\Theta}_S) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix} \dot{\mathbf{u}}_p + \frac{1}{s}\begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\alpha}^\top {}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \\ (\nu\mathbf{I} + \tilde{\boldsymbol{\alpha}})_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \end{bmatrix} \Bigg) . \tag{5.34} $$

Evaluating the *external* virtual work contribution yields

$$ \delta W^{\mathrm{ext}} = \begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_S \\ \delta s \\ \delta\mathbf{a} \end{bmatrix}^\top \begin{bmatrix} {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \\ E_S^{\mathrm{e}} \\ {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \end{bmatrix} , \tag{5.35} $$

$$ {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \coloneqq \int_B {}_{\mathrm{I}}\mathrm{d}\mathbf{F}^{\mathrm{e}} \;, \qquad E_S^{\mathrm{e}} \coloneqq \int_B {}_{\mathrm{K}}\bar{\boldsymbol{\rho}}^\top {}_{\mathrm{K}}\mathbf{R}^\top {}_{\mathrm{I}}\mathrm{d}\mathbf{F}^{\mathrm{e}} \;, \qquad {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \coloneqq \int_B {}_{\mathrm{K}}\tilde{\bar{\boldsymbol{\rho}}} \, {}_{\mathrm{K}}\mathbf{R}^\top {}_{\mathrm{I}}\mathrm{d}\mathbf{F}^{\mathrm{e}} \;, \tag{5.36} $$

where ${}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}}$ denotes the resultant external force in basis I, and E_S^{e} and ${}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}}$ denote the resultant external induced scaling force and moment, respectively, with respect to the center of gravity S in basis K.

Finally, the virtual work formulation (5.21) yields the variational formulation of the kinetic part of the equation of motion for the *scalable* body, that is,

$$\begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_S \\ \delta s \\ \delta\mathbf{a} \end{bmatrix}^{\top} \left(\begin{bmatrix} m\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\operatorname{Tr}({}_{\mathrm{K}}\boldsymbol{\Theta}_S) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix} \dot{\mathbf{u}}_p + \frac{1}{s} \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\alpha}^{\top}\,{}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \\ (\nu\mathbf{I} + \tilde{\boldsymbol{\alpha}})\,{}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \\ E_S^{\mathrm{e}} + \lambda_s \\ {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \end{bmatrix} \right) = 0 \tag{5.37}$$

for all virtual displacements ${}_{\mathrm{I}}\delta\mathbf{r}_S, \delta s, \delta\mathbf{a}$ at every time instant t. Evaluating the variation in (5.37) together with the kinematic part (4.19) yields the equation of motion of the *scalable* body for the generalized coordinates $\mathbf{q} = [{}_{\mathrm{I}}\mathbf{r}_S{}^{\top}, \mathbf{p}^{\top}]^{\top}$ and generalized velocities $\mathbf{u} = [{}_{\mathrm{I}}\dot{\mathbf{r}}_S{}^{\top}, \mathbf{w}^{\top}]^{\top}$ with $\mathbf{w} \coloneqq [\nu, \boldsymbol{\alpha}^{\top}]^{\top}$ as

$$\begin{bmatrix} {}_{\mathrm{I}}\mathbf{r}_S \\ \mathbf{p} \end{bmatrix}^{\cdot} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2s}\boldsymbol{\varphi}_L(\mathbf{P}) \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_S \\ \mathbf{w} \end{bmatrix}, \tag{5.38}$$

$$\begin{bmatrix} m\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\operatorname{Tr}({}_{\mathrm{K}}\boldsymbol{\Theta}_S) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_S \\ \nu \\ \boldsymbol{\alpha} \end{bmatrix}^{\cdot} + \frac{1}{s} \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\alpha}^{\top}\,{}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \\ (\nu\mathbf{I} + \tilde{\boldsymbol{\alpha}})\,{}_{\mathrm{K}}\boldsymbol{\Theta}_S\boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \\ E_S^{\mathrm{e}} + \lambda_s \\ {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \end{bmatrix} = \mathbf{0}\ . \tag{5.39}$$

The *rigid* body formulation is obtained from (5.39) if the scaling parameter s is restricted to one, that is, $s(t) = 1\ \forall t$. The constitutive force law (5.16) which enforces this restriction can be written as

$$s, \lambda_s \in \mathcal{D} = \{s, \lambda_s \mid s = 1,\ \lambda_s \in \mathbb{R}\}\ . \tag{5.40}$$

Equation (5.40) is the force law of a *bilateral* constraint which is *perfect* since it fulfills the principle of d'Alembert-Lagrange, that is, $\delta s \lambda_s = 0\ \forall {}_{\mathrm{I}}\delta\mathbf{r}_R, \delta\mathbf{a}, \delta s = 0$.

Using $\delta\mathbf{a} = s\ {}_{\mathrm{K}}\delta\boldsymbol{\phi}$ in (4.31) and $\boldsymbol{\alpha} = s\ {}_{\mathrm{K}}\boldsymbol{\Omega}$ in (5.24) and the constraint $s = 1$ in (5.40), the virtual work formulation in (5.37) can be transformed to obtain the *rigid* body formulation

$$\begin{bmatrix} {}_{\mathrm{I}}\delta\mathbf{r}_S \\ {}_{\mathrm{K}}\delta\boldsymbol{\phi} \end{bmatrix}^{\top} \left(\underbrace{\begin{bmatrix} m\mathbf{I} & \mathbf{0} \\ \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_S \\ {}_{\mathrm{K}}\boldsymbol{\Omega} \end{bmatrix}^{\cdot} - \underbrace{\begin{bmatrix} \mathbf{0} \\ -{}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}}\ {}_{\mathrm{K}}\boldsymbol{\Theta}_S\ {}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}} \end{bmatrix}}_{\mathbf{h}(\mathbf{q},t)} - \begin{bmatrix} {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \\ {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \end{bmatrix} \right) = 0 \quad \forall {}_{\mathrm{I}}\delta\mathbf{r}_S,\ {}_{\mathrm{K}}\delta\boldsymbol{\phi},\ t\ . \tag{5.41}$$

The matrix $\mathbf{M}$ denotes the constant mass matrix and vector $\mathbf{h}(\mathbf{q}, t)$ denotes the nonlinear term consisting of gyroscopic accelerations. Evaluating the variational term above yields the classical Newton-Euler equations of a rigid body. The equation of motion consisting of kinetic and kinematic part for the *rigid* body finally yields

$$\begin{bmatrix} {}_{\mathrm{I}}\mathbf{r}_S \\ \mathbf{p} \end{bmatrix}^{\cdot} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\boldsymbol{\varphi}_L(\mathbf{P})\,[\mathbf{0}, \mathbf{I}]^{\top} \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_S \\ {}_{\mathrm{K}}\boldsymbol{\Omega} \end{bmatrix}, \qquad \text{(kinematic part)} \tag{5.42}$$

$$\begin{bmatrix} m\mathbf{I} & \mathbf{0} \\ \mathbf{0} & {}_{\mathrm{K}}\boldsymbol{\Theta}_S \end{bmatrix} \begin{bmatrix} {}_{\mathrm{I}}\dot{\mathbf{r}}_S \\ {}_{\mathrm{K}}\boldsymbol{\Omega} \end{bmatrix}^{\cdot} = \begin{bmatrix} \mathbf{0} \\ -{}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}}\ {}_{\mathrm{K}}\boldsymbol{\Theta}_S\ {}_{\mathrm{K}}\tilde{\boldsymbol{\Omega}} \end{bmatrix} + \begin{bmatrix} {}_{\mathrm{I}}\mathbf{F}^{\mathrm{e}} \\ {}_{\mathrm{K}}\mathbf{M}_S^{\mathrm{e}} \end{bmatrix}. \qquad \text{(kinetic part)} \tag{5.43}$$

Note that the quaternion $\mathbf{p}$ describes a rotation ${}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}} = {}_{\mathrm{K}}\mathbf{R}_{\mathrm{KI}}$. The equations of motion (5.42) and (5.43) are used in the further course of this work to describe the dynamics of a granular material by many rigid bodies.

5.2 Equation of Motion for Generalized Coordinates

For the sake of completness, we briefly derive the virtual work expression for a mechanical system B in $\mathbb{E}^3$ undergoing a constrained motion $\boldsymbol{\xi}_\mathrm{c}(\boldsymbol{\rho}, \mathbf{q}(t), t)$, where the configurations are described by generalized coordinates $\mathbf{q}$. The following definitions will be used:

$$\boldsymbol{\xi} := \mathcal{K}_\mathrm{D} \circ \boldsymbol{\xi}_\mathrm{c}(\boldsymbol{\rho}, \mathbf{q}(t), t) \quad \text{(representation in some basis D)} \,, \tag{5.44}$$

$$\mathbf{p} := \int_B \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right]^\top \dot{\boldsymbol{\xi}} \mathrm{d}m \in \mathbb{R}^{n_q} \,, \qquad T(\mathbf{q}, \dot{\mathbf{q}}, t) := \int_B \frac{1}{2} \dot{\boldsymbol{\xi}}^\top \dot{\boldsymbol{\xi}} \mathrm{d}m \in \mathbb{R}_0^+ \,, \tag{5.45}$$

$$\dot{\boldsymbol{\xi}}(t) = \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial \boldsymbol{\xi}}{\partial t} \quad \Rightarrow \quad \delta\boldsymbol{\xi} = \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} \delta\mathbf{q} \tag{5.46}$$

$$\Rightarrow \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} = \frac{\partial \dot{\boldsymbol{\xi}}}{\partial \dot{\mathbf{q}}} \quad \Rightarrow \quad \frac{\mathrm{d}}{\mathrm{d}t} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} \overset{[108]}{=} \frac{\partial \dot{\boldsymbol{\xi}}}{\partial \mathbf{q}} \,. \tag{5.47}$$

The term $T(\mathbf{q}, \dot{\mathbf{q}}, t)$ is the definition for the kinetic energy of the mechanical system B and $\mathbf{p}$ denotes the generalized impulse (not to be confused with the aforementioned quaternion notation). The principle of virtual work (5.1) is now evaluated with the constrained motion $\boldsymbol{\xi}$ given above. We assume that the *impressed* internal virtual work contribution is zero and that the *constraint* internal virtual work contribution, which restricts the unconstrained mechanical system B to the motion $\boldsymbol{\xi}$ with n_q degrees of freedom, follows the principle of d'Alembert-Lagrange and will thus vanish. The principle of virtual work yields

$$0 = \delta W(\delta\boldsymbol{\xi}) = \int_B \delta\boldsymbol{\xi}^\top \ddot{\boldsymbol{\xi}} \mathrm{d}m - \delta\boldsymbol{\xi}^\top \mathrm{d}\mathbf{F}^\mathrm{e} \qquad \forall \delta\boldsymbol{\xi} \,,\, t \tag{5.48}$$

$$= \int_B \left(\frac{\mathrm{d}}{\mathrm{d}t} \left(\delta\boldsymbol{\xi}^\top \dot{\boldsymbol{\xi}} \right) - \frac{\mathrm{d}}{\mathrm{d}t} \left(\delta\boldsymbol{\xi}^\top \right) \dot{\boldsymbol{\xi}} \right) \mathrm{d}m - \int_B \delta\boldsymbol{\xi}^\top \mathrm{d}\mathbf{F}^\mathrm{e} \tag{5.49}$$

$$= \int_B \left(\frac{\mathrm{d}}{\mathrm{d}t} \left(\delta\boldsymbol{\xi}^\top \dot{\boldsymbol{\xi}} \right) - \delta \left(\frac{1}{2} \dot{\boldsymbol{\xi}}^\top \dot{\boldsymbol{\xi}} \right) \right) \mathrm{d}m - \delta\mathbf{q}^\top \underbrace{\int_B \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right]^\top \mathrm{d}\mathbf{F}^\mathrm{e}}_{\mathbf{f}^\mathrm{e}} \tag{5.50}$$

$$\Leftrightarrow 0 = \frac{\mathrm{d}}{\mathrm{d}t} \left(\delta\mathbf{q}^\top \int_B \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right]^\top \dot{\boldsymbol{\xi}} \mathrm{d}m \right) - \delta T(\mathbf{q}, \dot{\mathbf{q}}, t) - \delta\mathbf{q}^\top \mathbf{f}^\mathrm{e} \qquad \forall \delta\mathbf{q} \,,\, t \tag{5.51}$$

$$= \frac{\mathrm{d}}{\mathrm{d}t} \left(\delta\mathbf{q}^\top \mathbf{p} \right) - \delta T(\mathbf{q}, \dot{\mathbf{q}}, t) - \delta\mathbf{q}^\top \mathbf{f}^\mathrm{e} \tag{5.52}$$

$$= \delta\dot{\mathbf{q}}^\top \mathbf{p} + \delta\mathbf{q}^\top \dot{\mathbf{p}} - \frac{\partial T}{\partial \mathbf{q}} \delta\mathbf{q} - \frac{\partial T}{\partial \dot{\mathbf{q}}} \delta\dot{\mathbf{q}} - \delta\mathbf{q}^\top \mathbf{f}^\mathrm{e} \tag{5.53}$$

$$= \delta\mathbf{q}^\top \underbrace{\left(\dot{\mathbf{p}} - \left[\frac{\partial T}{\partial \mathbf{q}}\right]^\top - \mathbf{f}^\mathrm{e} \right)}_{\text{kinetic part}} + \delta\dot{\mathbf{q}}^\top \underbrace{\left(\mathbf{p} - \left[\frac{\partial T}{\partial \dot{\mathbf{q}}}\right]^\top \right)}_{\text{kinematic part}} \,. \tag{5.54}$$

The term $\mathbf{f}^\mathrm{e}$ in (5.50) denotes all *external* generalized forces. Choosing virtual displacements $\delta\mathbf{q}$ in the tangent space of the configuration manifold at $\mathbf{q}$ directly implies specific virtual displacements $\delta\boldsymbol{\xi} = \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} \delta\mathbf{q} \in T_{\boldsymbol{\xi}} \mathbb{E}^3 \cong \mathbb{E}^3$ for which equation (5.48) needs to

hold as well which justifies formulation (5.51). Furthermore, at a specific time t^*, we can restrict the set of virtual displacement fields $\delta\mathbf{q} \in \mathbb{R}^{n_q}$ in (5.54) to the family of virtual displacement fields with $\delta\mathbf{q}(t^*) = \hat{\mathbf{q}}_\varepsilon(t^*, \varepsilon_0)\delta\varepsilon \neq 0$ but with a vanishing gradient $\delta\dot{\mathbf{q}}(t^*) = \hat{\mathbf{q}}_{\varepsilon,t}(t^*, \varepsilon_0)\delta\varepsilon = 0$, see $\hat{\mathbf{q}}_1$ in figure 5.1b). For this specific choice, the second variational term in (5.54) vanishes and only the first variational term is zero for all virtual displacements in the chosen restricted set. By the fundamental lemma of calculus of variations, it follows that the first term in brackets needs to be zero for all times. With the opposite restriction on $\delta\mathbf{q} \in \mathbb{R}^{n_q}$, see $\hat{\mathbf{q}}_2$ in figure 5.1b, the second term in brackets needs to be zero as well. Since the principle of virtual work needs to be fulfilled point-wise in time, both terms need to be zero for all times. It follows that

$$\mathbf{p} = \left[\frac{\partial T}{\partial \dot{\mathbf{q}}}\right]^\top, \quad \dot{\mathbf{p}} = \left[\frac{\partial T}{\partial \mathbf{q}}\right]^\top + \mathbf{f}^\mathrm{e} \quad \forall t \tag{5.55}$$

$$\Leftrightarrow \quad \delta W(\delta\mathbf{q}) = \delta\mathbf{q}^\top \Bigg(\underbrace{\frac{\mathrm{d}}{\mathrm{d}t}\left[\frac{\partial T}{\partial \dot{\mathbf{q}}}\right]^\top - \left[\frac{\partial T}{\partial \mathbf{q}}\right]^\top}_{-\delta W^\mathrm{dyn}} - \mathbf{f}^\mathrm{e} \Bigg) = 0 \quad \forall \delta\mathbf{q}\,,\, t\,. \tag{5.56}$$

Equation (5.56) is the variational formulation of the equation of motion for a mechanical system B undergoing a constrained motion $\boldsymbol{\xi}_\mathrm{c}(\boldsymbol{\rho}, \mathbf{q}(t), t)$ enforced by *perfect* constraints in the sense of d'Alembert-Lagrange under the assumption that the *impressed* internal virtual work contribution is zero. Enforcing the bracket-enclosed term in (5.56) to be zero is very close to *Lagrange's equation of the second kind*. In fact, the *Lagrange II* equations appear when the generalized external forces $\mathbf{f}^\mathrm{e}$ can be expressed as a potential force $\mathbf{f}^\mathrm{e} = -\frac{\partial V(\mathbf{q},t)}{\partial \mathbf{q}}$ by a potential energy $V(\mathbf{q}, t)$.

The kinetic energy $T(\mathbf{q}, \dot{\mathbf{q}}, t)$ in (5.46) can be evaluated as

$$T(\mathbf{q}, \dot{\mathbf{q}}, t) = \frac{1}{2}\dot{\mathbf{q}}^\top \int_B \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right]^\top \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right] \mathrm{d}m\,\dot{\mathbf{q}} + \int_B \left[\frac{\partial \boldsymbol{\xi}}{\partial t}\right]^\top \left[\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}\right] \mathrm{d}m\,\dot{\mathbf{q}} \tag{5.57}$$

$$+ \int_B \frac{1}{2}\left[\frac{\partial \boldsymbol{\xi}}{\partial t}\right]^\top \left[\frac{\partial \boldsymbol{\xi}}{\partial t}\right] \mathrm{d}m \tag{5.58}$$

$$= \frac{1}{2}\dot{\mathbf{q}}^\top \mathbf{M}(\mathbf{q}, t)\dot{\mathbf{q}} + \mathbf{m}_1(\mathbf{q}, \dot{\mathbf{q}}, t)^\top \dot{\mathbf{q}} + m_2(\mathbf{q}, t)\,, \tag{5.59}$$

which shows that the kinetic energy in (5.46) is always a quadratic function in $\dot{\mathbf{q}}$, where $\mathbf{M}(\mathbf{q}, t)$ is generally the positive definite mass matrix and can be viewed as the metric tensor on the configuration manifold. If the mechanical system B does not contain any external excitation, then $\boldsymbol{\xi}$ is not explicitly dependent on t and it follows that $\mathbf{m}_1(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{0}$, $m_2(\mathbf{q}, t) = 0$ in (5.59). Inserting the quadratic form in (5.59) into (5.56) yields

$$\delta W(\delta\mathbf{q}) = \delta\mathbf{q}^\top \big(\mathbf{M}(\mathbf{q}, t)\ddot{\mathbf{q}} + \underbrace{\dot{\mathbf{M}}(\mathbf{q}, t)\dot{\mathbf{q}} + \dot{\mathbf{m}}_1(\mathbf{q}, \dot{\mathbf{q}}, t) - \left[\frac{\partial T}{\partial \mathbf{q}}\right]^\top}_{\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, t)} - \mathbf{f}^\mathrm{e} \big) = 0 \quad \forall \delta\mathbf{q}\,,\, t\,, \tag{5.60}$$

which finally yields the kinetic part of the equation of motion in variational form as

$$\boxed{\delta W(\delta\mathbf{q}) = \delta\mathbf{q}^\top \big(\mathbf{M}(\mathbf{q}, t)\ddot{\mathbf{q}} - \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, t) - \mathbf{f}^\mathrm{e} \big) = 0 \quad \forall \delta\mathbf{q}\,,\, t\,.} \tag{5.61}$$

The term $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, t)$ encompasses all nonlinear terms such as gyroscopic and Coriolis accelerations. To introduce generalized velocities $\mathbf{u}$ by relation (4.13), the virtual work expression in (5.61) is differentiated with respect to time and the time derivative $\delta\dot{\mathbf{q}}$ of the virtual displacement field $\delta\mathbf{q}$ is restricted such that $\underline{\delta}\dot{\mathbf{q}} = \{\delta\dot{\mathbf{q}} \mid \delta\mathbf{q} = 0\}$ (see $\hat{\mathbf{q}}_1$ in figure 5.1b) which yields the virtual power $\underline{\delta}P$ as

$$\underline{\delta}P(\underline{\delta}\dot{\mathbf{q}}) = \underline{\delta}\dot{\mathbf{q}}^\top\big(\mathbf{M}(\mathbf{q},t)\ddot{\mathbf{q}} - \mathbf{h}(\mathbf{q},\dot{\mathbf{q}},t) - \mathbf{f}^{\mathrm{e}}\big) = 0 \quad \forall\underline{\delta}\dot{\mathbf{q}}\,,\, t\ . \tag{5.62}$$

With the help of $\underline{\delta}\dot{\mathbf{q}} = \mathbf{F}\delta\mathbf{u}$ and equation (4.13), it is possible to completely substitute $\dot{\mathbf{q}}$ in (5.62) and obtain the *virtual power* $\underline{\delta}P$ in minimal coordinates as

$$\begin{aligned} &\underline{\delta}P(\underline{\delta}\dot{\mathbf{q}}) = \delta\mathbf{u}^\top\big(\hat{\mathbf{M}}(\mathbf{q},t)\dot{\mathbf{u}} - \hat{\mathbf{h}}(\mathbf{q},\mathbf{u},t) - \hat{\mathbf{f}}^{\mathrm{e}}\big) = 0 \quad \forall\delta\mathbf{u}\,,t \\ &\text{with:} \left[\begin{array}{l} \hat{\mathbf{M}}(\mathbf{q},t) \coloneqq \mathbf{F}^\top\mathbf{M}\mathbf{F}\ , \quad \hat{\mathbf{f}}^{\mathrm{e}} \coloneqq \mathbf{F}^\top\mathbf{f}^{\mathrm{e}}\ , \\ \hat{\mathbf{h}}(\mathbf{q},\mathbf{u},t) \coloneqq \mathbf{F}^\top\Big(\mathbf{h}(\mathbf{q},\mathbf{F}\mathbf{u}+\boldsymbol{\beta},t) - \mathbf{M}\dot{\mathbf{F}}\mathbf{u} - \mathbf{M}\dot{\boldsymbol{\beta}}\Big)\,. \end{array}\right. \end{aligned} \tag{5.63}$$

The reader should note, that equation (4.13) might arise from choosing generalized velocities $\mathbf{u}$ which generate admissible velocities $\dot{\mathbf{q}}$ and admissible virtual velocities $\underline{\delta}\dot{\mathbf{q}}$ with respect to some *kinematic* constraint $\boldsymbol{\gamma}(\mathbf{q},\dot{\mathbf{q}},t) = \mathbf{0}$. The generalized velocity $\mathbf{u}$ is then said to be *minimal* with respect to the kinematic constraint. The kinematic constraint $\boldsymbol{\gamma}(\mathbf{q},\dot{\mathbf{q}},t) = \mathbf{0}$ restricts the velocities $\dot{\mathbf{q}}$ to a subspace generated by $\mathbf{u} \in \mathbb{R}^{n_u}$ in (4.13). Enforcing the kinematic constraint $\boldsymbol{\gamma} = \mathbf{0}$ can be taken care of by an additional *internal* virtual work contribution in (5.62), whose constitutive force law follows the principle of Jourdain. The kinematic constraint is then said to be *perfect* and the constraint force will vanish when performing the transition from (5.62) to (5.63).

Principle 5.2 *(Principle of Jourdain)*:
A constitutive force law formulated by an internal virtual power contribution $\underline{\delta}P_C^{\mathrm{int}}$ which constrains the velocities $\dot{\boldsymbol{\xi}}$ of a motion $\boldsymbol{\xi}$ to some submanifold is said to be *perfect* if $\underline{\delta}P_C^{\mathrm{int}}$ vanishes for all admissible velocities, that is,

$$\underline{\delta}P^{\mathrm{int}}(\underline{\delta}\dot{\boldsymbol{\xi}}) = 0 \quad \forall\underline{\delta}\dot{\boldsymbol{\xi}} \text{ admissible}\ . \tag{5.64}$$

For a given parametrized constrained motion $\boldsymbol{\xi}_{\mathrm{c}}(\boldsymbol{\rho}, \mathbf{q}(t), t)$ of some continuous body, it is possible to derive the equations of motion by (5.48) or by the identical formalism in generalized coordinates in (5.56). However, for non-rigid bodies, the *internal* virtual work contribution is missing and needs to be modeled to obtain a complete description of the dynamics.

The next chapter will first introduce some basic concepts of convex optimization. These concepts will help to describe different constitutive force laws to model the internal interactions between rigid bodies in a granular material and possible external interactions between the granular material and its environment. The constitutive force laws of main interest in this work encompass two set-valued force laws, namely the unilateral contact and the spatial Coulomb friction. These force laws are used to model the dissipative dynamic behavior of a granular medium.

Chapter 6

Convex Optimization

This chapter aims at providing a very basic understanding of convex optimization and motivates the *gradient projection algorithm*, which is a solution method for a general constrained convex optimization problem. The *gradient projection algorithm* is close to the projected Jacobi and Gauss-Seidel iterations used to solve contact problems in rigid body dynamics. Furthermore, this chapter discusses some basic concepts from convex analysis and convex optimization which are used in the further course of this work. The theory presented in this chapter has been summarized to the best of the authors knowledge with the help of [161] and [32]. The reader should note that this chapter is neither mathematically rigorous nor complete.

This chapter mostly deals only with primal and dual coordinate tuples in $\mathbb{R}^n$. A coordinate tuple may still be regarded as a representation of a vector in a given basis. Therefore, we introduce the n-dimensional vector space V over the field $\mathbb{R}$ with standard basis $(\mathbf{e}_1, \dots, \mathbf{e}_n)$ and its corresponding dual vector space, the space of all linear functionals $V^* \coloneqq \{\alpha \mid \alpha \colon V \to \mathbb{R}, \alpha \text{ linear}\}$ with standard dual basis $(\boldsymbol{\epsilon}^1, \dots, \boldsymbol{\epsilon}^n)$ such that $\boldsymbol{\epsilon}^i(\mathbf{e}_j) \coloneqq \delta^i{}_j$, where $\delta^i{}_j$ denotes the Kronecker delta. A vector $\mathbf{x} \in V$ can be represented in the standard basis $(\mathbf{e}_1, \dots, \mathbf{e}_n) \in V$ as $\mathbf{x} = x^i \mathbf{e}_i \in V$. The tuple $[x^1, \dots, x^n]^\top \in \mathbb{R}^n$ is the coordinate representation of $\mathbf{x} \in V$ in the standard basis. Due to the choice of this basis and the fact that V is mostly chosen to be $\mathbb{R}^n$ itself, the coordinate tuple exactly coincides with $\mathbf{x} \in V$ and therefore $\mathbf{x} \equiv [x^1, \dots, x^n]^\top \in \mathbb{R}^n$. Therefore, a bold written term such as $\mathbf{a}$ is used interchangeably for both a vector in V as well as for its standard coordinate representation in $\mathbb{R}^n$. Similarly, a dual vector $\boldsymbol{\alpha}$ is represented in its standard dual basis as $\boldsymbol{\alpha} = \alpha_i \boldsymbol{\epsilon}^i \in V^*$ with coordinate tuple $\boldsymbol{\alpha} \equiv [\alpha_1, \dots, \alpha_n]^\top \in \mathbb{R}^n$.

We define the R-norm of a vector $\mathbf{x} \in V$ to be the induced norm $\|\mathbf{x}\|_R \coloneqq \sqrt{(\mathbf{x} \,|\, \mathbf{x})_R} \in \mathbb{R}_0^+$ by the inner product $(\mathbf{x} \,|\, \mathbf{y})_R \in \mathbb{R}$ which evaluates to $\mathbf{x}^\top \mathbf{R} \mathbf{y} \in \mathbb{R}$ in the standard basis, where $\mathbf{R} \in \mathbb{R}^{n \times n}$ is a positive definite symmetric matrix with entries $R_{ij} = (\mathbf{e}_i \,|\, \mathbf{e}_j)$. The duality pairing of a vector $\boldsymbol{\alpha} \in V^*$ and a vector $\mathbf{x} \in V$ is denoted by $\boldsymbol{\alpha}(\mathbf{a}) \coloneqq \langle \boldsymbol{\alpha} \,|\, \mathbf{a} \rangle \in \mathbb{R}$ which evaluates in the standard basis to $\boldsymbol{\alpha}^\top \mathbf{a} \in \mathbb{R}$.

Since the vector space V is equipped with an inner product $(\mathbf{x} \,|\, \mathbf{y})_R$, the *isomorphism* ϕ_V

induced by this inner product is defined by the linear map (cf. section D.9.2):

$$\begin{array}{rccl} \phi_V : & V & \to & V^* \\ & \mathbf{x} & \mapsto & \boldsymbol{\alpha} = \phi_V(\mathbf{x}) \coloneqq (\mathbf{x} \,|\, \cdot)_R \ . \end{array} \tag{6.1}$$

For finite-dimensional vector spaces, as considered here, the map ϕ_V is bijective and therefore every primal vector $\mathbf{x} \in V$ can be identified with a dual vector $\boldsymbol{\alpha} = \phi_V(\mathbf{x}) \in V^*$ and vice versa by ϕ_V^{-1}. Our definition of the inner product evaluated in the standard basis yields the relation $\boldsymbol{\alpha} = \mathbf{R}\mathbf{x}$ and the inverse relation $\mathbf{x} = \mathbf{R}^{-1}\boldsymbol{\alpha}$.

6.1 Convex Constrained Optimization

An optimization problem $\mathcal{P}$ with convex constraints can be stated in the following form

$$\boxed{\begin{array}{ll} \mathcal{P} : & \min f(\mathbf{x}) \\ & \textit{subject to} \quad \mathbf{x} \in \mathcal{C} \subseteq V \ . \end{array}} \tag{6.2}$$

The set $\mathcal{C} \subseteq V$ is a non-empty closed convex set and the function $f : V \to \mathbb{R} \cup \{\infty\}$ to be minimized is proper and not necessarily convex nor continuously differentiable. Let the extended real line be denoted as $\overline{\mathbb{R}} \coloneqq \mathbb{R} \cup \{\infty\}$. For the further course of this section, we assume any function denoted by f to be *proper*, meaning that $\exists \mathbf{x} \mid f(\mathbf{x}) < \infty$. The domain of a function $f : V \to \overline{\mathbb{R}}$ is the set $\mathrm{dom}(f) \subseteq V$ over which f is well-defined, that is, $\mathrm{dom}(f) \coloneqq \{\mathbf{x} \mid -\infty < f(\mathbf{x}) < \infty\}$. For the optimization problem (6.2), we assume that $\mathrm{dom}(f) \cap \mathcal{C} \neq \{\}$.

Definition 6.1 *(Convex Set)***:**
A set $\mathcal{C} \subseteq V$ is *convex* if and only if

$$\lambda\mathbf{a} + (1-\lambda)\mathbf{b} \in \mathcal{C} \quad \forall \mathbf{a}, \mathbf{b} \in \mathcal{C}, \lambda \in [0,1] \ . \tag{6.3}$$

Definition 6.2 *(Convex Function)***:**
A function $f : \mathcal{C} \to \overline{\mathbb{R}}$, defined on a convex set $\mathcal{C} \subseteq V$ is said to be *convex* on $\mathcal{C}$ if and only if

$$f(\lambda\mathbf{a} + (1-\lambda)\mathbf{b}) \leqslant \lambda f(\mathbf{a}) + (1-\lambda) f(\mathbf{b}) \quad \forall \mathbf{a}, \mathbf{b} \in \mathcal{C}, \lambda \in [0,1] \ . \tag{6.4}$$

Definition 6.3 *(Indicator Function)***:**
The indicator function $I_{\mathcal{C}} : V \to \overline{\mathbb{R}}$ of a set $\mathcal{C} \subseteq V$ is defined as

$$I_{\mathcal{C}}(\mathbf{x}) \coloneqq \begin{cases} 0 & \text{if } \mathbf{x} \in \mathcal{C} \\ \infty & \text{if } \mathbf{x} \notin \mathcal{C} \ . \end{cases} \tag{6.5}$$

If the set $\mathcal{C}$ is convex, then the indicator function $I_{\mathcal{C}}$ is convex and it is also proper if $\mathcal{C} \neq \{\}$.

The duality correspondence of convex functions is expressed by viewing a convex function as an envelope of tangents instead of a locus of points. This duality correspondence is defined by the conjugate of a convex function.

Definition 6.4 *(Conjugate of a Convex Function)*:
The convex conjugate $f^* : V^* \to \overline{\mathbb{R}}$ of a proper convex function $f : V \to \overline{\mathbb{R}}$ is defined as

$$f^*(\mathbf{y}) \coloneqq \sup_{\mathbf{x}\in V}\{\langle \mathbf{y} \,|\, \mathbf{x}\rangle - f(\mathbf{x})\} \ . \tag{6.6}$$

The convex conjugate of the indicator function is called *support* function, that is,

$$I^*_{\mathcal{C}}(\mathbf{y}) = \sup_{\mathbf{x}\in\mathcal{C}}\{\langle \mathbf{y} \,|\, \mathbf{x}\rangle\} \ . \tag{6.7}$$

The conjugation is denoted as $\cdot(\cdot)^*$ and is called *Legendre-Fenchel* transformation. The conjugate of the conjugate of a proper convex function is the function itself, that is, $(f^*)^* = f$.

The notion of a *cone* will extensively be used to describe the set-valued force laws in chapter 7. A set $\mathcal{K} \subseteq V$ is said to be a *cone* if $a\mathbf{x} \in \mathcal{K}\ \forall \mathbf{x} \in \mathcal{K}, \forall a \geq 0$ which is the union of all closed half-lines emanating from the origin $\mathbf{0}$. A cone is called *pointed* if it contains no line, that is, $\mathcal{K} \cup -\mathcal{K} \subseteq \{\mathbf{0}\}$, and otherwise *blunt*.

Definition 6.5 *(Polar Cone)*:
The *polar* cone $\mathcal{C}^\circ$ of a set $\mathcal{C} \subseteq V$ (not necessarily convex), is defined as

$$\mathcal{C}^\circ \coloneqq \{\mathbf{x}^* \mid \langle \mathbf{x}^* \,|\, \mathbf{x}\rangle \leqslant 0\ \forall \mathbf{x} \in \mathcal{C}\} \subseteq V^* \ , \tag{6.8}$$

which is always a convex set.

For convex cones there is the special property that the support function of a convex cone $\mathcal{K}$ is the indicator function of the *polar* cone, that is,

$$I_{\mathcal{K}}{}^* = I_{\mathcal{K}^\circ} \ . \tag{6.9}$$

This can be verified by using equations (6.7) and (6.8).

Definition 6.6 *(Directional Derivative)*:
The one-sided directional derivative $D_{\mathbf{v}}[f(\mathbf{x})]$ in the direction $\mathbf{v} \in V$ of a function $f : V \to \overline{\mathbb{R}}$ at $\mathbf{x} \in V$ is defined to be the limit

$$D_{\mathbf{v}}[f(\mathbf{x})] \coloneqq \lim_{0\leftarrow t} \frac{f(\mathbf{x}+t\mathbf{v}) - f(\mathbf{x})}{t} \tag{6.10}$$

provided it exists, (cf. [161] p. 241). The directional derivative is two-sided if $-D_{-\mathbf{v}}[f(\mathbf{x})] = D_{\mathbf{v}}[f(\mathbf{x})]$.

If a function $f : V \to \overline{\mathbb{R}}$ is differentiable and finite at $\mathbf{x}$, the directional derivative is two-sided and is equal to the duality pairing of the differential $\mathbf{df}(\mathbf{x}) \in V^*$ with the direction $\mathbf{v} \in V$, that is, $D_{\mathbf{v}}[f(\mathbf{x})] = \langle \mathbf{df}(\mathbf{x}) \,|\, \mathbf{v}\rangle$ and f is said to be Gateaux differentiable with Gateaux derivative $\mathbf{df}(\mathbf{x})$. For better readability, $\mathbf{df}(\mathbf{x})$ is abbreviated to $\mathbf{df}_{\mathbf{x}}$. The coordinate tuple of the differential $\mathbf{df}_{\mathbf{a}}$ of a differentiable function f at $\mathbf{a} \in V$ yields

$$\mathbf{df}_{\mathbf{a}} = \left[\frac{\partial f}{\partial \mathbf{x}}(\mathbf{a})\right]^\top = \left[\frac{\partial f}{\partial x_1}(\mathbf{a}), \ldots, \frac{\partial f}{\partial x_n}(\mathbf{a})\right]^\top \in V^* \ , \tag{6.11}$$

which can be verified by setting the direction $\mathbf{v}$ in (6.10) to all standard basis vectors, which yields

$$\forall i \in 1, \ldots, n \;\Rightarrow\; \langle \mathbf{df_a} \,|\, \mathbf{e}_i \rangle = \mathbf{df_a}^\top \mathbf{e}_i = \mathrm{df}_\mathrm{a}^i = \lim_{0 \leftarrow t} \frac{f(\mathbf{a} + t\mathbf{e}_i) - f(\mathbf{a})}{t} = \frac{\partial f}{\partial x_i}(\mathbf{a}) \; . \tag{6.12}$$

With the identification of primal and dual vectors by the inner product (cf. section D.9.2), the gradient with respect to an inner product can be defined.

Definition 6.7 *(Gradient)*:
The gradient $\nabla \mathbf{f}(\mathbf{x}) \coloneqq \nabla \mathbf{f}_\mathbf{x} \in V$ of a differentiable function $f : V \to \overline{\mathbb{R}}$ at $\mathbf{x} \in V$ is defined with respect to the inner product $(\mathbf{x} \,|\, \mathbf{y})_R \coloneqq \mathbf{x}^\top \mathbf{R} \mathbf{y}$ as

$$\boxed{\langle \mathbf{df_x} \,|\, \mathbf{v} \rangle \coloneqq (\nabla \mathbf{f_x} \,|\, \mathbf{v})_R \quad \forall \mathbf{v} \in V \; .} \tag{6.13}$$

This leads to the identification $\mathbf{R}^{-1}\mathbf{df_x} = \nabla \mathbf{f_x}$, which is the identification of $\mathbf{df_x}$ with its primal counterpart, the gradient $\nabla \mathbf{f_x}$, by means of the metric tensor $\mathbf{R}$.

For non-differentiable functions, the notion of the subgradient and subdifferential are extremely useful in the further course of this work. They will be defined as dual quantities belonging to V^*.

Definition 6.8 *(Subgradient)*:
A vector $\mathbf{y} \in V^*$ is said to be a subgradient of a function $f : V \to \overline{\mathbb{R}}$ at $\mathbf{x} \in \mathrm{dom}(f)$ if

$$f(\hat{\mathbf{x}}) \geqslant f(\mathbf{x}) + \langle \mathbf{y} \,|\, \hat{\mathbf{x}} - \mathbf{x} \rangle, \quad \forall \hat{\mathbf{x}} \in \mathrm{dom}(f) \; . \tag{6.14}$$

The subgradient inequality above has a simple geometric meaning when f is finite at $\mathbf{x}$: it says that the affine function

$$h(\mathbf{z}) \coloneqq f(\mathbf{x}) + \langle \mathbf{y} \,|\, \mathbf{z} - \mathbf{x} \rangle \tag{6.15}$$

with subgradient $\mathbf{y}$ is a global underestimator of f. For more information, the reader is refered to [28] and to [161, p. 215] for the convex case.

Definition 6.9 *(Subdifferential)*:
The subdifferential $\partial f(\mathbf{x})$ at a point $\mathbf{x} \in V$ is the set of all subgradients $\mathbf{y}$ of a function $f : V \to \overline{\mathbb{R}}$ at $\mathbf{x}$, that is,

$$\boxed{\partial f(\mathbf{x}) \coloneqq \{\mathbf{y} \mid f(\hat{\mathbf{x}}) \geqslant f(\mathbf{x}) + \langle \mathbf{y} \,|\, \hat{\mathbf{x}} - \mathbf{x} \rangle, \quad \forall \hat{\mathbf{x}} \in \mathrm{dom}(f)\} \; .} \tag{6.16}$$

The subdifferential is always convex as it is the intersection of an infinite system of linear inequalities, (cf. [161] p. 215). A function is called *subdifferentiable* at $\mathbf{x}$ if there exists at least one subgradient at $\mathbf{x}$ and f is called *subdifferentiable* if the latter holds for all $\mathbf{x} \in \mathrm{dom}(f)$. If f is differentiable at $\mathbf{x}$, then $\partial f(\mathbf{x}) = \{\mathbf{df}(\mathbf{x})\}$. Furthermore, if f is continuous at $\mathbf{x}$, the subdifferential $\partial f(\mathbf{x})$ is bounded and if f is convex then $\partial f(\mathbf{x})$ is bounded and non-empty.

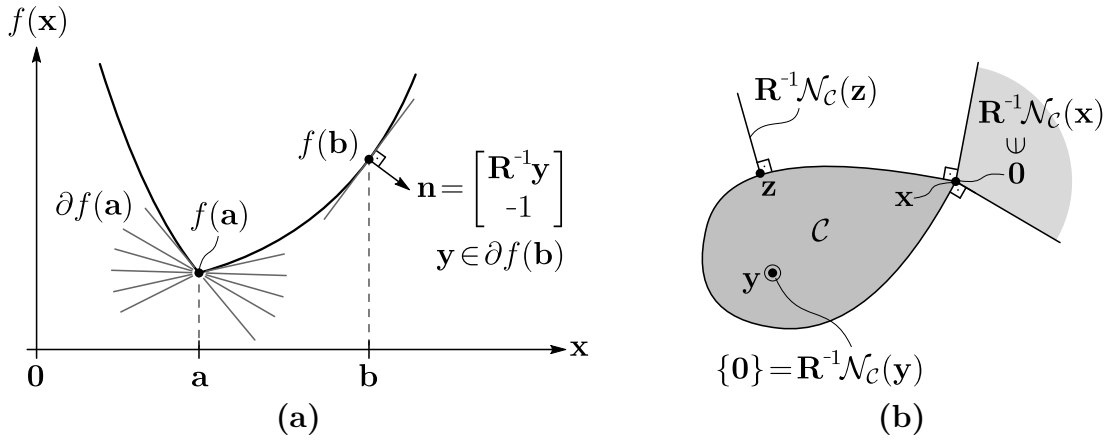


Figure 6.1: (a): Visualization of the subdifferential of the convex function $f(\mathbf{x})$ at the points $\mathbf{a}$ and $\mathbf{b}$ by the tangent lines as given in (6.15). (b): Various normal cones to the convex set $\mathcal{C}$ at the points $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$. Orthogonality in (a) and (b) is meant with respect to the metric $\mathbf{R}$.

The notion of the normal cone to a convex set $\mathcal{C}$ plays an important role for various constitutive laws in mechanics, especially the ones considered in this work and explained in chapter 7, that is, the unilateral contact and the spatial Coulomb friction.

Definition 6.10 *(Normal Cone)*:
The normal cone to a closed convex set $\mathcal{C} \subseteq V$ at point $\mathbf{x} \in \mathcal{C}$ is defined as the set

$$\mathcal{N}_{\mathcal{C}}(\mathbf{x}) := \{\mathbf{y} \mid \langle \mathbf{y} \,|\, \hat{\mathbf{x}} - \mathbf{x} \rangle \leqslant 0, \quad \forall \hat{\mathbf{x}} \in \mathcal{C}\} \subseteq V^* \ . \tag{6.17}$$

The normal cone is a convex cone and the normal cone at $\mathbf{x} \notin \mathcal{C}$ is defined to be the empty set.

The normal cone $\mathcal{N}_{\mathcal{C}}(\mathbf{x})$ is identical to the subdifferential of the indicator function of $\mathcal{C}$ at $\mathbf{x}$, that is, $\mathcal{N}_{\mathcal{C}}(\mathbf{x}) = \partial I_{\mathcal{C}}(\mathbf{x})$. The subdifferential and the normal cone are visualized in figure 6.1

Proof:
By definition 6.10 it follows directly that

$$\partial I_{\mathcal{C}}(\mathbf{x}) = \{\mathbf{y} \mid I_{\mathcal{C}}(\hat{\mathbf{x}}) \geqslant I_{\mathcal{C}}(\mathbf{x}) + \langle \mathbf{y} \,|\, \hat{\mathbf{x}} - \mathbf{x} \rangle, \quad \forall \hat{\mathbf{x}} \in \mathrm{dom}(I_{\mathcal{C}}) = \mathcal{C}\} \ . \tag{6.18}$$

For $\mathbf{x} \notin \mathcal{C}$, this results in the empty set. For $\mathbf{x} \in \mathcal{C}$, this yields $0 \geqslant \langle \mathbf{y} \,|\, \hat{\mathbf{x}} - \mathbf{x} \rangle \ \forall \hat{\mathbf{x}} \in \mathcal{C}$ which is exactly the definition of the normal cone $\mathcal{N}_{\mathcal{C}}(\mathbf{x})$. ■

With the above definitions, it is possible to give some necessary conditions on a minimizer $\mathbf{x}^*$ of the optimization problem $\mathcal{P}$ in (6.2).

Theorem 6.1 *(First-Order Optimality Theorem)*:
Let $\mathbf{x}^*$ be a solution to problem $\mathcal{P}$ in (6.2), meaning $\mathbf{x}^*$ belongs to the set of all minimizers $\mathbf{x}^* \in \{\mathbf{x} \mid f(\mathbf{x}) \leqslant \min_{\mathbf{z}\in\mathcal{C}} f(\mathbf{z})\}$ then it holds that

$$\mathbf{0} \in \partial f(\mathbf{x}^*) + \mathcal{N}_{\mathcal{C}}(\mathbf{x}^*) \ , \tag{6.19}$$
$$\mathbf{0} \in \mathbf{df}_{\mathbf{x}^*} + \mathcal{N}_{\mathcal{C}}(\mathbf{x}^*) \quad \text{(if } f \text{ differentiable)} \ , \tag{6.20}$$
$$\mathbf{0} \in \mathbf{df}_{\mathbf{x}^*} + \mathcal{N}_{\mathcal{C}}(\mathbf{x}^*) \ \Leftrightarrow \ \mathbf{x}^* \text{ global minimizer of } \mathcal{P} \quad \text{(if } f \text{ diff. and convex)} \ . \tag{6.21}$$

If the function f is strictly convex and differentiable, then $\mathbf{x}^*$ is also a unique global minimizer. The statements above can be obtained by writing problem $\mathcal{P}$ as a unconstrained optimization problem $\min g(\mathbf{x})$ with $g(x) \coloneqq f(\mathbf{x}) + I_{\mathcal{C}}(\mathbf{x})$ and requiring that $\mathbf{0} \in \partial g(\mathbf{x})$.

Definition 6.11 *(Generalized Proximal Mapping)*:
The proximal mapping of a convex function $f : V \to \overline{\mathbb{R}}$ at $\mathbf{x}$ is

$$\begin{aligned} \operatorname{prox}_f^R : \ V &\to V \\ \mathbf{x} &\mapsto \mathbf{y} = \operatorname*{argmin}_{\mathbf{x}^*} f(\mathbf{x}^*) + \tfrac{1}{2}\left\|\mathbf{x}-\mathbf{x}^*\right\|_R^2 \ . \end{aligned} \tag{6.22}$$

Definition 6.12 *(Projection onto Convex Set)*:
The proximal function $\operatorname{prox}_{\mathcal{C}}^R$ which maps a point $\mathbf{x}$ to a point $\mathbf{y} \in \mathcal{C}$ such that the squared R-norm distance is minimized is given as

$$\begin{aligned} \operatorname{prox}_{\mathcal{C}}^R : \ V &\to \mathcal{C} \subseteq V \\ \mathbf{x} &\mapsto \mathbf{y} = \operatorname*{argmin}_{\mathbf{x}^*\in\mathcal{C}} \left\|\mathbf{x}-\mathbf{x}^*\right\|_R^2 \ . \end{aligned} \tag{6.23}$$

This definition follows directly from definition 6.11 with $f = I_{\mathcal{C}}$ and therefore $\operatorname{prox}_{\mathcal{C}}^R = \operatorname{prox}_{I_{\mathcal{C}}}^R$.

With this definition, the following projection theorem can be stated:

Theorem 6.2 *(Projection Theorem for Convex Sets)*:
Let $\mathbf{x} \in \mathbb{R}$ and let $\mathcal{C} \subseteq V$ be a non-empty closed convex set. Then $\mathbf{y} \in \mathcal{C}$ solves the problem $\mathbf{y} = \operatorname{prox}_{\mathcal{C}}^R(\mathbf{x})$ (i.e., $\mathbf{y}$ is a proximal point) *if and only if* $\mathbf{R}(\mathbf{x}-\mathbf{y}) \in \mathcal{N}_{\mathcal{C}}(\mathbf{y})$.

Theorem 6.2 can be proven by taking the subdifferential of $\operatorname{prox}_{I_{\mathcal{C}}}^R$ and invoking (6.21) (cf. [108]). Theorem 6.2 can be stated as

$$\begin{aligned} \mathbf{y} = \operatorname{prox}_{\mathcal{C}}^R(\mathbf{x}) \ &\Leftrightarrow \ \mathbf{R}(\mathbf{x}-\mathbf{y}) \in \mathcal{N}_{\mathcal{C}}(\mathbf{y}) \ , \\ \mathbf{y} \in \mathcal{N}_{\mathcal{C}}(\mathbf{x}) \ &\Leftrightarrow \ \mathbf{x} = \operatorname{prox}_{\mathcal{C}}^R(\mathbf{x} + \mathbf{R}^{-1}\mathbf{y}) \ . \end{aligned} \tag{6.24}$$

The second relation in (6.24) can be obtained from the first one by setting $\mathbf{z} = \mathbf{R}(\mathbf{x}-\mathbf{y})$. It is worth noting that the tuple $(\mathbf{x}-\mathbf{y})$ in (6.24) has been transformed with the help of the isomorphism (6.1) from the primal to the dual space. This means that $\mathbf{R}(\mathbf{x}-\mathbf{y})$ is

a dual tuple which is in accordance to the fact that the normal cone belongs to the dual space. For the sake of completeness, the following useful transformation rules derived in [108] are presented:

$$\begin{array}{ccc} \mathbf{y} \in \mathcal{N}_{\mathcal{C}}(\mathbf{x}) & \Leftrightarrow & \mathbf{y} \in \mathbf{A}^{\top}\mathcal{N}_{\mathbf{A}\mathcal{C}}(\mathbf{A}\mathbf{x}) \\ \Updownarrow & & \Updownarrow \\ \mathbf{x} = \operatorname{prox}_{\mathcal{C}}^{R}(\mathbf{x} + \mathbf{R}^{-1}\mathbf{y}) & \Leftrightarrow & \mathbf{A}\mathbf{x} = \operatorname{prox}_{\mathbf{A}\mathcal{C}}^{\hat{R}}(\mathbf{A}\mathbf{x} + \hat{\mathbf{R}}^{-1}\mathbf{A}^{-\top}\mathbf{y}) \ , \end{array} \tag{6.25}$$

where $\mathbf{A}$ is a linear bijective transformation and $\mathbf{R}$ and $\hat{\mathbf{R}}$ are two metrics, that is, two symmetric positive definite matrices. The transformations in (6.25) can be used to simplify projections. For example, if $\mathcal{C}$ is an ellipsoid in $\mathbb{R}^3$, the map $\mathbf{A}$ can be used to transforms $\mathcal{C}$ to a sphere $\mathbf{A}\mathcal{C}$ onto which a projection of a vector can easily be computed.

The projective mapping $\operatorname{prox}_{\mathcal{C}}^{R}$ fulfills the following three important properties:

Lemma 6.1 *(Prox Properties)*:
Let $\mathrm{P} \coloneqq \operatorname{prox}_{\mathcal{C}}^{R}$ be the projection onto the convex set $\mathcal{C}$:

(a) $\mathrm{P} : V \to \mathcal{C}$ is surjective.
(b) $(\mathrm{P}(\mathbf{x}) - \mathbf{x} \,|\, \mathrm{P}(\mathbf{x}) - \mathbf{x}^*)_R \leqslant 0, \quad \forall \mathbf{x}^* \in \mathcal{C}, \quad \mathbf{x} \in V.$
(c) P is a monotone function with respect to the R-norm, that is,

$$(\mathrm{P}(\mathbf{x}_1) - \mathrm{P}(\mathbf{x}_2) \,|\, \mathbf{x}_1 - \mathbf{x}_2)_R \geqslant 0 \ \forall \mathbf{x}_1, \mathbf{x}_2 \in V \ .$$

If $\mathrm{P}(\mathbf{x}_1) \neq \mathrm{P}(\mathbf{x}_2)$ then strict inequality holds.
(d) P is a non-expansive operator, that is, $\|\mathrm{P}(\mathbf{x}_1) - \mathrm{P}(\mathbf{x}_2)\|_R \leqslant \|\mathbf{x}_1 - \mathbf{x}_2\|_R$.

The proof of lemma 6.1 is given in appendix A.1.

If the function f of the optimization problem $\mathcal{P}$ in (6.2) is differentiable, a minimizer $\mathbf{x}^*$ of $\mathcal{P}$ fullfils

$$-\mathbf{df}_{\mathbf{x}^*} \in \mathcal{N}_{\mathcal{C}}(\mathbf{x}^*) \quad \Leftrightarrow \quad \mathbf{x}^* = \operatorname{prox}_{\mathcal{C}}^{R}(\mathbf{x}^* - \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^*}) \ . \tag{6.26}$$

6.2 The Gradient Projection Algorithm

The *gradient projection algorithm* explained in the following was pioneered by Goldstein [65] at the University of Washington in 1964. The equivalent projection method was further developed by Levitin and Polyak [97] in the late seventies, by Bertsekas [23] in the late eighties and recently in [202].

With the help of the aforementioned concepts, we can now state a useful proposition which is used to describe the gradient projection algorithm.

Definition 6.13 *(Projected Descent Direction)*:
We define the projected descent direction at a point $\mathbf{x} \in \mathcal{C}$ as

$$\mathbf{t}(\mathbf{x}) \coloneqq \operatorname{prox}_{\mathcal{C}}^{R}(\mathbf{x} - \nabla\mathbf{f}_{\mathbf{x}}) - \mathbf{x} \in V \quad \text{with: } \nabla\mathbf{f}_{\mathbf{x}} = \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}} \ . \tag{6.27}$$

The reader should note that the descent direction is a function of the negative gradient. For better readability, we abbreviate $\mathbf{t}(\mathbf{x})$ to $\mathbf{t}_\mathbf{x}$ in the further course of this section.

Proposition 6.1 *(Descent)*:
The projected descent direction $\mathbf{t}_\mathbf{x}$ fulfills the property

$$-\|\mathbf{t}_\mathbf{x}\|_R^2 \geqslant (\nabla \mathbf{f}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R, \quad \forall \mathbf{x} \in \mathcal{C} \ . \tag{6.28}$$

Proof:
Expanding the norm of $\mathbf{t}_\mathbf{x}$ at point $\mathbf{x} \in \mathcal{C}$ yields

$$\|\mathbf{t}_\mathbf{x}\|_R^2 = (\mathbf{t}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R = (\mathbf{t}_\mathbf{x} + \mathbf{R}^{-1}\mathbf{df}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R - (\mathbf{R}^{-1}\mathbf{df}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R \ .$$

Inserting the definition for $\mathbf{t}_\mathbf{x}$ and using the abbreviation $\mathbf{z} \coloneqq \mathbf{x} - \mathbf{R}^{-1}\mathbf{df}_\mathbf{x}$ yields

$$\begin{aligned}(\mathbf{t}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R &= \underbrace{(\mathrm{prox}_\mathcal{C}^R(\mathbf{z}) - \mathbf{z} \,|\, \mathrm{prox}_\mathcal{C}^R(\mathbf{z}) - \mathbf{x})_R}_{\leqslant 0 \text{ with lemma 6.1(b)}} - (\mathbf{R}^{-1}\mathbf{df}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R \\ &\leqslant -(\mathbf{R}^{-1}\mathbf{df}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R = -(\nabla \mathbf{f}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R \ .\end{aligned}$$

■

Definition 6.1 means that the gradient $\nabla \mathbf{f}_\mathbf{x}$ projected onto $\mathbf{t}_\mathbf{x}$ or equivalently $D_{\mathbf{t}_\mathbf{x}}[f(\mathbf{x})] \coloneqq \langle \mathbf{df}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x} \rangle = (\nabla \mathbf{f}_\mathbf{x} \,|\, \mathbf{t}_\mathbf{x})_R$ is always non-positive for all $\mathbf{x} \in \mathcal{C}$. This result is especially useful for obtaining a solution to an optimization problem like $\mathcal{P}$ in (6.2). A lot of numerical optimization methods use a descent direction to be able to iteratively converge to a feasible optimal solution of the problem (6.2). The gradient projection algorithm, explained in the following, uses exactly the descent direction $\mathbf{t}_\mathbf{x}$ in (6.27) to converge to a feasible optimal solution.

Based on the observations of the direction $\mathbf{t}_\mathbf{x}$, we are able to formulate the basic gradient projection algorithm shown in algorithm 6.1.

The update formula in (6.31) can be rewritten as

$$\mathbf{x}^{k+1} = (1 - \gamma^{opt})\mathbf{x}^k + \gamma^{opt}\mathrm{prox}_\mathcal{C}^R(\mathbf{x}^k - \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^k}) \ , \tag{6.32}$$

which is a convex combination of two points in the set $\mathcal{C}$ with the parameter $\gamma^{opt} \in (0, 1]$. Therefore, using this update with an initial $\mathbf{x}^0 \in \mathcal{C}$, the sequence $\{\mathbf{x}^k\}$ is guaranteed to stay feasible, that is, $\mathbf{x}^k \in \mathcal{C}\ \forall k$. Another important observation is that (6.32) is basically a weighted step towards the minimum value $\mathbf{y} \in \mathcal{C}$ of the second-order approximated objective function f at $\mathbf{x}^k$ in (6.2), that is,

$$\begin{aligned}\mathbf{y} &= \mathrm{prox}_\mathcal{C}^R(\mathbf{x}^k - \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^k}) && (6.33)\\ &= \underset{\mathbf{x}^* \in \mathcal{C}}{\mathrm{argmin}} \frac{1}{2}\left\|\mathbf{x}^* - \mathbf{x}^k + \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^k}\right\|_R^2 && (6.34)\\ &= \underset{\mathbf{x}^* \in \mathcal{C}}{\mathrm{argmin}} \langle \mathbf{df}_{\mathbf{x}^k} \,|\, \mathbf{x}^* - \mathbf{x}^k \rangle + \frac{1}{2}\left\|\mathbf{x}^* - \mathbf{x}^k\right\|_R^2 + \frac{1}{2}\left\|\mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^k}\right\|_R^2 && (6.35)\\ &= \underset{\mathbf{x}^* \in \mathcal{C}}{\mathrm{argmin}} \langle \mathbf{df}_{\mathbf{x}^k} \,|\, \mathbf{x}^* - \mathbf{x}^k \rangle + \frac{1}{2}(\mathbf{x}^* - \mathbf{x}^k)^\top \mathbf{R}(\mathbf{x}^* - \mathbf{x}^k) && (6.36)\end{aligned}$$

Algorithm 6.1 *(Basic Gradient Projection Algorithm)*:
Let $\mathbf{x}^0 \in \mathcal{C}$ and $\gamma, c \in (0,1]$ and compute $\mathbf{x}_{k+1}$ from $\mathbf{x}_k$ as follows:

1. **Search Direction:** Compute the descent direction as

$$\boxed{\mathbf{t}^k := \operatorname{prox}_{\mathcal{C}}^{R}(\mathbf{x}^k - \nabla \mathbf{f}_{\mathbf{x}^k}) - \mathbf{x}^k \quad \text{with: } \nabla \mathbf{f}_{\mathbf{x}^k} = \mathbf{R}^{-1}\mathbf{df}_{\mathbf{x}^k} \ .} \tag{6.29}$$

2. **Line Search:** Do a *backtracking line search* with the Armijo-Goldstein termination condition as follows:

$$\boxed{\begin{aligned} \gamma^{opt} &= \operatorname*{argmin}_s \ (\gamma)^s \\ \textit{subj. to} \quad & s \in \{1,2,3,\dots\}, \\ & f(\mathbf{x}^k + \gamma^s \mathbf{t}^k) \leqslant f(\mathbf{x}^k) + c\ \gamma^s \langle \mathbf{df}_{\mathbf{x}^k} \,|\, \mathbf{t}^k \rangle \ . \end{aligned}} \tag{6.30}$$

3. **Update:**

$$\boxed{\mathbf{x}^{k+1} = \mathbf{x}^k + \gamma^{opt}\mathbf{t}^k \ .} \tag{6.31}$$

$$= \operatorname*{argmin}_{\mathbf{x}^* \in \mathcal{C}} \ f(\mathbf{x}^k) + \langle \mathbf{df}_{\mathbf{x}^k} \,|\, \mathbf{x}^* - \mathbf{x}^k \rangle + \frac{1}{2}(\mathbf{x}^* - \mathbf{x}^k)^\top \mathbf{R}(\mathbf{x}^* - \mathbf{x}^k) \ . \tag{6.37}$$

From (6.37) it follows that the metric $\mathbf{R}$ plays the role of the Hessian of the objective function f in (6.2). The optimization problem in (6.37) can also be stated in the descent direction $\mathbf{t} = \mathbf{x}^* - \mathbf{x}^k$ as

$$\mathbf{t}^k = \operatorname*{argmin}_{\mathbf{t} \in \mathcal{C} - \mathbf{x}^k} \ f(\mathbf{x}^k) + \langle \mathbf{df}_{\mathbf{x}^k} \,|\, \mathbf{t} \rangle + \frac{1}{2}\mathbf{t}^\top \mathbf{R}\mathbf{t} \ . \tag{6.38}$$

If $\mathbf{R}$ is chosen to be an approximation of the Hessian of f at $\mathbf{x}^k$ when determining the search direction in the first step in algorithm 6.1, the gradient projection method becomes a proximal Newton-type method as presented in [92]. It has to be noted that evaluating the projection in a general metric $\mathbf{R}$ is anything but trivial for a general convex set $\mathcal{C}$. However, if the convex set $\mathcal{C}$ is a Cartesian product of several convex sets, that is, $\mathcal{C} = \mathcal{C}_i \times \cdots \times \mathcal{C}_n$, the projections can be split up into individual projections onto the sets $\mathcal{C}_i$ if the metric $\mathbf{R}$ is split up into diagonal block entries $\mathbf{R}_i \in \mathbb{R}^{d_i \times d_i}$ with $d_i = \dim(\mathcal{C}_i)$, that is, $\mathbf{R} = \operatorname{diag}(\mathbf{R}_1, \dots, \mathbf{R}_n)$. Furthermore, the projection in (6.29) drastically simplifies if $\mathbf{R}$ (or equivalently $\mathbf{R}_i$) is chosen proportional to the identity matrix, that is, $\mathbf{R} = r\mathbf{I},\ r > 0$ or $\mathbf{R}_i = r_i\mathbf{I},\ r_i > 0$. These simplifications become especially important for solving the contact problem in section 8.3.

Why the choice of $\mathbf{R}$ to be an approximation of the Hessian of f at $\mathbf{x}^k$ is favorable can be seen by the following simple example. Take the quadratic optimization function

$$\min_{\mathbf{x} \in \mathcal{C}} f(\mathbf{x}) = \min_{\mathbf{x} \in \mathcal{C}} \frac{1}{2}\mathbf{x}^\top \mathbf{G}\mathbf{x} + \mathbf{c}^\top \mathbf{x} \tag{6.39}$$

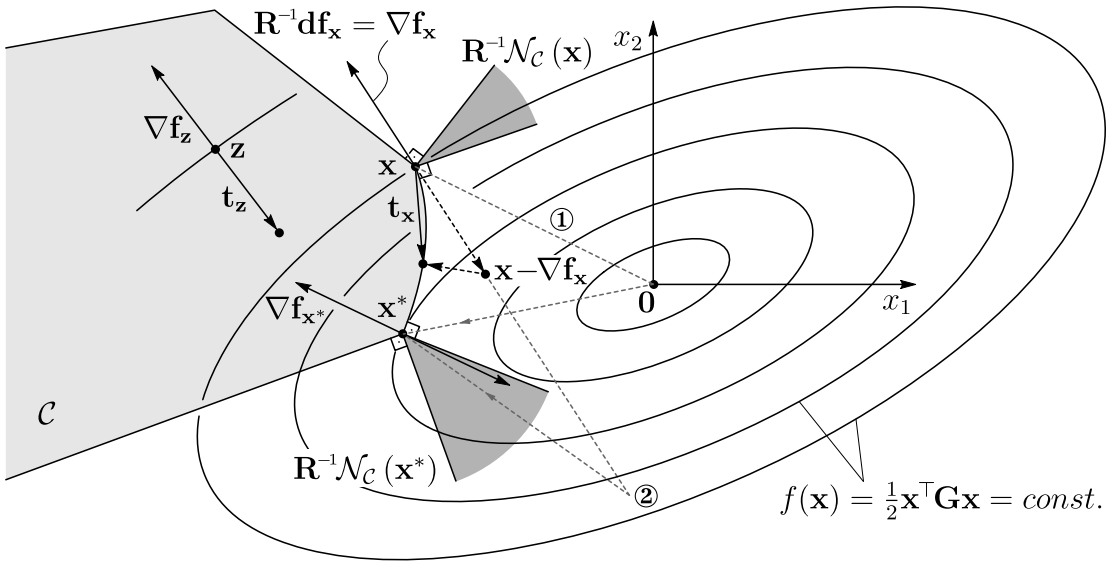


Figure 6.2: Gradient projection algorithm applied to the convex constrained optimization problem in (6.39) with a convex set $\mathcal{C}$ in the space $\mathbb{E}^2$ with metric $\mathbf{R} = \mathbf{I}$, $\mathbf{c} = \mathbf{0}$ and $\gamma = 1$.

over the convex set $\mathcal{C}$ and assume $\mathbf{G}$ to be symmetric and positive definite such that f is strictly convex. Assume $\mathbf{x}^*$ is a minimizer of f. Then by the first order optimality condition, $\mathbf{x}^*$ fullfils

$$-(\mathbf{G}\mathbf{x}^* + \mathbf{c}) \in \mathcal{N}_C(\mathbf{x}^*) \quad \Leftrightarrow \quad \mathbf{x}^* = \mathrm{prox}_\mathcal{C}^R(\mathbf{x}^* - \mathbf{R}^{-1}\mathbf{df_{x^*}}) \; . \tag{6.40}$$

If we choose the metric $\mathbf{R}$ to be the Hessian of f, that is, $\mathbf{R} = \mathbf{G}$, and apply the update step in (6.32) with $\gamma = 1$ as

$$\mathbf{x}^{k+1} = \mathrm{prox}_\mathcal{C}^R\left(\mathbf{x}^k - \mathbf{R}^{-1}(\mathbf{G}\mathbf{x}^* + \mathbf{c})\right) = \mathrm{prox}_\mathcal{C}^R(\mathbf{c}) \tag{6.41}$$

directly results in the solution $\mathbf{x}^* = \mathrm{prox}_\mathcal{C}^R(\mathbf{c})$ to the optimization problem (6.39) in just a single step. The level sets of the quadratic optimization function in (6.39) for the two-dimensional case $\mathbb{R}^2$ with $\mathbf{c} = \mathbf{0}$ are depicted in figure 6.2. The unconstrained minimum lies at the center $\mathbf{0}$. The orthogonality between vectors in figure 6.2 is meant with respect to the inner product with metric $\mathbf{R} = \mathbf{I}$. The optimal value $\mathbf{x}^*$ fullfils the first-order optimality theorem in (6.1), that is, the negative differential $\mathbf{df_{x^*}}$ is an element of the normal cone at the point $\mathbf{x}^*$. Two descent steps $\mathbf{t_x}$ at $\mathbf{x}$ and $\mathbf{t_z}$ at $\mathbf{z}$ are visualized. Since the result of the step $\mathbf{t_z}$ is inside the set $\mathcal{C}$, the mapping $\mathrm{prox}_\mathcal{C}^R$ is the identity map. Depending on the choice of the metric $\mathbf{R}$, the update step might directly project to the optimal value $\mathbf{x}^*$ as indicated with gray dashed lines denoted by ① and ②. The descent step denoted by ① is obtained if $\mathbf{R} = \mathbf{G}$ and step ② if $\mathbf{R} = s\mathbf{I}$ with $s > 0$. The gradient projection algorithm 6.1 will mentioned again when discussion the contact problem in section 8.3. The next chapter discusses several contact laws between rigid bodies with the help of the preliminaries introduced in this chapter.

Chapter 7

Contact Laws

To model the interaction between bodies in a rigid body assembly, constitutive force laws are formulated which fulfill the variational law of interaction 5.2. A constitutive force law is a relation between a geometric quantity $\mathbf{g}(\mathbf{q}, t)$ on displacement level or a kinematic quantity $\boldsymbol{\gamma}(\mathbf{q}, \mathbf{u}, t)$ on velocity level and a corresponding dual quantity, that is, a force $\boldsymbol{\lambda}$. Formulating a constitutive force law on displacement level means adding a corresponding internal (or external) virtual work contribution to the virtual work expression in axiom 5.1. An internal (or external) virtual power contribution is needed if the force law is formulated on velocity level. In the following, we discuss constitutive force laws in the physical space $\mathbb{E}^3$ and subscripts denoting the coordinate representation are neglected since the virtual work is independent of any coordinate representation. Furthermore, we concentrate on internal forces as the discussion for external forces is similar. A constitutive force law on displacement level can be expressed in generalized coordinates $\mathbf{q}$ as

$$\begin{aligned} \delta W^{\text{int}}(\delta\mathbf{g}) &:= \delta\mathbf{g}^\top \boldsymbol{\lambda} \ , && \text{(internal virtual work contribution)} \\ (\mathbf{g}(\mathbf{q}, t), \boldsymbol{\lambda}) &\in \mathcal{R} \ . && \text{(constitutive force law)} \end{aligned} \tag{7.1}$$

The f-dimensional geometric quantity $\mathbf{g} \in V \cong \mathbb{R}^f$ measures some abstract form of displacement in the mechanical system of interest. Such a measurement can include a length between two points or a relative angle between two bodies or some more abstract measure as the sum of various angles between lines. The dual quantity to the virtual displacement $\delta\mathbf{g}$ is the force $\boldsymbol{\lambda} \in V^* \cong \mathbb{R}^{f*}$. The internal virtual work contribution in (7.1) can be rewritten in terms of the generalized virtual displacement $\delta\mathbf{q}$ and generalized force $\mathbf{f}$ in analogy to the virtual work of the external forces in (5.50) as

$$\delta W^{\text{int}}(\delta\mathbf{q}) = \delta\mathbf{q}^\top \mathbf{f} \stackrel{!}{=} \delta\mathbf{g}^\top \boldsymbol{\lambda} = \left(\frac{\partial \mathbf{g}}{\partial \mathbf{q}} \delta\mathbf{q}\right)^\top \boldsymbol{\lambda} \quad \Rightarrow \quad \mathbf{f} = \mathbf{W}\boldsymbol{\lambda} \ , \quad \mathbf{W}(\mathbf{q}, t) := \left[\frac{\partial \mathbf{g}}{\partial \mathbf{q}}\right]^\top . \tag{7.2}$$

The change of coordinates in (7.2) allows to add the term $\delta\mathbf{q}^\top \mathbf{W}\boldsymbol{\lambda}$ to the virtual work expression of the mechanical system in the variational formulation (5.61). The matrix $\mathbf{W}$ denotes the generalized force direction of the constitutive force law. The binary relation $(V, V^*, \mathcal{R})$ with graph $\mathcal{R} \subseteq V \times V^*$ in (7.1) defines the constitutive force law. Specifying the graph $\mathcal{R}$ is a modeling task and can represent, for example, a nonlinear spring or a

bilateral constraint in a mechanical system. In general, a graph $\mathcal{R}$ can be of set-valued nature.

A force law on velocity level can be expressed in the same way by an internal virtual power contribution as

$$\begin{aligned} \underline{\delta} P^{\text{int}}(\underline{\delta}\boldsymbol{\gamma}) &\coloneqq \underline{\delta}\boldsymbol{\gamma}^\top \boldsymbol{\lambda} \ , && \text{(internal virtual work contribution)} \\ (\boldsymbol{\gamma}(\mathbf{q},\mathbf{u},t),\boldsymbol{\lambda}) &\in \mathcal{S} \ . && \text{(constitutive force law)} \end{aligned} \tag{7.3}$$

The f-dimensional kinematic quantity $\boldsymbol{\gamma}(\mathbf{q},\mathbf{u},t) \in V \cong \mathbb{R}^f$ measures some abstract form of velocity in the mechanical system of interest, this includes for example a relative velocity between two bodies, either translational or rotational or some sort of combination. As for the geometric constraint in (7.1), the binary relation with graph $\mathcal{S} \subset V \times V^*$ links the kinematic quantity $\boldsymbol{\gamma} \in V \cong \mathbb{R}^f$ to its corresponding force $\boldsymbol{\lambda} \in V^* \cong \mathbb{R}^{f*}$. The internal power contribution can be written as

$$\underline{\delta} P^{\text{int}} = \delta\mathbf{u}^\top \hat{\mathbf{f}} \overset{!}{=} \underline{\delta}\boldsymbol{\gamma}^\top \boldsymbol{\lambda} = \left(\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{u}} \delta\mathbf{u}\right)^\top \boldsymbol{\lambda} = \delta\mathbf{u}^\top \hat{\mathbf{W}} \boldsymbol{\lambda} \tag{7.4}$$

$$\Rightarrow \hat{\mathbf{f}} = \hat{\mathbf{W}}\boldsymbol{\lambda} \ , \quad \hat{\mathbf{W}} \coloneqq \left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{u}}\right]^\top . \tag{7.5}$$

The gap function in (7.1) can be differentiated with respect to time as

$$\dot{\mathbf{g}}(\mathbf{q},\dot{\mathbf{q}},t) = \frac{\partial \mathbf{g}}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial \mathbf{g}}{\partial t} = \mathbf{W}(\mathbf{q},t)^\top \dot{\mathbf{q}} + \boldsymbol{\chi}(\mathbf{q},t) \tag{7.6}$$

and can formulated in terms of $\mathbf{u}$ with help of (4.13) as

$$\dot{\mathbf{g}}(\mathbf{q},\dot{\mathbf{q}},t) = \dot{\mathbf{g}}(\mathbf{q},\mathbf{F}(\mathbf{q},t)\mathbf{u} + \boldsymbol{\beta}(\mathbf{q},t),t) = \hat{\mathbf{W}}(\mathbf{q},t)^\top \mathbf{u} + \hat{\boldsymbol{\chi}}(\mathbf{q},t) = \boldsymbol{\gamma}(\mathbf{q},\mathbf{u},t) \ , \tag{7.7}$$

where

$$\hat{\mathbf{W}}(\mathbf{q},t) \coloneqq \mathbf{F}(\mathbf{q},t)^\top \mathbf{W}(\mathbf{q},t), \qquad \hat{\boldsymbol{\chi}}(\mathbf{q},t) \coloneqq \mathbf{W}(\mathbf{q},t)^\top \boldsymbol{\beta}(\mathbf{q},t) + \boldsymbol{\chi}(\mathbf{q},t) \ . \tag{7.8}$$

The relation between the generalized force $\mathbf{f}$ and $\hat{\mathbf{f}}$ is given as $\hat{\mathbf{f}} = \mathbf{F}(\mathbf{q},t)^\top \mathbf{f}$. The internal power contribution in (7.5) can be added to the variational formulation (5.63).

The opposite way is in general not possible, meaning that not every constitutive law on velocity level, for example $\dot{\mathbf{g}}(\mathbf{q},\dot{\mathbf{q}},t)$ or $\boldsymbol{\gamma}(\mathbf{q},\mathbf{u},t)$ in (7.7), can be described with an equivalent formulation (7.1) on displacement level. One example of such a constitutive force law is the Coulomb friction law discussed in the further course of this chapter.

Binary relations which describe bilateral geometric and bilateral kinematic constraints, respectively, can be described by the following graphs

$$\mathcal{R}_c \coloneqq \{(\mathbf{g},\boldsymbol{\lambda}) \in V \times V^* \mid \mathbf{g}(\mathbf{q},t) = \mathbf{0}\} \ , \qquad \text{(geometric constraint)} \tag{7.9}$$

$$\mathcal{S}_c \coloneqq \{(\mathbf{g},\boldsymbol{\lambda}) \in V \times V^* \mid \boldsymbol{\gamma}(\mathbf{q},\mathbf{u},t) = \mathbf{0}\} \ . \qquad \text{(kinematic constraint)} \tag{7.10}$$

The geometric constraint in (7.9) fulfills the principle of d'Alembert-Lagrange 5.1 and analogously, the kinematic constraint in (7.10) fulfills the principle of Jourdain 5.2. A

kinematic constraint which can not be described by an equivalent geometric constraint is called *non-holonomic*.

The graph $\mathcal{R}$ (or $\mathcal{S}$) of a binary relation can be represented by a set-valued (also called multi-valued) function $F\colon V \to \mathbb{P}(V^*)$, where $\mathbb{P}(V^*)$ denotes the power set of V^* such that the graph $\mathcal{G}_F$ of F is identical to $\mathcal{R}$, that is,

$$\mathcal{G}_F \coloneqq \{(\mathbf{x}, \mathbf{y}) \in V \times V^* \mid \mathbf{y} \in F(\mathbf{x})\} = \mathcal{R} \ . \tag{7.11}$$

We will see in the next section, when discussing constitutive force laws for contacts, that the set-valued function F is given by a subdifferential of a proper convex lower semi-continuous function, also called *potential*. This is especially convenient as it embeds the constitutive force laws in the field of convex analysis and convex optimization which are two well-studied fields in mathematics. The algorithm used in this work for solving set-valued force laws in multi-body dynamics is based on the gradient projection method presented in section 6.2.

The bilateral constraints in (7.9) and (7.10) can be formulated with the help of the normal cone in definition 6.10 as

$$\mathcal{R}_c \coloneqq \{(\mathbf{g}, \boldsymbol{\lambda}) \in V \times V^* \mid -\boldsymbol{\lambda} \in \mathcal{N}_{\{\mathbf{0}\}}(\mathbf{g})\} \qquad \text{(geometric constraint)} \ , \tag{7.12}$$

$$\mathcal{S}_c \coloneqq \{(\boldsymbol{\gamma}, \boldsymbol{\lambda}) \in V \times V^* \mid -\boldsymbol{\lambda} \in \mathcal{N}_{\{\mathbf{0}\}}(\boldsymbol{\gamma})\} \ , \qquad \text{(kinematic constraint)} \tag{7.13}$$

where $-\mathcal{N}_{\{\mathbf{0}\}}$ corresponds to F in (7.11).

7.1 Force Laws for Contacts

This section covers different formulations of set-valued contact laws between rigid bodies, especially the unilateral contact with Coulomb friction. For a broader treatment of set-valued force laws the reader is referred to [60, 93].

A constitutive force law for a contact, referred to as contact force law, describes the interaction of two contacting surfaces of two different bodies. The contact surfaces of interest in the context of this work are boundary surface points of two convex rigid bodies in $\mathbb{E}^3$. To overcome the convexity restriction for a general multi-body simulation with non-convex bodies, lots of geometric precautions and intricacies have to be dealt with. For the context of this work, we will look only at convex rigid bodies which simplifies the discussion a great deal. Figure 7.1 shows a free-body diagram of two detached contacting convex bodies B_a and B_b which share a common contact point C which is denoted by C_a for body B_a and by C_b for body B_b. The gap distance $\mathbf{g}$ is given as the difference of the displacement vectors of two points C_a and C_b, that is, $\mathbf{g} \coloneqq \mathbf{r}_{C_a} - \mathbf{r}_{C_b}$. If the gap distance $\mathbf{g} = \mathbf{0}$, then $C_a = C_b = C$. The convex tangent cone at the contact point C to the convex set B_a is denoted as $\mathcal{T}_{C_a} \coloneqq \mathcal{T}_{B_a}(\mathbf{r}_{C_a}) \subset \mathbb{E}^3$ and $\mathcal{T}_{C_b} \subset \mathbb{E}^3$ for body B_b. Especially for the case of the *unilateral contact*, which describes the impenetrability condition of the two bodies, it is necessary to define a common tangent plane $\mathcal{T}_C$ at the contact point C. A common tangent plane $\mathcal{T}_C$ which separates the two bodies (except for the contact point)

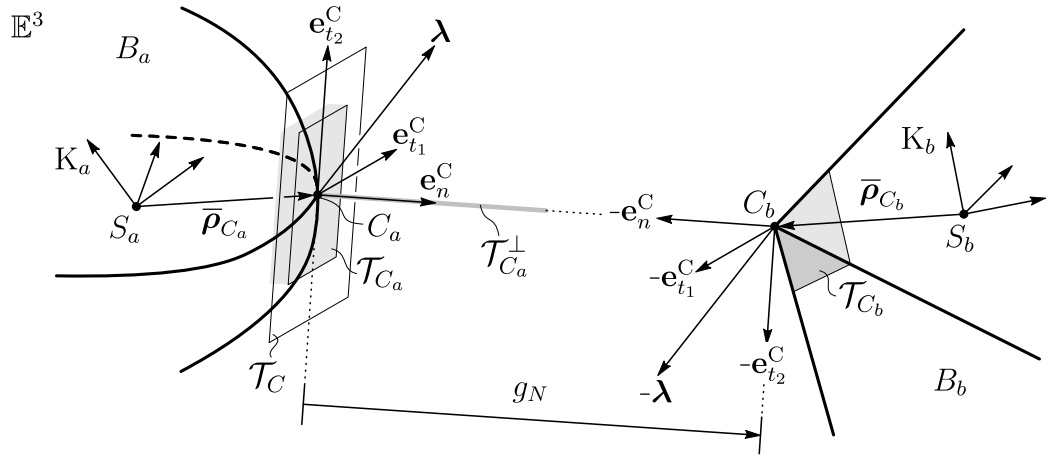


Figure 7.1: Detached contact configuration for two convex bodies $B_a \subset \mathbb{E}^3$ and $B_b \subset \mathbb{E}^3$ having contact at a common contact point C. Body B_a with center of gravity S_a has a smooth boundary surface and body B_b with center of gravity S_b has a polygonal boundary surface. The convex tangent cone $\mathcal{T}_{C_a}$ covers the entire half-line in direction $-\mathbf{e}^{\mathrm{C}}_n$ and similar for the tangent cone $\mathcal{T}_{C_b}$ which is generated by the three outgoing edges of body B_b at C_b. The contact force is identified in $\mathbb{E}^3$ and denoted by $\boldsymbol{\lambda}$.

can be defined by choosing a normal vector $\mathbf{e}^{\mathrm{C}}_n$ from the intersection of the orthogonal cones $\mathcal{T}^{\perp}_{C_a}$ and $-\mathcal{T}^{\perp}_{C_b}$, that is,

$$\mathbf{e}^{\mathrm{C}}_n \in \mathcal{T}^{\perp}_{C_a} \cap -\mathcal{T}^{\perp}_{C_b} \,. \tag{7.14}$$

The orthogonal cone $\mathcal{T}^{\perp} \subseteq \mathbb{E}^3$ to a convex cone $\mathcal{T} \subseteq \mathbb{E}^3$ with respect to a given inner product $(\bullet \,|\, \bullet)$ is defined as

$$\mathcal{T}^{\perp} \coloneqq \{\mathbf{x} \mid (\mathbf{x} \,|\, \mathbf{y}) \leqslant 0 \quad \forall \mathbf{y} \in \mathcal{T}\} \tag{7.15}$$

and is also convex. A local orthonormal contact coordinate system C with basis vectors $\bar{\bar{\mathbf{e}}}^{\mathrm{C}} \coloneqq \{\mathbf{e}^{\mathrm{C}}_n, \mathbf{e}^{\mathrm{C}}_{t_1}, \mathbf{e}^{\mathrm{C}}_{t_2}\}$ is then defined such that the two tangential vectors $\mathbf{e}^{\mathrm{C}}_{t_1}, \mathbf{e}^{\mathrm{C}}_{t_2}$ span the tangent plane $\mathcal{T}_C$ with normal vector $\mathbf{e}^{\mathrm{C}}_n$. The local contact coordinate system C at a contact C is used to represent set-valued contact force laws explained in the following. The intersection for the case depicted in figure 7.1 is trivial since $\mathcal{T}^{\perp}_{C_a} = \{\alpha \mathbf{e}^{\mathrm{C}}_n, \alpha \geqslant 0\}$. The gap distance in normal direction g_N is the first coordinate of the representation of $\mathbf{r}_{C_aC_b}$ in coordinate system C, that is, $g_N = (\mathbf{e}^{\mathrm{C}}_n \,|\, \mathbf{r}_{C_aC_b})$. For certain geometries, the intersection in (7.14) yields not a unique element. In figure 7.1, this would be the case if body B_a had the same polygonal geometry as body B_b. In this case, there exists a set of possible (infinitely many) tangent planes $\mathcal{T}_C$. A constitutive force law for the impenetrability of two convex bodies which takes this ambiguity of the normal direction into account is out of the scope of this work, and the choice of the common tangent plane $\mathcal{T}_C$ defined by the normal vector $\mathbf{e}^{\mathrm{C}}_n$ is treated in this work as a model assumption. Thus, the impenetrability condition described by the unilateral contact discussed in the next section is formulated with a single normal direction $\mathbf{e}^{\mathrm{C}}_n$ fulfilling (7.14), a scalar normal gap distance g_N and a scalar normal force λ_N.

A set-valued force law for a contact on velocity level relates the relative velocity $\boldsymbol{\gamma}(\mathbf{q}, \mathbf{u})$ between C_b and C_a expressed in coordinate system C as

$$\boldsymbol{\gamma} \coloneqq \gamma_N \mathbf{e}_n^{\mathrm{C}} + \boldsymbol{\gamma}_T \in \mathbb{E}^3 \ , \qquad \boldsymbol{\gamma}_T \coloneqq \gamma_{T_1} \mathbf{e}_{t_1}^{\mathrm{C}} + \gamma_{T_2} \mathbf{e}_{t_2}^{\mathrm{C}} \tag{7.16}$$

to the contact force

$$\boldsymbol{\lambda} \coloneqq \lambda_N \boldsymbol{\epsilon}_n^{\mathrm{C}} + \boldsymbol{\lambda}_T \in \mathbb{E}^{3*} \ , \qquad \boldsymbol{\lambda}_T \coloneqq \lambda_{T_1} \boldsymbol{\epsilon}_{t_1}^{\mathrm{C}} + \lambda_{T_2} \boldsymbol{\epsilon}_{t_2}^{\mathrm{C}} \ , \tag{7.17}$$

where $(\boldsymbol{\epsilon}_n^{\mathrm{C}}, \boldsymbol{\epsilon}_{t_1}^{\mathrm{C}}, \boldsymbol{\epsilon}_{t_2}^{\mathrm{C}})$ is the dual basis to $(\mathbf{e}_n^{\mathrm{C}}, \mathbf{e}_{t_1}^{\mathrm{C}}, \mathbf{e}_{t_2}^{\mathrm{C}})$. The formulation of contact laws on velocity level is important for the discretization of the equations of motion of a multi-body system in chapter 8.

The relative velocity ${}_{\mathrm{C}}\boldsymbol{\gamma}(\mathbf{q}, \mathbf{u}) = [\gamma_N, \gamma_{T_1}, \gamma_{T_2}]^\top$ can be computed with the help of the right-hand side of (4.10) by omitting the scaling parameter s as

$$\begin{aligned}
{}_{\mathrm{C}}\boldsymbol{\gamma}(\mathbf{q}, \mathbf{u}) = {}_{\mathrm{C}}\dot{\mathbf{g}} &= \mathbf{A}_{\mathrm{CI}} ({}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{rig},C_b} - {}_{\mathrm{I}}\dot{\boldsymbol{\xi}}_{\mathrm{rig},C_a}) && (7.18)\\
&= \mathbf{A}_{\mathrm{CI}} \begin{bmatrix} \mathbf{I} & -\mathbf{R}_{\mathrm{K}_b\mathrm{I}} \, \mathbf{A}_{\mathrm{K}_b\mathrm{I}} \, {}_{\mathrm{I}}\tilde{\boldsymbol{\rho}}_{t,C_b} \mathbf{A}_{\mathrm{K}_b\mathrm{I}}{}^\top \end{bmatrix} \mathbf{u}_b && (7.19)\\
&\quad - \mathbf{A}_{\mathrm{CI}} \begin{bmatrix} \mathbf{I} & -\mathbf{R}_{\mathrm{K}_a\mathrm{I}} \, \mathbf{A}_{\mathrm{K}_a\mathrm{I}} \, {}_{\mathrm{I}}\tilde{\boldsymbol{\rho}}_{t,C_a} \mathbf{A}_{\mathrm{K}_a\mathrm{I}}{}^\top \end{bmatrix} \mathbf{u}_a && (7.20)\\
&= \mathbf{A}_{\mathrm{CI}} \begin{bmatrix} \mathbf{I} & -{}_{\mathrm{I}}\tilde{\boldsymbol{\rho}}_{t,C_b} \mathbf{A}_{\mathrm{K}_b\mathrm{I}}{}^\top \end{bmatrix} \mathbf{u}_b - \mathbf{A}_{\mathrm{CI}} \begin{bmatrix} \mathbf{I} & -{}_{\mathrm{I}}\tilde{\boldsymbol{\rho}}_{t,C_a} \mathbf{A}_{\mathrm{K}_a\mathrm{I}}{}^\top \end{bmatrix} \mathbf{u}_a && (7.21)\\
&= \hat{\mathbf{W}}_{C,B_b}{}^\top \mathbf{u}_b - \hat{\mathbf{W}}_{C,B_a}{}^\top \mathbf{u}_a \ . && (7.22)
\end{aligned}$$

7.1.1 Unilateral Contact

The impenetrability condition for the two bodies B_a and B_b in normal direction $\mathbf{e}_n^{\mathrm{C}}$ in figure 7.1 can be expressed by the complementarity condition on displacement level as

$$g_N \geqslant 0 \quad \wedge \quad \lambda_N \geqslant 0 \quad \wedge \quad \langle g_N \,|\, \lambda_N \rangle = 0 \quad \Leftrightarrow \quad 0 \leqslant g_N \perp \lambda_N \geqslant 0 \tag{7.23}$$

or equivalently by a normal cone inclusion given as

$$\boxed{\mathcal{R}_N = \{(g_N, \lambda_N) \in \mathbb{R} \times \mathbb{R}^* \mid -g_N \in \mathcal{N}_{\mathcal{C}_N}(\lambda_N)\}, \quad \mathcal{C}_N \coloneqq \mathbb{R}_0^+ \ . \quad \text{(displ. level)}} \tag{7.24}$$

The notation $\perp$ in (7.23) represents the annihilation by the duality pairing. In the field of continuum mechanics, especially in linear elasticity, the complementarity condition (7.23) is also known as Signorini's problem (cf. [86]). The graph $\mathcal{R}_N$ describing the constitutive force law of the unilateral contact is visualized in figure 7.2a. The normal cone inclusion in (7.24) is simply the subdifferential $\partial I_{\mathcal{C}_N}(\lambda_N)$ of the convex indicator function $I_{\mathcal{C}_N}$ at the point λ_N. By the dual relationship of convex functions given by the conjugation in definition 6.4, the indicator function $I_{\mathcal{C}_N}$ can be viewed as the dual of $(I_{\mathcal{C}_N})^*$, which is the support function of the set $\mathcal{C}_N$. Since $\mathcal{C}_N$ is a pointed convex cone, it follows by relation (6.9) that $(I_{\mathcal{C}_N})^* = I_{\mathcal{C}_N^\circ} = I_{\mathbb{R}_0^-}$ and applying the subdifferential yields

$$\lambda_N \in \partial I_{\mathbb{R}_0^-}(-g_N) = \mathcal{N}_{\mathbb{R}_0^-}(-g_N) \qquad \Leftrightarrow \qquad -\lambda_N \in \mathcal{N}_{\mathbb{R}_0^+}(g_N) \ , \tag{7.25}$$

where $\mathcal{N}_{\mathbb{R}_0^+} = \partial I_{\mathbb{R}_0^+}$ corresponds to $-F$ in (7.11). We could have written the unilateral contact in (7.24) also as

$$g_N \in \mathcal{N}_{\mathbb{R}_0^-}(-\lambda_N) \ , \tag{7.26}$$

which is more consistent with the concept of deriving force laws from *potential* and *complementary potential* energies. The indicator function $I_{\mathbb{R}_0^+}$ can be viewed as the *potential energy* and its convex conjugate $I_{\mathbb{R}_0^-}$ as the *complementary potential energy*. The opposite formulation in (7.24) is natural as the convex set $\mathcal{C}_N$ can directly be seen as the force reservoir for the normal force λ_N and the notation becomes simpler when combining the unilateral contact (7.24) with other set-valued force laws. The formulation of the unilateral contact on velocity level is necessary to be able to combine it with the formulation of the Coulomb friction discussed in the following. The formulation on velocity level is discussed in [60] and with $\gamma_N \coloneqq \dot{g}_N$ given as

$$\mathcal{S}_N(g_N) = \left\{ (\gamma_N, \lambda_N) \;\middle|\; \begin{array}{ll} -\gamma_N \in \mathcal{N}_{\mathbb{R}_0^+}(\lambda_N) & \text{if } g_N = 0 \\ \gamma_N \in \mathbb{R},\ \lambda_N = 0 & \text{if } g_N > 0 \end{array} \right\} . \qquad \text{(velocity level)} \tag{7.27}$$

In a numerical context, a unilateral contact is not considered if the normal gap distance g_N is positive and thus we simply write $\mathcal{S}_N$ without the dependence on g_N. Also, the gap function g_N is conveniently replaced by $g_N \leqslant 0$ in a discretized framework as discussed in chapter 8.

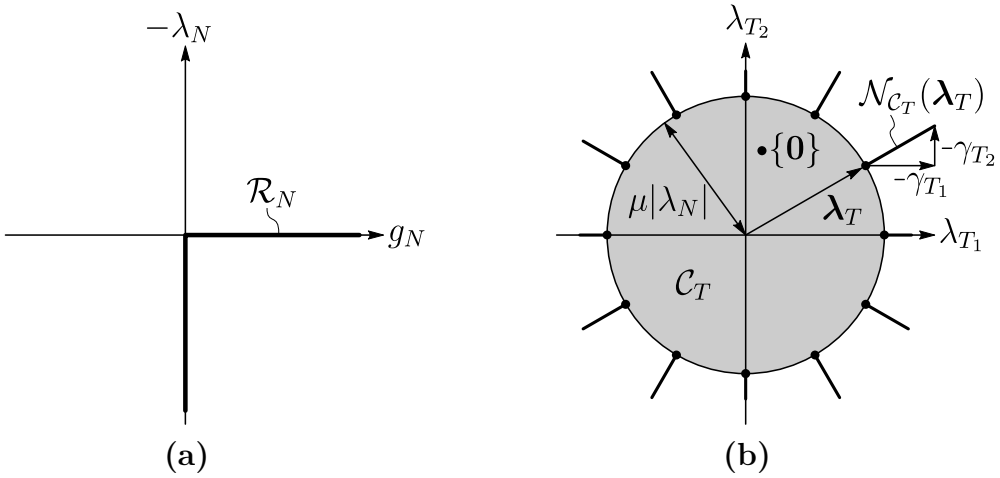


Figure 7.2: Two set-valued force laws. (a): constitutive force law for the unilateral contact in (7.24). (b): the two-dimensional Coulomb friction in (7.28).

7.1.2 Spatial Coulomb Friction

Coulomb friction is the dissipative effect of sticking and sliding of two surfaces relative to each other. The relative movement of the two contact points C_a and C_b in the common tangent plane $\mathcal{T}_C$ is described by the relative tangential contact velocity $\boldsymbol{\gamma}_T \in \mathcal{T}_C \subset \mathbb{E}^3$.

Coulomb friction relates the two-dimensional friction force $\boldsymbol{\lambda}_T \in \mathcal{T}_C^* \subset \mathbb{E}^{3*}$ to the two-dimensional tangential relative velocity $\boldsymbol{\gamma}_T$. The two surfaces slide relative to each other in the tangential plane $\mathcal{T}_C$ if the magnitude of $\boldsymbol{\gamma}_T$ is positive and the friction force $\boldsymbol{\lambda}_T$ points in the opposite direction of $\boldsymbol{\gamma}_T$. In the case of sliding, its magnitude is constant and proportional to the normal force by the friction coefficient $\mu \geqslant 0$, that is, $\|\boldsymbol{\lambda}_T\|_2 = \mu\lambda_N$. In the case of sticking, the relative velocity $\boldsymbol{\gamma}_T$ is zero and the magnitude of the friction force is $\|\boldsymbol{\lambda}_T\|_2 \leqslant \mu\lambda_N$. The graph $\mathcal{S}_T$ of the binary relation for the constitutive force law on velocity level which describes this sticking and sliding behavior can be formulated by a normal cone inclusion as

$$\mathcal{S}_T \coloneqq \left\{(\boldsymbol{\gamma}_T, \boldsymbol{\lambda}_T) \mid -\boldsymbol{\gamma}_T \in \mathcal{N}_{\mathcal{C}_T(\mu\lambda_N)}(\boldsymbol{\lambda}_T), \quad \mathcal{C}_T(r) \coloneqq r\mathcal{B}_2\right\}, \quad \text{(velocity level)} \tag{7.28}$$

where the convex set $\mathcal{C}_T(\mu\lambda_N)$ is the force reservoir of the tangential friction force $\boldsymbol{\lambda}_T$. The closed two-dimensional unit ball is denoted by $\mathcal{B}_2 \coloneqq \{\mathbf{x} \in \mathbb{E}^2 \mid \|\mathbf{x}\|_2 \leqslant 1\}$.

The normal cone inclusion in (7.28) is depicted in 7.2b. Strictly speaking, the normal cone in (7.28) lives in the primal space, that is, the space of relative velocities, and drawing the cone in the dual space in figure 7.2b is achieved by the choosing the standard Eucledian metric $\mathbf{I}$ (cf. section D.9.2). Note that, if $\boldsymbol{\lambda}_T$ is inside of $\mathcal{C}_T(\mu\lambda_N)$, the normal cone evaluates to the zero element, that is, $-\boldsymbol{\gamma}_T \in \{\mathbf{0}\}$, which describes the sticking state.

The formulation in (7.28) describes an isotropic Coulomb friction law because every tangential direction has the same maximal force range $[-\mu\lambda_N, \mu\lambda_N]$. In the interior of this range, the contact is in the sticking state. The convex conjugate of $I_{\mathcal{C}_T}$ is the support function of the set $\mathcal{C}_T$ which can be evaluated to $(I_{\mathcal{C}_T})^*(\boldsymbol{\gamma}_T) = \mu\lambda_N \|\boldsymbol{\gamma}_T\|_2$ with the help of definition 6.4. The support function is called *quasi*-potential in [6] because of its dependence on the normal force λ_N. The Coulomb friction in (7.28) fulfills the *principle of maximal dissipation* which states that the friction force $\boldsymbol{\lambda}_T$ maximizes the dissipation $D_{\boldsymbol{\gamma}_T}(\boldsymbol{\lambda}_T) = -\langle \boldsymbol{\lambda}_T \,|\, \boldsymbol{\gamma}_T \rangle$ for every relative velocity $\boldsymbol{\gamma}_T$. Indeed, a minimizer of the optimization problem

$$\begin{aligned} &\min_{\boldsymbol{\lambda}_T} -D_{\boldsymbol{\gamma}_T}(\boldsymbol{\lambda}_T) \\ &\textit{subject to} \quad \boldsymbol{\lambda}_T \in \mathcal{C}_T(\mu\lambda_N) \end{aligned} \tag{7.29}$$

fulfills the normal cone inclusion in (7.28). This can be checked by rewriting the constraint optimization problem into a unconstrained problem with an additional indicator function $I_{\mathcal{C}_T}(\boldsymbol{\lambda}_T)$ and applying the *first order optimality theorem* in 6.1. On the other hand, a constitutive force law described by maximizing the dissipation $D_{\boldsymbol{\gamma}}(\boldsymbol{\lambda}) \coloneqq \langle \boldsymbol{\lambda} \,|\, \boldsymbol{\gamma} \rangle$ modeled as the duality pairing between force $\boldsymbol{\lambda}$ and relative velocity $\boldsymbol{\gamma}$ together with a closed convex set $\mathcal{C}$ as the force reservoir directly implies a normal cone inclusion force law. The *principle of maximal dissipation* also applies, for example, for the non-isotropic Coulomb friction law. Other isotropic and non-isotropic friction laws, such as the Coulomb-Contensou formulation can be found in [108, 94, 95].

In the following, we discuss four different formulations on velocity level for the combination of the unilateral contact (7.27) with the two-dimensional Coulomb friction (7.28).

7.1.3 Formulations of Unilateral Contact with Coulomb Friction

A closed unilateral contact with Coulomb friction at a contact point C between two bodies B_a and B_b as depicted in figure 7.1. In the following, four different formulations on velocity level for a unilateral contact with Coulomb friction are presented.

Non-Linear Complementarity Formulation (NCF)

The unilateral Coulomb frictional contact can be formulated with a complementarity in the normal direction, a nonlinear complementarity in the tangential direction and an additional anti-collinearity constraint as

$$\begin{aligned} 0 \leqslant \gamma_N \quad &\perp \quad \lambda_N \geqslant 0, && \text{(unilateral contact)} \\ 0 \leqslant \|\boldsymbol{\gamma}_T\|_2 \quad &\perp \quad \mu\lambda_N - \|\boldsymbol{\lambda}_T\|_2 \geqslant 0 \ , && \text{(sticking and sliding)} \\ \|\boldsymbol{\lambda}_T\|_2 \, \boldsymbol{\gamma}_T + \|\boldsymbol{\gamma}_T\|_2 \, \boldsymbol{\lambda}_T &= \mathbf{0} \ . && \text{(anti-collinearity)} \end{aligned} \tag{7.30}$$

By substituting $s \coloneqq \|\boldsymbol{\gamma}_T\|_2$ and $t \coloneqq \mu\lambda_N - \|\boldsymbol{\lambda}_T\|_2$, the above problem can be formulated as two complementarity conditions with three nonlinear equations.

Separated Normal Cone Formulation (SNCF)

The separated normal cone formulation was first introduced by Alart & Curnier in [6] and directly follows from combining (7.24) and (7.28) as

$$\mathcal{S}_C = \left\{ (\boldsymbol{\gamma}, \boldsymbol{\lambda}) \;\middle|\; \begin{aligned} -\gamma_N &\in \mathcal{N}_{\mathcal{C}_N}(\lambda_N), & \mathcal{C}_N &\coloneqq \mathbb{R}_0^+ \\ -\boldsymbol{\gamma}_T &\in \mathcal{N}_{\mathcal{C}_T(\mu\lambda_N)}(\boldsymbol{\lambda}_T), & \mathcal{C}_T(r) &\coloneqq \mathcal{B}_2 \cdot r \end{aligned} \right\} \ . \qquad \text{(velocity level)} \tag{7.31}$$

Note that the normal cone inclusion for the normal and tangential direction depend asymmetrically on each other, that is, only the tangential direction depends on the normal direction. The dependence on λ_N, as required by the Coulomb friction, is the reason for the intricacies arising when trying to solve contact problems with Coulomb friction. The dependence on λ_N leads to a quasi-optimization problem. Depending on the mechanical problem, the existance of a solution compared to the convex case with $\mu = 0$ is no more guaranteed. The inclusion problem for unilateral contacts with Coulomb friction is further explained in section 7.2.1.

Unified Normal Cone Formulation (UNCF)

The unilateral frictional contact can also be expressed by a single normal cone inclusion to the closed convex friction cone $\mathcal{K}_\mu$ given by

$$\mathcal{K}_\mu \coloneqq \left\{ \boldsymbol{\lambda} \in \mathbb{E}^{3*} \mid \|\boldsymbol{\lambda}_T\|_2 \leqslant \mu\lambda_N, \ \lambda_N \geqslant 0 \right\} \ . \tag{7.32}$$

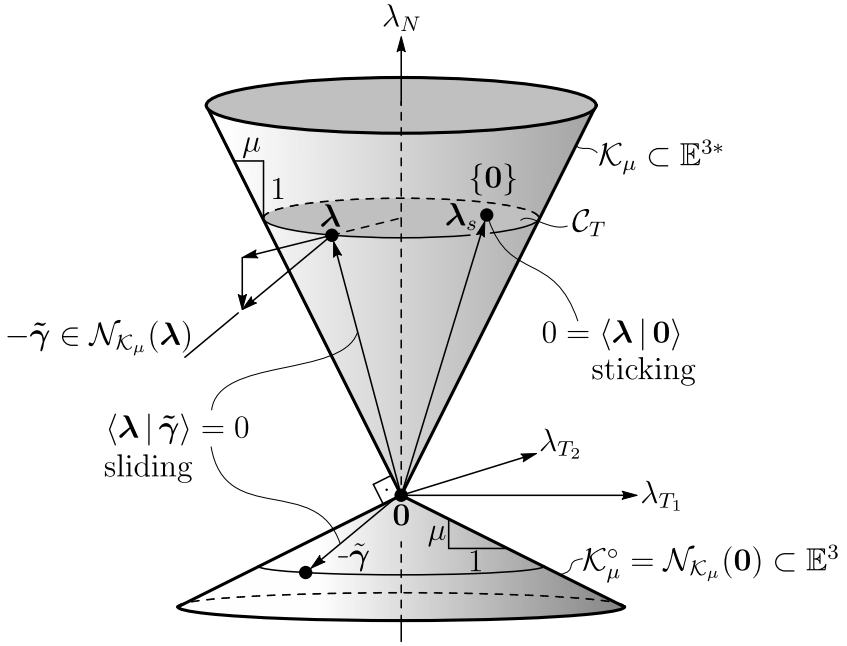


Figure 7.3: Visualization of De Saxé's unified normal cone formulation in (7.33) and the cone complementary condition in (7.38). The Coulomb friction cone $\mathcal{K}_\mu$ is polar to the cone $\mathcal{K}_\mu^\circ$ which is the space of all negative perturbed velocities $-\tilde{\boldsymbol{\gamma}}$. The normal cone at $\boldsymbol{\lambda}_s$ is $\{\mathbf{0}\}$ which is the only feasible element to fulfill the cone complementarity condition $\langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}} \rangle$ and is an example for sticking.

Using the friction cone $\mathcal{K}_\mu$, the unilateral Coulomb frictional contact in combined form can be written as

$$\mathcal{S}_C = \left\{ (\boldsymbol{\gamma}, \boldsymbol{\lambda}) \mid -\tilde{\boldsymbol{\gamma}} \in \mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda}), \quad \tilde{\boldsymbol{\gamma}} := \boldsymbol{\gamma} + \mu \left\| \boldsymbol{\gamma}_T \right\|_2 \mathbf{e}_n^{\mathrm{C}} \right\} . \qquad \text{(velocity level)} \tag{7.33}$$

This formulation is described in the extensive work of De Saxé [166]. If the normal cone is evaluated for the sliding case, that is, $\boldsymbol{\lambda}$ is at the boundary of the friction cone $\mathcal{K}_\mu$, exactly the correction term $\mu \left\| \boldsymbol{\gamma}_T \right\|_2 \mathbf{e}_n$ needs to be added to the left side of the inclusion in (7.33) to be in accordance with the unilateral contact in normal direction. If the correction term is neglected, that is, if $\tilde{\boldsymbol{\gamma}}$ is replaced by $\boldsymbol{\gamma}$ in (7.33), then the resulting normal cone at the boundary of $\mathcal{K}_\mu$ would lead to an incorrect positive relative velocity (take off) in normal direction. The unified normal cone formulation is depicted in figure 7.3. As shown in [166], there does not exist a complementary potential $\pi(\boldsymbol{\lambda})$ such that $-\boldsymbol{\gamma} \in \partial\pi(\boldsymbol{\lambda})$ describes the Coulomb friction because the set-valued mapping defined by the two normal cone inclusions in (7.31) is not *cyclically monotonic* (cf. [166] and [161]). However, there exists a bipotential $\pi_\mu(\mathbf{x}, \boldsymbol{\alpha})$ given as

$$\pi_\mu(\mathbf{x}, \boldsymbol{\alpha}) := I_{\mathbb{R}_0^-}(x_N) + I_{\mathcal{K}_\mu}(\boldsymbol{\alpha}) + \mu \alpha_N \left\| \mathbf{x}_T \right\|_2 \tag{7.34}$$

such that both inclusions

$$-\boldsymbol{\gamma} \in \partial_2 \pi_\mu(-\boldsymbol{\gamma}, \boldsymbol{\lambda}) = \mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda}) - \mu \lambda_N \left\| \boldsymbol{\lambda}_T \right\|_2 \tag{7.35}$$

$$\Updownarrow \tag{7.36}$$

$$\boldsymbol{\lambda} \in \partial_1 \pi_\mu(-\boldsymbol{\gamma}, \boldsymbol{\lambda}) = \begin{bmatrix} \mathcal{N}_{\mathbb{R}_0^-}(-\gamma_N) \\ \partial(\mu \lambda_N \left\| \boldsymbol{\lambda}_T \right\|_2) \end{bmatrix} \tag{7.37}$$

describe the unilateral contact with Coulomb friction. The bipotential $\pi_\mu(\mathbf{x}, \boldsymbol{\alpha})$ is only biconvex, which means that it is convex in the first argument if the second argument is fixed and vice versa. The bipotential (7.34) is not a convex function due to the non-convex term $\mu \alpha_N \left\| \mathbf{x}_T \right\|_2$.

Cone Complementarity Formulation (CCF)

The formulation in (7.33) can also be stated in a (second order) cone complementarity formulation between the convex friction cone $\mathcal{K}_\mu \subset \mathbb{E}^{3*}$ and its convex polar cone $\mathcal{K}_\mu^\circ \subset \mathbb{E}^{3**}$ which is isomorphic to $\{\mathbf{y} \in \mathbb{E}^3 \mid \langle \boldsymbol{\lambda} \,|\, \mathbf{y} \rangle \leqslant 0, \quad \forall \boldsymbol{\lambda} \in \mathcal{K}_\mu\}$ by the *reflexivity*[1] of $\mathbb{E}^3$.

The cone complementarity formulation is given as

$$\mathcal{S}_C = \left\{ (\boldsymbol{\gamma}, \boldsymbol{\lambda}) \;\middle|\; \begin{array}{l} -\tilde{\boldsymbol{\gamma}} \in \mathcal{K}_\mu^\circ, \quad \boldsymbol{\lambda} \in \mathcal{K}_\mu, \quad \langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}} \rangle = 0 \\ \tilde{\boldsymbol{\gamma}} \coloneqq \boldsymbol{\gamma} + \mu \left\| \boldsymbol{\gamma}_T \right\|_2 \mathbf{e}_n^{\mathrm{C}} \end{array} \right\} . \qquad \text{(velocity level)} \tag{7.38}$$

In the literature, equation (7.38) is also written as $\mathcal{K}_\mu^\circ \ni -\tilde{\boldsymbol{\gamma}} \perp \boldsymbol{\lambda} \in \mathcal{K}_\mu$, where the notation $\perp$ denotes the annihilation of the two vectors by the duality pairing. The equivalence of the cone complementarity formulation and the unified normal cone formulation is shown in appendix A.2. The cone complementary formulation with the duality pairing is visualized in figure 7.3.

The formulation of set-valued force laws as normal cone inclusions is the main approach used to simulate large-scale multi-body systems in this work. With the relations in (6.24), any normal cone inclusion to a convex set $\mathcal{C}$ can be rewritten as an implicit equation with a projection onto the convex set $\mathcal{C}$. In the next chapter, the general inclusion problem is introduced which is the combination of the equations of motion together with set-valued force laws of normal cone type. The case of only Coulomb frictional contacts formulated with the unified normal cone formulation in (7.33) is given at the end of the next chapter and allows to see that a corresponding optimization problem does not exist on acceleration level.

[1] A space V is reflexive if its natural embedding into the bidual space V^{**} is an isomorphism.

7.2 The Inclusion Problem

The equations of motion for a mechanical system without impacts together with k set-valued force laws of normal cone type on velocity level can be written as

$$\begin{aligned} \mathbf{M}(\mathbf{q},t)\dot{\mathbf{u}} - \mathbf{h}(\mathbf{q},\mathbf{u},t) - \mathbf{W}(\mathbf{q},t)\boldsymbol{\lambda} &= \mathbf{0} \ , && \text{(equation of motion, cf. (5.63))} \\ \dot{\mathbf{q}} &= \mathbf{F}(\mathbf{q},t)\mathbf{u} + \boldsymbol{\beta}(\mathbf{q},t) \ , && \text{(kinematic equation, cf. (4.13))} \end{aligned} \tag{7.39}$$

$$\left.\begin{aligned} \boldsymbol{\gamma}_i &= \mathbf{W}_i(\mathbf{q},t)\mathbf{u} + \boldsymbol{\chi}_i(\mathbf{q},t) \ , \\ \boldsymbol{\gamma}_i, \boldsymbol{\lambda}_i &\in \mathcal{S}_i \end{aligned}\right\} \quad \forall i \in [1\,;\,k] \ , \quad \begin{aligned} &\text{(relative velocity)} \\ &\text{(force law of normal cone type)} \end{aligned} \tag{7.40}$$

$$\mathcal{S}_i \coloneqq \{(\boldsymbol{\gamma}_i, \boldsymbol{\lambda}_i) \mid -\boldsymbol{\gamma}_i \in \mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i),\ \mathcal{C}_i \coloneqq \mathcal{C}_{i,1} \times \cdots \times \mathcal{C}_{i,l}\} \ . \tag{7.41}$$

Each set-valued force law with set $\mathcal{S}_i$ may contain a Cartesian combination of k normal cone inclusions. The reason for this notation is that each force law i represents an entity, for example a unilateral contact with Coulomb friction consisting of two normal cone inclusion as given in (7.31). Consequently, the convex sets $\mathcal{C}_i$ might also depend on $\boldsymbol{\lambda}$ in general. The reader should note that for simplicity reasons, the formulation (7.40) excludes normal cone inclusions which use some perturbed relative velocity $\tilde{\boldsymbol{\gamma}}$ as for De Saxé's unified normal cone inclusion in (7.33) and also excludes the damped versions of the unilateral constraint with Coulomb friction. For simplicity, we assume that all k set-valued force laws are potentially active on acceleration level such that the resulting force $\boldsymbol{\lambda}$ cannot be determined by a given state $(\mathbf{q},\mathbf{u})$ and they cannot be incorporated as *impressed* forces in the nonlinear term $\mathbf{h}(\mathbf{q},\mathbf{u},t)$.

To combine all k set-valued force laws with the equation of motion, each force law has to be stated in terms of accelerations instead of velocities or displacements. This step involves a lot of intricacies and is much more difficult for the Coulomb friction introduced in (7.28) as compared to a simple unilateral contact given in (7.27). In the following, we give a mathematical non-rigorous motivation to obtain a formulation of a set-valued force law of normal cone type on acceleration level.

The time-dependent vector $-\boldsymbol{\gamma}_i$ stays in the closed convex cone $\mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i)$ if the absolute time derivative $-\dot{\boldsymbol{\gamma}}_i$ belongs at every instance of time to the tangent cone of $\mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i)$ at $\boldsymbol{\gamma}_i$, that is, $-\dot{\boldsymbol{\gamma}}_i \in \mathcal{T}_{\mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i)}(\boldsymbol{\gamma}_i)\ \forall t$ (cf. in [60, sec. 7.2] and in [96, p. 75]). For a definition of the tangent cone the reader is referred to [61]. In this way, it is ensured that $\boldsymbol{\gamma}_i$ stays in the set $\mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda})$ over time. The tangent cone $\mathcal{T}_{\mathcal{K}}(\mathbf{x})$ of a closed convex cone $\mathcal{K}$ at $\mathbf{x} \in \mathcal{K}$ is a superset of the closed convex cone $\mathcal{K}$ itself, that is, $\mathcal{T}_{\mathcal{K}}(\mathbf{x}) \supset \mathcal{K}$. This reasoning motivates the requirement $-\dot{\boldsymbol{\gamma}}_i \in \mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i)$ which is more restrictive and still ensures that $\boldsymbol{\gamma}$ stays in the set $\mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}_i)$ for all times. This reasoning could also be applied to transfer normal cone inclusions on displacement level to normal cone inclusions on acceleration level. Stating every force law in (7.40) on acceleration level yields

$$\left.\begin{aligned} \dot{\boldsymbol{\gamma}}_i &= \mathbf{W}_i(\mathbf{q},t)^{\top}\,\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}}_i(\mathbf{q},\mathbf{u},t) \ , \\ -\dot{\boldsymbol{\gamma}}_i &\in \mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda}) \end{aligned}\right\} \quad \forall i \in [1\,;\,k] \qquad \Leftrightarrow \qquad \begin{aligned} \dot{\boldsymbol{\gamma}} &= \mathbf{W}^{\top}\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}} \ , \\ -\dot{\boldsymbol{\gamma}} &\in \mathcal{N}_{\mathcal{C}}(\boldsymbol{\lambda}) \ , \end{aligned} \tag{7.42}$$

where $\mathcal{C} \coloneqq \mathcal{C}_1 \times \cdots \times \mathcal{C}_k$ is the Cartesian product of all convex sets $\mathcal{C}_i$ and all individual terms have been assembled such that $\mathbf{W} \coloneqq [\mathbf{W}_1, \ldots, \mathbf{W}_k]$ denotes the generalized force

direction of the force $\boldsymbol{\lambda} \coloneqq [\boldsymbol{\lambda}_1{}^\top, \dots, \boldsymbol{\lambda}_k{}^\top]^\top$ corresponding to the relative velocity $\dot{\boldsymbol{\gamma}} \coloneqq [\dot{\boldsymbol{\gamma}}_1{}^\top, \dots, \dot{\boldsymbol{\gamma}}_k{}^\top]^\top$.
Inserting the equation of motion (7.39) into (7.42) yields

$$\boxed{\begin{aligned} &-(\mathbf{G}\boldsymbol{\lambda} + \mathbf{c}) \in \mathcal{N}_{\mathcal{C}}(\boldsymbol{\lambda}) \\ \text{with:}\quad &\begin{bmatrix} \mathbf{G}(\mathbf{q}, t) \coloneqq \mathbf{W}^\top \mathbf{M}^{-1} \mathbf{W} \ , \quad \mathbf{c}(\mathbf{q}, \mathbf{u}, t) \coloneqq \mathbf{W}^\top \mathbf{M}^{-1} \mathbf{h} + \hat{\boldsymbol{\chi}} \\ \dot{\boldsymbol{\gamma}} = \mathbf{G}\boldsymbol{\lambda} + \mathbf{c} \ . \end{bmatrix} \end{aligned}} \tag{7.43}$$

Equation (7.43) is the *inclusion problem* of the force $\boldsymbol{\lambda}$ for a given displacement $\mathbf{q}$ and velocity $\mathbf{u}$. The matrix $\mathbf{G}$ is the symmetric positive semi-definite Delassus operator and defines the coupling between all contact forces, which is called *contact graph*. The inclusion problem can be converted directly to an implicit proximal equation by (6.24) which yields

$$\boxed{\boldsymbol{\lambda} = \operatorname{prox}_{\mathcal{C}}^{R}\Big(\boldsymbol{\lambda} - \mathbf{R}^{-1}(\mathbf{G}\boldsymbol{\lambda} + \mathbf{c})\Big) \ .} \tag{7.44}$$

If the set $\mathcal{C}$ does not depend on $\boldsymbol{\lambda}$, the inclusion problem in (7.43) can be interpreted as a necessary condition for a minimizer $\boldsymbol{\lambda}^*$ of the convex quadratic optimization problem

$$\min_{\boldsymbol{\lambda}} \frac{1}{2}\boldsymbol{\lambda}^\top \mathbf{G}\boldsymbol{\lambda} + \mathbf{c}^\top \boldsymbol{\lambda} + I_{\mathcal{C}}(\boldsymbol{\lambda}) \quad \Leftrightarrow \quad \min_{\boldsymbol{\lambda} \in \mathcal{C}} \frac{1}{2}\boldsymbol{\lambda}^\top \mathbf{G}\boldsymbol{\lambda} + \mathbf{c}^\top \boldsymbol{\lambda} \tag{7.45}$$

for some given displacement $\mathbf{q}$ and velocity $\mathbf{u}$. The reader should note that (7.45) is not valid for unilateral contacts with Coulomb friction in (7.31) since the convex set $\mathcal{C}_{T,i}$ for each contact i depends on its normal force $\boldsymbol{\lambda}_{N,i}$ and (7.43) is *not* the optimality condition of (7.45). In the next section, the discussion of the case for frictional contacts with the formulation of De Saxé in (7.33) will provide some more insight why no optimization problem exists on acceleration level.

If the set $\mathcal{C}$ does not depend on $\boldsymbol{\lambda}$, the primal convex optimization problem in the variable $\dot{\mathbf{u}}$ can be obtained by conjugating all normal cone inclusions in (7.42) which yields the subdifferential of the support function (6.7) as

$$\boldsymbol{\lambda} \in \partial I_{\mathcal{C}}^*(-\dot{\boldsymbol{\gamma}}) = \partial I_{\mathcal{C}}^*\big(-(\mathbf{W}\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}})\big) \quad \Leftrightarrow \quad \mathbf{f} \in -\mathbf{W}\partial I_{\mathcal{C}}^*\big(-(\mathbf{W}\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}})\big) \ . \tag{7.46}$$

The equation of motion in (7.39) can then be written as

$$\mathbf{M}(\mathbf{q}, t)\dot{\mathbf{u}} - \mathbf{h}(\mathbf{q}, \mathbf{u}, t) - \mathbf{W}\partial I_{\mathcal{C}}^*\big(-(\mathbf{W}\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}})\big) \ni \mathbf{0} \ . \tag{7.47}$$

The inclusion in (7.47) can be viewed as the optimality condition of the following convex unconstrained optimization problem for a given $\mathbf{q}$ and $\mathbf{u}$ at a time instant t:

$$\min_{\dot{\mathbf{u}}} \frac{1}{2}\dot{\mathbf{u}}^\top \mathbf{M}\dot{\mathbf{u}} + \mathbf{h}^\top \dot{\mathbf{u}} + I_{\mathcal{C}}^*\big(\dot{\boldsymbol{\gamma}}(\mathbf{q}, \mathbf{u}, \dot{\mathbf{u}}, t)\big) \ . \tag{7.48}$$

For more information on the dual and primal optimization problems in (7.45) and (7.48), the reader is referred to [60].

7.2.1 Inclusion Problem for Unilateral Contacts with Coulomb Friction

In the following, we investigate the case where all k set-valued force laws in (7.40) are unilateral contacts with Coulomb friction. All these contact laws are formulated using the unified normal cone formulation in (7.33), that is, $\mathcal{S}_i = \mathcal{S}_C \; \forall i \in [1\,;\,k]$. The set-valued force laws for k unilateral contacts with Coulomb friction can be expressed as

$$\left.\begin{aligned} \boldsymbol{\gamma}_i = \mathbf{W}_i{}^{\top}(\mathbf{q},t)\mathbf{u} + \boldsymbol{\chi}_i(\mathbf{q},t)\;, \\ -\tilde{\boldsymbol{\gamma}}_i \in \mathcal{N}_{\mathcal{K}_{\mu_i}}(\boldsymbol{\lambda}_i) \end{aligned}\right\} \quad \forall i \in \mathcal{I}(\mathbf{q},t)\;. \quad \begin{matrix} \text{(relative velocity for contact } i\text{)} \\ \text{(force law for contact } i\text{)} \end{matrix} \tag{7.49}$$

The set $\mathcal{I}(\mathbf{q},t) \coloneqq \{i \mid g_{N,i}(\mathbf{q},t) = 0\}$ consists of k indices for the contacts which are closed on displacement level. The shifted relative velocity is defined according to (7.33) as $\tilde{\boldsymbol{\gamma}}_i \coloneqq \boldsymbol{\gamma}_i + \mu_i \left\|\boldsymbol{\gamma}_{T,i}\right\|_2 \mathbf{e}_n^{\mathrm{C}_i}$.

All normal cone inclusions in (7.49) can be combined as

$$\tilde{\boldsymbol{\gamma}} = \boldsymbol{\gamma} + \mathbf{E}\;\boldsymbol{\mu}\;\mathbf{s}(\boldsymbol{\gamma}) \in \mathcal{N}_{\mathcal{K}}(\boldsymbol{\lambda})\;, \tag{7.50}$$

$$\mathbf{E} \coloneqq \begin{bmatrix} \mathbf{e}_n^{\mathrm{C}_1} & \mathbf{0} & \cdots \\ \mathbf{0} & \ddots & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{e}_n^{\mathrm{C}_k} \end{bmatrix} \in \mathbb{R}^{3k\times k}\;, \quad \begin{aligned} \boldsymbol{\mu} &\coloneqq \operatorname{diag}\left([\mu_1,\ldots,\mu_k]\right) \in \mathbb{R}^{k\times k}\;, \\ \mathbf{s}(\boldsymbol{\gamma}) &\coloneqq \left[\left\|\boldsymbol{\gamma}_{T,1}\right\|_2,\ldots,\left\|\boldsymbol{\gamma}_{T,k}\right\|_2\right]^{\top} \in \mathbb{R}^{k}\;, \\ \boldsymbol{\gamma} &\coloneqq \left[\boldsymbol{\gamma}_1{}^{\top},\ldots,\boldsymbol{\gamma}_k{}^{\top}\right]^{\top} \in \mathbb{R}^{3k}\;, \end{aligned} \tag{7.51}$$

where $\mathcal{K} \coloneqq \mathcal{K}_{\mu_1} \times \cdots \times \mathcal{K}_{\mu_k}$ is the Cartesian product of all friction cones.

Taking the derivative $\tilde{\boldsymbol{\gamma}}$ with respect to time, the force laws on acceleration level can be assembled as

$$\dot{\boldsymbol{\gamma}} = \mathbf{W}^{\top}\dot{\mathbf{u}} + \hat{\boldsymbol{\chi}}, \quad -\Big(\underbrace{\left[\mathbf{I} + \mathbf{E}\boldsymbol{\mu}\frac{\partial \mathbf{s}}{\partial \boldsymbol{\gamma}}\right]}_{\bar{\mathbf{E}}(\mathbf{q},\mathbf{u},t)}\dot{\boldsymbol{\gamma}} + \underbrace{\dot{\mathbf{E}}\boldsymbol{\mu}\mathbf{s}(\boldsymbol{\gamma})}_{\bar{\mathbf{s}}(\mathbf{q},\mathbf{u},t)}\Big) \in \mathcal{N}_{\mathcal{K}}(\boldsymbol{\lambda})\;. \tag{7.52}$$

The matrix $\bar{\mathbf{E}}(\mathbf{q},\boldsymbol{\gamma},t)$ is upper unitriangular and thus always invertible. Substituting $\dot{\boldsymbol{\gamma}}$ and $\dot{\mathbf{u}}$ in (7.39) into (7.52), results in the following normal cone inclusion:

$$\begin{aligned} &-\left(\bar{\mathbf{E}}\mathbf{G}\boldsymbol{\lambda} + \bar{\mathbf{E}}\mathbf{c} + \bar{\mathbf{s}}\right) \in \mathcal{N}_{\mathcal{K}}(\boldsymbol{\lambda}) \\ \text{with:}\quad &\begin{bmatrix} \mathbf{G}(\mathbf{q},t) \coloneqq \mathbf{W}^{\top}\mathbf{M}^{-1}\mathbf{W}\;, & \mathbf{c}(\mathbf{q},\mathbf{u},t) \coloneqq \mathbf{W}^{\top}\mathbf{M}^{-1}\mathbf{h} + \hat{\boldsymbol{\chi}}\;, \\ \bar{\mathbf{E}}(\mathbf{q},\mathbf{u},t) \coloneqq \mathbf{I} + \mathbf{E}\boldsymbol{\mu}\dfrac{\partial \mathbf{s}}{\partial \boldsymbol{\gamma}}\;, & \bar{\mathbf{s}}(\mathbf{q},\mathbf{u},t) \coloneqq \dot{\mathbf{E}}\boldsymbol{\mu}\mathbf{s}(\boldsymbol{\gamma})\;. \end{bmatrix} \end{aligned} \tag{7.53}$$

Equation (7.53) is the resulting inclusion problem for a given state $\mathbf{q}$ and $\mathbf{u}$ at a time instant t. Unfortunately, the left-hand side of (7.53) can not be interpreted anymore as the negative differential of a function $f(\boldsymbol{\lambda})$ simply because the non-singular matrix $\bar{\mathbf{E}}\mathbf{G}$ is non-symmetric for which does not exist an anti-derivative. This is directly reflected in the fact that there does not exist an optimization problem in the form of $\min_{\boldsymbol{\lambda}\in\mathcal{K}} f(\boldsymbol{\lambda})$

for unilateral contacts with Coulomb friction. Rewriting (7.53) with a proximal equation yields

$$\boldsymbol{\lambda} = \operatorname{prox}_{\mathcal{K}}^{R}\big(\boldsymbol{\lambda} - \mathbf{R}^{-1}\big(\bar{\mathbf{E}}\mathbf{G}\boldsymbol{\lambda} + \bar{\mathbf{E}}\mathbf{c} + \bar{\mathbf{s}}\big)\big) \ . \tag{7.54}$$

For the case that the equation of motion is discretized, as for example with Moreau's time-stepping scheme discussed in chapter 8, the inclusion problem for one time step can be written as a non-convex optimization problem for a given fixed $\mathbf{s}(\boldsymbol{\gamma})$ in (7.50) as shown in [33]. For the case of using the separate formulation in (7.31) for the set $\mathcal{C}_i$ in (7.40), that is,

$$\mathcal{C} \coloneqq \underbrace{\mathcal{C}_{N,i} \times \mathcal{C}_{T,1}(\mu_i \lambda_{N,1})}_{\mathcal{C}_1} \times \cdots \times \underbrace{\mathcal{C}_{N,k} \times \mathcal{C}_{T,k}(\mu_k \lambda_{N,k})}_{\mathcal{C}_k} \ , \tag{7.55}$$

one recognizes that the optimization problem (7.45) becomes convex in the normal direction if all tangential forces $\boldsymbol{\lambda}_{T,i}$ are known and convex in the tangential direction if all normal forces $\lambda_{N,i}$ are known. This *biconvexity* can be used to solve the inclusion problem by iteratively solving normal and tangential direction in an interleaved fashion.

7.3 Impacts

This short chapter deals with impacts in non-smooth mechanical systems and does not aim to be a complete treatise. We will rather give a brief summary based on the profound introduction in [61]. In reality, a collision or impact between two bodies of a mechanical system is a wave or energy propagation phenomenon which takes place over a short amount of time. The model used in rigid body dynamics to describe these collision phenomena is to condense the time interval of the collision dynamics into a single time instant, where the velocity eventually becomes discontinuous. The interaction between the collision partners producing this discontinuity is modeled by forces of Dirac-like character, namely impulsive forces.

In the following, we consider k unilateral constraints imposed on a mechanical system described by a Riemannian configuration manifold $\mathcal{M}$ with metric $\mathbf{M}(\mathbf{q}, t)$, precisely the mass matrix of the variational equation of motion in (5.63). In this way, the configuration manifold $\mathcal{M}$ is restricted to a non-empty set $\mathcal{C}$ given as

$$\mathcal{C} \coloneqq \{g_i(\mathbf{q}) \geqslant 0 \ \forall i \in [1\,;\,k]\} \ , \tag{7.56}$$

where we assume the gap distances g_i to be C^1-continuous and such that their level curves $g_i(\mathbf{q}) = 0$ intersect.

The constraints in (7.56) can be comprised, for example, of k unilateral contacts among rigid bodies in a multi-body system. The time evolution of a mechanical system is described by a continuous path $\mathbf{q}(\cdot)$ on the manifold $\mathcal{M}$ restricted to the set $\mathcal{C}$. The trajectory $\mathbf{q}(\cdot)$ is therefore required to remain in the set $\mathcal{C}$ and any crossing of the boundary of $\mathcal{C}$ is prohibited. This is ensured by allowing discontinuities in the velocities $\mathbf{u}$. Whenever the trajectory $\mathbf{q}(t^*)$ collides with the boundary of $\mathcal{C}$ at a certain time t^*, the velocity $\mathbf{u}$ of the mechanical system may jump and ensures that the resulting future time evolution of $\mathbf{q}$ either stays on the boundary or leaves the boundary towards the interior of $\mathcal{C}$. The event at which a discontinuity in $\mathbf{u}$ occurs is called *impact* and the pre-impact velocity and post-impact velocity at an impact time t^* are denoted as $\mathbf{u}^-(t^*)$ and $\mathbf{u}^+(t^*)$ and correspond to the left and right limit of $\mathbf{u}$ at t^*, respectively. A solution curve $\mathbf{q}(\cdot)$ on the submanifold $\mathcal{C}$ is visualized in figure 7.4a together with the tangent space $\mathcal{T}_{\mathbf{q}}\mathcal{M}$ at $\mathbf{q}(t^*)$. The classical approach [119, 120, 78, 78] to handle discontinuities in $\mathbf{u}$ is to introduce a measure-differential formulation of the equation of motion given as

$$\mathbf{M}(\mathbf{q}, t)\mathrm{d}\mathbf{u} - \mathbf{h}(\mathbf{q}, \mathbf{u}, t)\mathrm{d}t - \mathbf{W}\mathrm{d}\mathbf{P} = \mathbf{0} \ , \tag{7.57}$$

$$\dot{\mathbf{q}}\mathrm{d}t = \mathbf{F}(\mathbf{q}, t)\mathbf{u}\mathrm{d}t \ + \mathbf{b}(\mathbf{q}, t)\mathrm{d}t \ . \tag{7.58}$$

The velocities $\mathbf{u}$ and unilateral constraint forces $\boldsymbol{\lambda}$ have been replaced by measures $\mathrm{d}\mathbf{u}$ and $\mathrm{d}\mathbf{P}$, respectively. A measure is intrinsically connected to an integral and (7.57) it is understood as an *equation of measures* rather than a differential equation. The measures in (7.57) are comprised of a continuous part with Lebesque measure $\mathrm{d}t$ and an atomic part with atomic measure $\mathrm{d}\eta$ as

$$\mathrm{d}\mathbf{u} \coloneqq \dot{\mathbf{u}}\mathrm{d}t + (\mathbf{u}^+ - \mathbf{u}^-)\mathrm{d}\eta \ , \qquad \mathrm{d}\mathbf{P} \coloneqq \boldsymbol{\lambda}\mathrm{d}t + \underbrace{(\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-)}_{\boldsymbol{\Lambda}}\mathrm{d}\eta \ . \tag{7.59}$$

The density of the Lebesque measure $\mathrm{d}t$ is denoted by $\dot{\mathbf{u}}$ and $\boldsymbol{\lambda}$, respectively. The density with respect to the atomic measure $\mathrm{d}\eta$ in $\mathrm{d}\mathbf{u}$ and $\mathrm{d}\mathbf{P}$ consists of a velocity jump density $\mathbf{u}^+ - \mathbf{u}^-$ and a force jump density $\boldsymbol{\Lambda} \coloneqq \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$, respectively. The Lebesque measure $\mathrm{d}t$ and the atomic measure $\mathrm{d}\eta$ are singular. This means that if there is a velocity jump at a singleton $\{t^*\}$, that is, the set with a single time instant t^*, the integral of the Lebesque part $\dot{\mathbf{u}}\mathrm{d}t$ over this singleton vanishes and the integral of the atomic part $(\mathbf{u}^+ - \mathbf{u}^-)\mathrm{d}\eta$ measures the jump in $\mathbf{u}$, that is,

$$\int\limits_{\{t^*\}} \mathrm{d}\mathbf{u} = \int\limits_{\{t^*\}} (\mathbf{u}^+ - \mathbf{u}^-)\mathrm{d}\eta = \mathbf{u}^+(t^*) - \mathbf{u}^-(t^*) \neq \mathbf{0} \ . \tag{7.60}$$

Vice versa, the atomic measure vanishes in intervals where $\mathbf{u}$ is continuous. The measure $\mathrm{d}\mathbf{u}$, for example, can be integrated over a closed non-empty interval $[t_0, t_1] \subset \mathbb{R}, t_1 > t_0$ containing the singletons $\{t_0\}$ and $\{t_1\}$ which yields

$$\begin{aligned}
\int\limits_{[t_0,t_1]} \mathrm{d}\mathbf{u} &= \int\limits_{\{t_0\}} \mathrm{d}\mathbf{u} + \int\limits_{(t_0,t_1)} \mathrm{d}\mathbf{u} + \int\limits_{\{t_1\}} \mathrm{d}\mathbf{u} && (7.61)\\
&= \mathbf{u}^+(t_0) - \mathbf{u}^-(t_0) + \mathbf{u}^-(t_1) - \mathbf{u}^+(t_0) + \mathbf{u}^+(t_1) - \mathbf{u}^-(t_1) && (7.62)\\
&= \mathbf{u}^+(t_1) - \mathbf{u}^-(t_0) \ . && (7.63)
\end{aligned}$$

The integral in (7.61) is the limit of the Riemann-Stieltjes sum and $\mathrm{d}\mathbf{u}$ is the Lebesque-Stieltjes measure (cf. [96]). The equation of measures in (7.57) contain both the continuous and discontinuous motion of the mechanical system. If the equation of measures in (7.57) is integrated over a singleton $\{t\}$ (or equivalently evaluated with $\{t\}$), the *impact equation* is obtained, that is

$$\mathbf{M}(\mathbf{q}, t)\left(\mathbf{u}^+(t) - \mathbf{u}^-(t)\right) - \mathbf{W}(\mathbf{q}, t)\boldsymbol{\Lambda} = \mathbf{0}. \tag{7.64}$$

We assumed from the beginning that the nonlinear term $\mathbf{h}(\mathbf{q}, \mathbf{u}, t)$ consisting of Coriolis forces, gyroscopic forces and impressed forces, does not contain any contribution which can generate a discontinuity in $\mathbf{u}$. The impact equations above need to be completed with an impact law which is a relation between the pre- and post-impact velocity, $\mathbf{u}^-$ and $\mathbf{u}^+$, respectively, and the corresponding impulse $\boldsymbol{\Lambda}$ given by the graph $\mathcal{P}$, that is, $(\mathbf{u}^-, \mathbf{u}^+, \boldsymbol{\Lambda}) \in \mathcal{P}$.

7.3.1 Geometric Concepts

In the following, we briefly explain the geometric concepts of a *perfect* impact and its requirements on the graph $\mathcal{P}$ of the constitutive impact law. We consider in this discussion only impacts in normal direction and thus exclude friction. To simplify the discussion, we assume that $\mathbf{F}(\mathbf{q}, t) = \mathbf{I}$ and $\mathbf{b}(\mathbf{q}, t) = \mathbf{0}$ in (7.58) and that the mechanical system does not contain external excitations. Furthermore, we assume that at an impact configuration $\mathbf{q}(t^*)$ at time t^*, abbreviated as $\mathbf{q}$, several unilateral constraints in (7.56) are active and pre- and post-impact velocities at t^* are abbreviated as $\mathbf{u}^-$ and $\mathbf{u}^+$, respectively. The reader is referred to figure 7.4 for the remainder of this chapter.

We denote the differential of the gap distance $\mathbf{d}g_i(\mathbf{q})$ by $\mathbf{w}_i$ and the indices of the set of active constraints by $\mathcal{I}$, that is,

$$\mathbf{d}g_i(\mathbf{q}) \coloneqq \mathbf{w}_i \coloneqq \left[\frac{\partial g_i}{\partial \mathbf{q}}\right]^\top \quad \forall i \in \mathcal{I}(\mathbf{q}) \coloneqq \{i \mid g_i(\mathbf{q}) = 0 \; \forall i \in [1\,;\,k]\} \; . \tag{7.65}$$

The gradient of a function g_i can be defined similar to definition (6.7) as $\nabla g_i(\mathbf{q}) = \mathbf{M}^{-1}\mathbf{d}g_i(\mathbf{q})$. The normal cone $\mathcal{N}_\mathcal{C}(\mathbf{q})$ to the set $\mathcal{C}$ at $\mathbf{q}$ is then finitely generated by the negative differentials $\mathbf{d}g_i(\mathbf{q})$ as

$$\mathcal{N}_\mathcal{C}(\mathbf{q}) \coloneqq \Big\{\mathbf{f} \mid \mathbf{f} = \sum_{i\in\mathcal{I}(\mathbf{q})} -\mathbf{d}g_i(\mathbf{q})\Lambda_i \; , \quad \Lambda_i \geqslant 0\Big\} \subset T^*_\mathbf{q}\mathcal{M} \; . \tag{7.66}$$

The impact equation (7.64) for all active constraints becomes

$$\mathbf{u}^+ - \mathbf{u}^- = \sum_{i\in\mathcal{I}(\mathbf{q})} \nabla g_i(\mathbf{q})\Lambda_i \; , \qquad \Lambda_i \geqslant 0 \qquad \forall i \in \mathcal{I}(\mathbf{q}) \; . \tag{7.67}$$

By the isomorphism induced by the inner product $(\bullet \mid \bullet)_M$ with metric $\mathbf{M}(\mathbf{q},t)$ (cf. section D.9.2), the normal cone can be mapped to its associated primal cone in the tangent space $T_\mathbf{q}\mathcal{M}$, which is the orthogonal tangent cone $\mathcal{T}^\perp_\mathcal{C}(\mathbf{q}) = \mathbf{M}^{-1}\mathcal{N}_\mathcal{C}(\mathbf{q})$ to the set $\mathcal{C}$ at the point $\mathbf{q}$ given by

$$\mathcal{T}^\perp_\mathcal{C}(\mathbf{q}) \coloneqq \Big\{\mathbf{v} \mid \mathbf{v} = \sum_{i\in\mathcal{I}(\mathbf{q})} -\nabla g_i(\mathbf{q})\Lambda_i \; , \quad \Lambda_i \geqslant 0\Big\} \subset T_\mathbf{q}\mathcal{M} \; . \tag{7.68}$$

The tangent cone $\mathcal{T}_\mathcal{C}(\mathbf{q})$ to $\mathcal{C}$ at $\mathbf{q}$ can either be constructed by the *polar* cone to $\mathcal{N}_\mathcal{C}(\mathbf{q})$ analogously to definition 6.5 or by the orthogonal cone to $\mathcal{T}^\perp_\mathcal{C}(\mathbf{q})$ similar to (7.15). Both constructions lead to

$$\mathcal{T}_\mathcal{C}(\mathbf{q}) \coloneqq \Big\{\mathbf{u} \mid \langle \mathbf{d}g_i(\mathbf{q}) \,|\, \mathbf{u}\rangle = (\nabla g_i(\mathbf{q}) \,|\, \mathbf{u})_M \geqslant 0 \quad \forall i \in \mathcal{I}(\mathbf{q})\Big\} \subset T_\mathbf{q}\mathcal{M} \; , \tag{7.69}$$

where $\langle \bullet \mid \bullet \rangle$ denotes the duality pairing. The tangent cone $\mathcal{T}_\mathcal{C}$ and its orthogonal counterpart $\mathcal{T}^\perp_\mathcal{C}$ are visualized in figure 7.4.

A *perfect* impact needs to fulfill three requirements which leads to its geometric concept. First, the kinematics of all unilateral constraints need to be consistent. This means that the relative post impact velocity fulfills $\gamma_i^+ = \gamma_i(\mathbf{q},\mathbf{u}^+) \geqslant 0$ such that the unilateral constraint either becomes inactive or stays active in the future. The relative pre-impact velocity is assumed to be $\gamma_i^- = \gamma_i(\mathbf{q},\mathbf{u}^-) \leqslant 0$ which means that the mechanical system has arrived at $g_i(\mathbf{q}) = 0$ from an admissible state in the past. The second and third requirements are that the impact equation (7.67) is fulfilled and that the total energy, that is, the kinetic energy, does not increase over the impact time t^*. All these requirements can be summarized to the following restrictions for all $i \in \mathcal{I}(\mathbf{q})$:

$$\gamma_i(\mathbf{q},\mathbf{u}) = \langle \mathbf{d}g_i(\mathbf{q}) \,|\, \mathbf{u}\rangle = \mathbf{w}_i{}^\top\mathbf{u} \; , \qquad \gamma_i^+ \geqslant 0 \; , \; \gamma_i^- \leqslant 0 \; , \quad \text{(kinematic consistency)} \tag{7.70}$$

$$\mathbf{u}^+ - \mathbf{u}^- = \sum_{i\in\mathcal{I}} \nabla g_i(\mathbf{q})\Lambda_i \; , \qquad \Lambda_i \geqslant 0 \; , \quad \text{(kinetic consistency)} \tag{7.71}$$

$$T(\mathbf{u}) = \frac{1}{2}\,\|\mathbf{u}\|^2_M \qquad T(\mathbf{u}^+) \leqslant T(\mathbf{u}^-) \; . \quad \text{(energetic consistency)} \tag{7.72}$$

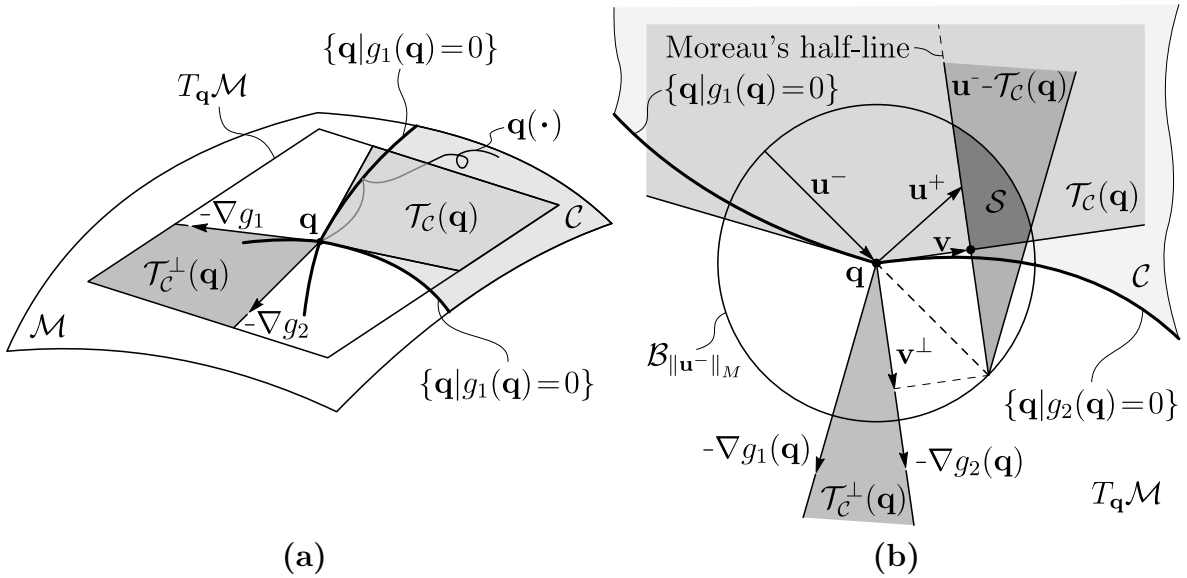


Figure 7.4: (a): Configuration manifold $\mathcal{M}$ with two unilateral constraints which restricts the trajectory $\mathbf{q}(\cdot)$ to the submanifold $\mathcal{C}$. (b): Geometric visualization of the requirements in (7.73) for an impact configuration at $\mathbf{q}$. The set of all admissible post-impact velocities is denoted as $\mathcal{S}$. The shown post-impact velocity $\mathbf{u}^+$ is given by Moreau's impact law with $\epsilon = 0.5$.

These restriction can also be formulated by using the tangent cone and its orthogonal counterpart defined above which yields

$$\begin{aligned}
&\mathbf{u}^+ \in \mathcal{T}_\mathcal{C}(\mathbf{q}) \ , \quad \mathbf{u}^- \in -\mathcal{T}_\mathcal{C}(\mathbf{q}) \ , && \text{(kinematic consistency)} \\
&\mathbf{u}^+ \in \mathbf{u}^- - \mathcal{T}_\mathcal{C}^\perp(\mathbf{q}) \ , && \text{(kinetic consistency)} \\
&\mathbf{u}^+ \in \mathcal{B}_{\|\mathbf{u}^-\|_M}(\mathbf{q}) \coloneqq \left\{ \mathbf{u} \mid \|\mathbf{u}\|_M \leqslant \left\|\mathbf{u}^-\right\|_M \right\} \ , && \text{(energetic consistency)}
\end{aligned} \tag{7.73}$$

where $\mathcal{B}_{\|\mathbf{u}-\|_M}(\mathbf{q})$ denotes the engery ball with radius $\|\mathbf{u}^-\|_M$. The restrictions in (7.73) are visualized in figure 7.4b and the resulting admissible set for the post-impact velocity $\mathbf{u}^+$ is denoted as $\mathcal{S}$. If the pre-impact velocity $\mathbf{u}^-$ is uniquely decomposed into two orthogonal velocities (with respect to the inner product $(\cdot \,|\, \cdot)_M$), that is, $\mathbf{u}^- = \mathbf{v} + \mathbf{v}^\perp$, such that $\mathbf{v}$ belongs to the tangent cone and $\mathbf{v}^\perp$ belongs to the orthogonal counterpart $\mathcal{T}_\mathcal{C}^\perp(\mathbf{q})$, then by inverting $\mathbf{v}^\perp$ and adding it back to $\mathbf{v}$ to obtain $\mathbf{u}^+ = \mathbf{v} - \varepsilon\mathbf{v}^\perp$ for any $\varepsilon \in [0,1]$, the post-impact velocity $\mathbf{u}^+$ lies in the $[0,1]$-interval of Moreau's half-line depicted in 7.4b. This impact law was proposed by Moreau in [120] and fulfills all properties in (7.73). It was shown in [61], that Moreau's impact law is equivalent to the following normal cone formulation in local contact coordinates given as

$$-\xi_i \coloneqq -(\gamma_i^+ + \varepsilon_i \gamma_i^-) \in \mathcal{N}_{\mathbb{R}_0^+}(\Lambda_i) \ , \quad \varepsilon_i \in [0,1] \quad \forall i \in \mathcal{I}(\mathbf{q}) \tag{7.74}$$

if all *restitution* coefficients ε_i are chosen equal to each other, that is, $\varepsilon_i = \varepsilon \ \forall i \in \mathcal{I}(\mathbf{q})$. The impact law in (7.74) is called *Newton-type* impact law of normal cone type because whenever the impact force of a unilateral constraint i is such that $\Lambda_i > 0$, the kinematics

fulfills Newton's impact law, that is, $\gamma_i^+ = -\varepsilon_i \gamma_i^-$. If no impact takes place, that is, $\Lambda_i = 0$, the kinematics requires that $\gamma_i^+ \geqslant -\varepsilon_i \gamma_i^-$. It is possible to equip all set-valued force laws of normal cone type, including the ones discussed in chapter 7, with the above Newton-type impact law. For the unilateral contact with Coulomb friction in (7.31), the following graph $\mathcal{P}_C$ for the impact law is obtained:

$$\mathcal{P}_C = \left\{ (\boldsymbol{\xi}, \boldsymbol{\Lambda}) \;\middle|\; \begin{array}{ll} -\xi_N \coloneqq -(\gamma_N^+ + \varepsilon_N \gamma_N^-) \in \mathcal{N}_{\mathcal{C}_N}(\Lambda_N) \;, & \mathcal{C}_N \coloneqq \mathbb{R}_0^+ \\ -\boldsymbol{\xi}_T \coloneqq -(\boldsymbol{\gamma}_T^+ + \varepsilon_T \boldsymbol{\gamma}_T^-) \in \mathcal{N}_{\mathcal{C}_T(\mu\Lambda_N)}(\boldsymbol{\Lambda}_T) \;, & \mathcal{C}_T(r) \coloneqq \mathcal{B}_2 \cdot r \end{array} \right\} . \tag{7.75}$$

The restitution coefficient ε_T in tangential direction is useful for modeling a super-ball behavior as presented in [62]. With regard to a multi-body system which models a granular material, the Newton-type impact law (7.74) provides a simple and effective way to introduce dissipation into the mechanical system. However, the Newton-type impact law is guaranteed to be energetically consistent if a single global restitution coefficient ε is used. This includes the case for *only* inelastic impacts, where the global restitution coefficient is zero and also for systems using Coulomb friction as in (7.75) (cf. [62]). As granular flows are highly dissipative, a common choice for the restitution coefficients is a completely *inelastic impact*, that is, $\varepsilon_N = 0$ and $\varepsilon_T = 0$, which corresponds to the point $\mathbf{v}$ of *maximal dissipation* in figure 7.4b. The reader should note that the Newton-type impact law in (7.74) may produce incompatible post-impact velocities $\mathbf{u}^+$ if $\mathbf{u}^- \notin -\mathcal{T}_{\mathcal{C}}(\mathbf{q})$. That is the case if some relative pre-impact velocities γ_i^- are strictly positive with $\varepsilon > 0$ as can be deduced from figure 7.4b. A completely inelastic impact circumvents this problem, which is important from a numerical point of view.

An alternative impact law is the extended Poisson impact law as discussed in [63]. It splits the impact process into a *compression* and a *expansion* phase and treats the *compression* phase as a completely inelastic Newton-type impact and the *expansion* phase differently. Poisson's impact law does not suffer from inadmissible post-impact velocities but also suffers from energetic inconsistencies under certain circumstances. To the best of the authors knowledge, an impact law with a local parametrization to access the entire admissible space of post-impact velocities $\mathcal{S}$ in a tractable manner is not known. Broader and in-depth information on the Newton-type impact law and Poison's impact law is given in [62, 63] and with a focus on granular materials in [134].

The next chapter discusses the time discretization of the equation of measures in (7.57) and (7.58) and provides insight into the contact problem.

Chapter 8

Time Discretization

In this chapter, the time discretization of a non-smooth rigid multi-body system is discussed. Event-capturing time-stepping methods in non-smooth dynamics discretize the equation of motion and the impact equation together. Event-driven methods split the integration into smooth and non-smooth parts where all non-smooth events, such as impacts or stick-slip transitions, are captured and resolved at the time they occur. In contrast, event-capturing time-stepping methods approximate the integrated smooth forces and impulsive forces over the time interval by *percussions*.

With the aim of simulating a multi-body system consisting of millions of rigid bodies to approximate a granular material, time-stepping methods are preferred and do not have difficulties resolving impact accumulation points. Furthermore, due to performance reasons, explicit time-stepping methods, despite being less accurate, are favorable compared to implicit schemes. Implicit integration schemes consist of additional non-linearities which need to be solved in each time step for the displacement and velocity update. Time-stepping methods can be obtained from manually discretizing the equation of motion in (5.61) together with the impact equation (7.64). By proceeding in this way, enforcing certain properties such as energy consistency or conservation of angular momentum during continuous free motion of a rigid body, is a painstaking process and needs a lot of ingenuity to arrive at a discretization scheme which fulfills such requirements. In this respect, the trend in future research is devoted to deriving time-stepping schemes directly from the *principle of virtual action*, which is the time-integrated virtual work expression in (5.1). This is analogue to the discretization procedures used in continuum mechanics where the discretized equation of motion of a deformable body in space is obtained by approximating the displacement with elementary trial functions which are substituted directly into the virtual work expression. The same approach can be done in time which leads to so called variational integrators which may possess an additional nice structure by construction, such as symplecticity or the weak enforcement of constraints. References and further work on this topic can be found in [71].

One major numerical problem when simulating large-scale multi-body systems is that time-differentiated constraints are in general prone to drift. A time-stepping procedure on velocity level for a system with geometric unilateral constraints for example does not enforce the impenetrability condition a priori. After some time steps the bodies start to

penetrate progressively and this behavior can only be diminished but not avoided by a smaller time step although the gap function is strictly enforced to be positive in theory. From a numerical point of view, one common method to alleviate these artifacts is often to introduce constraint stabilization techniques, also called drift correction. Constraint stabilization is a broad field and several methods exist. The reader is referred to the references in [185, 20]. Some methods introduce drift correction terms in the numerical time stepping scheme which are mostly of pure non-physical nature and rather a numerical trickery. Other methods perform a projective drift correction at the end of the time step to fulfill the constraint equations to a certain accuracy. Constraint stabilization, when not applied carefully, may lead to unexpected results, such as unwanted energy increase or unwanted impact behavior which can lead to strange non-physical phenomena, for example non-physical wave propagation phenomena in large-scale granular simulations. Furthermore, the numerical parameters of such drift correction techniques are directly coupled to the underlying mechanical system and adjusting these parameters quickly becomes a nightmare. For large-scale multi-body systems with complex geometries, the gap functions of geometric bilateral or unilateral constraints are generally not available in analytical form and only time derivatives are available. This fact makes the development of better time-stepping schemes, which enforce such constraints by construction, a difficult task. Future research on variational integrators might help to address such constraint violation during integration by fulfilling the constraints over a time interval on average. In this work, we mainly use Moreau's time-stepping scheme with and without drift correction for the simulation of a large-scale multi-body system which models a granular material. The drift correction procedure to correct the penetration of unilateral contacts is either performed on velocity level as shown in section 8.2 or on displacement level by an integrated impulse which allows for displacement jumps in the generalized coordinates. The drift correction on displacement level is derived from an explicit integration method and discussed in appendix C. In the following, we briefly summarize Moreau's time-stepping scheme and refer the reader to [108, 170, 120, 117] for a more elaborate discussion and derivation of it.

For the time discretization discussed in the following, the complete set of equations are briefly summarized. The non-smooth equation of motion of a multi-body system can be written as the equation of measures in (7.58) together with set-valued force laws of normal cone type and corresponding Newton-type impact laws as

$$
\begin{aligned}
&\mathbf{M}(\mathbf{q},t)\mathrm{d}\mathbf{u} - \mathbf{h}(\mathbf{q},\mathbf{u},t)\mathrm{d}t - \mathbf{W}\mathrm{d}\mathbf{P} = \mathbf{0}\ , && \text{(equation of motion/impact equation)}\\
&\quad \dot{\mathbf{q}}\mathrm{d}t = \mathbf{F}(\mathbf{q},t)\mathbf{u}\mathrm{d}t\ + \mathbf{b}(\mathbf{q},t)\mathrm{d}t\ , && \text{(kinematic equation)}\\
&\left.\begin{aligned}\boldsymbol{\gamma}_i = \mathbf{W}_i(\mathbf{q},t)^\top\mathbf{u} + \boldsymbol{\chi}_i(\mathbf{q},t)\ ,\\ \boldsymbol{\gamma}_i, \boldsymbol{\lambda}_i \in \mathcal{S}_i\end{aligned}\right\} \quad \forall i \in [1\,;\,k]\ , && \begin{aligned}&\text{(relative velocities)}\\ &\text{(force laws of normal cone type)}\end{aligned}\\
&\mathcal{S}_i \coloneqq \{(\boldsymbol{\gamma}_i, \boldsymbol{\lambda}_i) \mid -\boldsymbol{\gamma}_i \in \mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\lambda})\ ,\ \mathcal{C}_i \coloneqq \mathcal{C}_{i,1} \times \cdots \times \mathcal{C}_{i,l}\}\ ,\\
&\left.\begin{aligned}\boldsymbol{\gamma}_i^{\pm} = \mathbf{W}_i(\mathbf{q},t)^\top\mathbf{u}^{\pm} + \boldsymbol{\chi}_i(\mathbf{q},t)\ ,\\ \boldsymbol{\xi}_i \coloneqq \boldsymbol{\gamma}_i^{+} + \varepsilon_i\boldsymbol{\gamma}_i^{-}\ ,\\ \boldsymbol{\xi}_i, \boldsymbol{\Lambda}_i \in \mathcal{P}_i\end{aligned}\right\} \quad \forall i \in [1\,;\,k]\ , && \begin{aligned}&\text{(pre-/post impact relative velocities)}\\ &\\ &\text{(Newton-type impact laws)}\end{aligned}\\
&\mathcal{P}_i \coloneqq \{(\boldsymbol{\gamma}_i, \boldsymbol{\Lambda}_i) \mid -\boldsymbol{\xi}_i \in \mathcal{N}_{\mathcal{C}_i}(\boldsymbol{\Lambda}_i)\}\ . && (8.1)
\end{aligned}
$$

For reasons of simplicity and in the context of this work, all k set-valued force laws are contact laws discussed in section 7.1 and are formulated on velocity level. Other set-valued laws, such as geometric or kinematic bilateral constraints, can be added to the above set of equations without increasing the complexity of the underlying contact problem. Note that the convex sets $\mathcal{C}_{i,l}$ for a contact i can depend on the contact force $\boldsymbol{\lambda}$ as for the case of a unilateral contact with Coulomb friction. This dependence is omitted for clarity in the remainder of this chapter.

8.1 Moreau's Time-Stepping Scheme

Moreau's explicit event-capturing time-stepping scheme was first introduced in [120] and is basically a midpoint discretization on displacement level and an Euler backward method on velocity level. The intricacies in deriving Moreau's time-stepping method lie in the discretization of the equation of measures together with an approximation of the set-valued force laws in (8.1) over a time interval. Moreau's time-stepping scheme is given in algorithm 8.1.

The reader should note that in the literature the update in step 4 in algorithm 8.1 is commonly written as

$$\mathbf{q}^E = \mathbf{q}^M + \frac{\Delta t}{2}(\mathbf{F}(\mathbf{q}^M, t^M)\mathbf{u}^E + \mathbf{b}(\mathbf{q}^M, t^M)) \tag{8.2}$$

$$\Rightarrow \mathbf{q}^E = \mathbf{q}^S + \frac{\Delta t}{2}(\mathbf{F}^S\mathbf{u}^S + \mathbf{F}^M\mathbf{u}^E + \mathbf{b}^S + \mathbf{b}^M) \ , \tag{8.3}$$

which is a non-symmetric update due to $\mathbf{F}^S := \mathbf{F}(\mathbf{q}^S, t^S)$ and $\mathbf{F}^M := \mathbf{F}(\mathbf{q}^M, t^M)$. We suggest the symmetric version in step 4 because it is closer to Moreau's original update and a better approximation of the implicit update which maintains the unit quaternion constraint for rigid bodies parametrized by quaternions (cf. section 5.2.5 in [108] where $\mathbf{b} = \mathbf{0}$). The numerical treatise of step 3 in algorithm 8.1 is discussed in detail in section 8.3. The next section briefly discusses a commonly used drift correction term in Moreau's time-stepping scheme to reduce the penetration of unilateral contacts and shows why it should not be used for elastic Newton-type impacts.

8.2 Drift Correction in Moreau's Scheme

Some modifications of Moreau's time-stepping scheme exist to add a drift correction term in the discrete normal cone inclusion on velocity level for every inelastic unilateral contact $i \in [1\,;\,k]$ with $\varepsilon_{N,i} = 0$ in the form

$$-\gamma_{N,i}^E \in \mathcal{N}_{\mathbb{R}_0^+}(P_{N,i}) \quad \rightarrow \quad -\left(\gamma_{N,i}^E + \alpha\frac{g_{N,i}^M}{\Delta t/2}\right) \in \mathcal{N}_{\mathbb{R}_0^+}(P_{N,i}) \quad \forall i \in [1\,;\,k] \ , \tag{8.4}$$

Algorithm 8.1 *(Moreau's Time-Stepping Scheme)*:
For a given start time t^S and known displacement $\mathbf{q}^S := \mathbf{q}(t^S)$ and velocity $\mathbf{u}^S := \mathbf{u}(t^S)$ the following 4 steps compute an approximation $\mathbf{q}^E \approx \mathbf{q}(t^E)$ and $\mathbf{u}^E \approx \mathbf{u}(t^E)$ at the end time t^E of the time interval (t^S, t^E) with $\Delta t := t^E - t^S$.

Step 1 : Calculate the displacement $\mathbf{q}^M$ at the midpoint $t^M = t^S + \frac{1}{2}\Delta t$ by:

$$\mathbf{q}^M = \mathbf{q}^S + \frac{1}{2}\Delta t \, \left(\mathbf{F}(\mathbf{q}^S, t^S)\mathbf{u}^S + \mathbf{b}(\mathbf{q}^S, t^S)\right) . \tag{8.6}$$

Step 2 : Calculate the mass matrix $\mathbf{M} := \mathbf{M}(\mathbf{q}^M, t^M)$ and nonlinear term $\mathbf{h} := \mathbf{h}(\mathbf{q}^M, \mathbf{u}^S, t^M)$ at the midpoint. Detect all closed contacts and define the index set

$$\mathcal{I} := \{i \mid g_{N,i}(\mathbf{q}^M, t^M) \leqslant 0 \quad \forall i \in [1\,;\,k]\} . \tag{8.7}$$

Compute all generalized force directions $\mathbf{W}_i := \mathbf{W}_i(\mathbf{q}^M, t^M)$ and nonlinear terms $\boldsymbol{\chi}_i = \boldsymbol{\chi}_i(\mathbf{q}^M, t^M)$ for all contacts in $i \in \mathcal{I}$ and set up the contact graph.

Step 3 : Compute the velocity $\mathbf{u}^E$ by solving the following contact problem:

$$\begin{aligned} &\mathbf{M}\,(\mathbf{u}^E - \mathbf{u}^S) - \mathbf{h}\Delta t - \mathbf{W}\mathbf{P} = \mathbf{0} \ , \\ &\left.\begin{aligned} \boldsymbol{\gamma}_i^S &= \mathbf{W}_i^\top \mathbf{u}^S + \boldsymbol{\chi}_i \ , \quad \boldsymbol{\xi}_i := \boldsymbol{\gamma}_i^E + \boldsymbol{\epsilon}_i \boldsymbol{\gamma}_i^S \\ \boldsymbol{\gamma}_i^E &= \mathbf{W}_i^\top \mathbf{u}^E + \boldsymbol{\chi}_i \ , \quad \boldsymbol{\xi}_i \,, \mathbf{P}_i \in \mathcal{P}_i \end{aligned}\right\} \quad \forall i \in \mathcal{I} \ , \end{aligned} \tag{8.8}$$

with the assembled generalized force direction $\mathbf{W} := [\dots, \mathbf{W}_i, \dots]$ corresponding to the contact percussion $\mathbf{P} := [\dots, {\mathbf{P}_i}^\top, \dots]^\top \approx \mathrm{d}\mathbf{P}\big((t^S, t^E)\big)$.

Step 4 : Compute the displacement at the end time $t^E = t^M + \frac{1}{2}\Delta t$ as

$$\mathbf{q}^E = \mathbf{q}^S + \Delta t \ \mathbf{F}(\mathbf{q}^M, t^M)\frac{\mathbf{u}^S + \mathbf{u}^E}{2} + \Delta t \mathbf{b}(\mathbf{q}^M, t^M) . \tag{8.9}$$

where α is a tuning parameter in a reasonable range $\alpha \in [0, 1]$. The additional term in (8.4) can be motivated by the analysis shown in the following. Enforcing every unilateral contact i at the end time is equivalent with

$$-g_{N,i}(\mathbf{q}^E, t^E) \in \mathcal{N}_{\mathbb{R}_0^+}(P_{N,i}) \quad \forall i \in [1\,;\,k] . \tag{8.5}$$

Substituting $g_{N,i}(\mathbf{q}^E, t^E)$ in (8.5) by the first order terms of the Taylor expansion of $g_{N,i}(\mathbf{q}^E, t^E)$ at $\mathbf{q}^M$ and t^M yields (subscript i omitted)

$$g_N(\mathbf{q}^E, t^E) \approx g_N(\mathbf{q}^M, t^M) + \left.\frac{\partial g_N}{\partial \mathbf{q}}\right|_{\mathbf{q}^M, t^M} (\mathbf{q}^E - \mathbf{q}^M) + \underbrace{\left.\frac{\partial g_N}{\partial t}\right|_{\mathbf{q}^M, t^M} (t^E - t^M)}_{\chi_N^M \frac{\Delta t}{2}} . \tag{8.10}$$

By using Moreau's midpoint rule $\mathbf{q}^E = \mathbf{q}^M + \frac{\Delta t}{2}(\mathbf{F}^M\mathbf{u}^E + \mathbf{b}^M)$, one obtains

$$g_N(\mathbf{q}^E, t^E) \approx g_N^M + \left(\left.\frac{\partial g_N}{\partial \mathbf{q}}\right|_{\mathbf{q}^M, t^M} (\mathbf{F}^M\mathbf{u}^E + \mathbf{b}^M) + \chi_N^M \right) \frac{\Delta t}{2} \tag{8.11}$$

$$\approx g_N^M + \gamma_N^E \frac{\Delta t}{2} \ . \tag{8.12}$$

Inserting the approximation (8.12) into (8.5) gives

$$-\left(\gamma_{N,i}^E \frac{\Delta t}{2} + g_{N,i}^M\right) \in \mathcal{N}_{\mathbb{R}_0^+}(P_{N,i}) \quad \forall i \in [1\,;\,k] \ . \tag{8.13}$$

Multiplying the left-hand side with $2/\Delta t$ leads to the drift correction in (8.4). The drift correction term $2g_{N,i}^M/\Delta t$ is a quantity for a relative velocity which can be interpreted as a term $\varepsilon_i \boldsymbol{\gamma}_{N,i}^B$ where ε_i is an artificial restitution coefficient for a Newton-type impact. This artificial restitution coefficient ε_i is dependent on the gap distance in normal direction which differs among different impact configurations and time step lengths Δt. Some simple numerical simulations with few rigid bodies show that already small changes in Δt lead to completely different impact behavior and solutions of the mechanical system. This drift correction, although counteracting the penetration in normal direction, leads to strange behavior such as energy increase which manifests itself in arbitrary wave propagation phenomena when used for large-scale rigid body simulations. This is an undesirable side-effect and occurs whenever the tuning parameter α is chosen too large. Using the same drift correction for elastic normal impacts, where $\varepsilon_{N,i} \in [0, 1]$, is not suggested since the aforementioned behavior is even worse due to the direct interference with the Newton-type impact law.

A better drift correction method can be obtained by allowing displacement jumps in the kinematics equation in (8.1). Appendix C presents some experimental work of an explicit integrator with displacements jumps which has been proven useful for obtaining a drift correction on displacement level that does not interfere with the impact law.

8.3 The Contact Problem

The contact problem in (8.8) within Moreau's time-stepping scheme can be rewritten with the property (6.24) as implicit proximal equations, that is,

$$\begin{aligned} \boldsymbol{\xi} &= \mathbf{G}\mathbf{P} + \mathbf{c} \ , \\ \boldsymbol{\xi}_i &= \boldsymbol{\gamma}_i^E + \boldsymbol{\epsilon}_i \boldsymbol{\gamma}_i^S \ , \quad (\boldsymbol{\xi}_i, \mathbf{P}_i) \in \mathcal{P}_i \ , \quad \forall i \in \mathcal{I} \\ \text{with: } \mathcal{P}_i &:= \left\{ (\boldsymbol{\xi}_i, \mathbf{P}_i) \mid \mathbf{P}_i = \operatorname{prox}^{R_i}_{\mathcal{C}_{i,1}\times\cdots\times\mathcal{C}_{i,l}}(\mathbf{P}_i - \mathbf{R}_i^{-1}\boldsymbol{\xi}_i) \right\} , \end{aligned} \tag{8.14}$$

where the following definitions are used

$$\begin{aligned}
\mathbf{G} &\coloneqq \mathbf{W}^\top \mathbf{M}^{-1} \mathbf{W} \;, & \mathbf{c} &\coloneqq \mathbf{W}^\top \mathbf{M}^{-1} \mathbf{h} \Delta t + (\mathbf{I} + \boldsymbol{\epsilon})(\mathbf{W}^\top \mathbf{u}^S + \boldsymbol{\chi}) \;, \\
\mathbf{P} &\coloneqq [\dots, \mathbf{P}_i^\top, \dots]^\top \;, & \boldsymbol{\xi} &\coloneqq [\dots, \boldsymbol{\xi}_i^\top, \dots]^\top \;, \\
\boldsymbol{\chi} &\coloneqq [\dots, \boldsymbol{\chi}_i^\top, \dots]^\top \;, & \boldsymbol{\epsilon} &\coloneqq \operatorname{diag}(\dots, \boldsymbol{\epsilon}_i, \dots) \;, \\
\mathbf{R}_i &\coloneqq \operatorname{diag}(\mathbf{R}_{i,1}, \dots, \mathbf{R}_{i,l}) \;, & \boldsymbol{\epsilon}_i &\coloneqq \operatorname{diag}(\boldsymbol{\epsilon}_{i,1}, \dots, \boldsymbol{\epsilon}_{i,l}) \,.
\end{aligned} \tag{8.15}$$

A contact law i consisting of a unilateral contact with Coulomb friction possesses the following structure

$$\mathcal{C}_i \coloneqq \mathcal{C}_{i,N} \times \mathcal{C}_{i,T}(\mu P_N) \;, \quad \boldsymbol{\gamma}_i \coloneqq [\gamma_N, \boldsymbol{\gamma}_T{}^\top]^\top \in \mathbb{R}^3 \;, \quad \mathbf{P}_i \coloneqq [P_N, \mathbf{P}_T{}^\top]^\top \in \mathbb{R}^3 \;, \tag{8.16}$$

$$\boldsymbol{\epsilon}_{i,N} \coloneqq \varepsilon_N \;, \quad \boldsymbol{\epsilon}_{i,T} \coloneqq \operatorname{diag}(\varepsilon_T, \varepsilon_T) \;, \quad \mathbf{R}_{i,N} \coloneqq r_N \;, \quad \mathbf{R}_{i,T} \coloneqq r_T \mathbf{I}_2 \;, \tag{8.17}$$

where the projection metric for every normal cone inclusion, that is, $\mathbf{R}_i \coloneqq \operatorname{diag}(\mathbf{R}_{i,N}, \mathbf{R}_{i,l})$, is chosen proportional to the identity matrix and positive definite with $r_N, r_T > 0$, to simplify the projection.

The contact problem in (8.14) is the discretized version of the general inclusion problem (7.43). Equation (8.14) can be rewritten as a compact global proximal equation including all contacts $i \in [1\,;\, k]$, that is,

$$\begin{aligned}
&\mathbf{P} = \operatorname{prox}^R_{\mathcal{C}_1 \times \dots \times \mathcal{C}_k}\big(\mathbf{P} - \mathbf{R}^{-1}(\mathbf{G}\mathbf{P} + \mathbf{c})\big) \\
&\quad \text{with: } \mathbf{R}^{-1} \coloneqq \operatorname{diag}(\dots, \mathbf{R}_i \,, \dots)
\end{aligned} \quad \Leftrightarrow \quad -(\mathbf{G}\mathbf{P} + \mathbf{c}) \in \mathcal{N}_{\mathcal{C}_1 \times \dots \times \mathcal{C}_k}(\mathbf{P}) \;. \tag{8.18}$$

The nonlinear implicit proximal equation (8.18) needs to be solved for the contact percussion $\mathbf{P}$ at every time step within Moreau's time-stepping scheme. The Delassus matrix $\mathbf{G}$ couples all set-valued force laws and its shape depends on the contact situation at the midpoint time t^M. A contact graph with its corresponding Delassus matrix $\mathbf{G}$ is visualized in figure 8.1 for the example of three contacting bodies $\{A, B, C\}$ and four contacts $\{1, 2, 3, 4\}$. For a granular material model consisting of millions of rigid bodies and contacts, most of the computation time is spent in solving the contact problem (8.14). The formulation in (8.14) with implicit proximal equations can be solved iteratively with a fixed-point iteration scheme over all contacts $i \in \mathcal{I}(\mathbf{q}, t)$. The next section briefly discusses the JOR and SOR Prox iteration schemes which are useful for dense contact problems where the Delassus matrix $\mathbf{G}$ is highly non-sparse. The JOR and SOR Prox iteration schemes on velocity level are used for sparse contact problems and are discussed in the last section. Since the inclusion problem is related to a convex optimization problem, although biconvex for contacts with Coulomb friction but fully convex for only unilateral contacts, the solution strategy directly suggests the use of methods from convex optimization. Solving a convex optimization problem is a broad field and many different approaches exist, for example accelerated gradient projection methods, interior point methods, Krylov subspace methods and many more.

The gradient projection algorithm explained in section 6.2 is a first-order method and is directly related to the JOR Prox fixed-point iteration. For large-scale granular systems, first-order convex optimization procedures provide a good trade-off between accuracy and

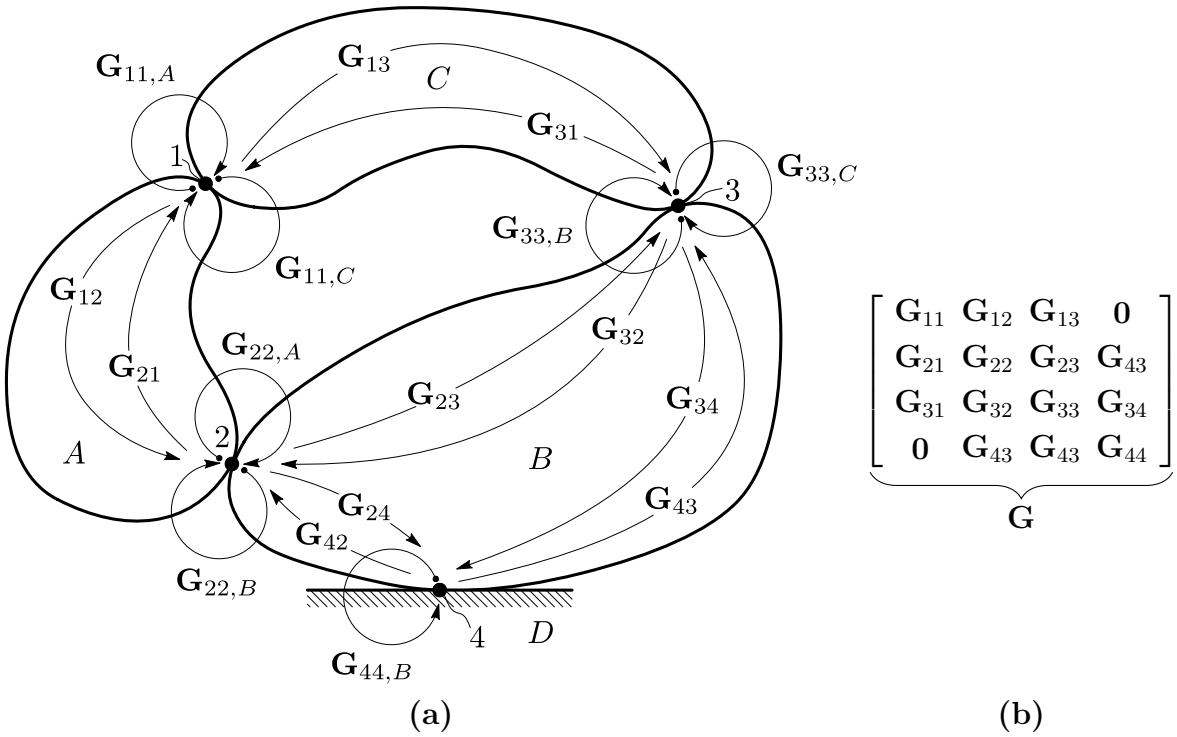


Figure 8.1: Three contacting bodies $\{A, B, C\}$ with a static ground floor D and four contacts $\{1, 2, 3, 4\}$ in (a) and the corresponding Delassus matrix $\mathbf{G}$ where the diagonal block entries contain a sum from both collision partners, e.g. $\mathbf{G}_{11} = \mathbf{G}_{11,A} + \mathbf{G}_{11,C}$.

efficiency and are favorable. The reader is referred to the work in [3] for an extensive overview of current state of the art solution methods used in multi-body dynamics, to the work in [142, 170, 108, 180] for the JOR and SOR Prox algorithms, and to [102] as another reference for different state of the art optimization procedures used especially for granular multi-body systems including their promising accelerated gradient projection algorithm based on the method of Nestrov. Three popular accelerated first-order methods, which achieve ϵ-suboptimality within $\mathcal{O}(1/\sqrt{\epsilon})$ iterations, are presented in [21, 195, 14] and originated from the four ideas of Nestrov in [130, 129, 132, 131] and are well suited for granular rigid body simulations. A projective conjugate gradient method with application to granular dynamics has also been studied in [157] for the case of a polyhedral approximation of the Coulomb friction.

8.3.1 JOR and SOR Prox Iteration

The projected over-relaxed Jacobi (JOR Prox) and projected over-relaxed Gauss-Seidel iteration (SOR Prox) in [142, 170, 108, 181] are two common iteration schemes implemented in the `GRS` framework explained in part II. As a good instructional practice, a motivation of both methods is presented in the following.

Both algorithms are closely related to the classical Jacobi and SOR schemes for the linear

system of equations $\mathbf{GP} + \mathbf{c} = \mathbf{0}$ which can be written as

$$\mathbf{P}^{\eta+1} = \mathbf{P}^{\eta} - \alpha \mathbf{D}^{-1}(\mathbf{G}\mathbf{P}^{\eta} + \mathbf{c}) \ , \qquad \text{(JOR iteration)} \tag{8.19}$$

$$\mathbf{P}^{\eta+1} = \mathbf{P}^{\eta} - \alpha \mathbf{D}^{-1}(\mathbf{L}\mathbf{P}^{\eta+1} + (\mathbf{U}+\mathbf{D})\mathbf{P}^{\eta} + \mathbf{c}) \ , \qquad \text{(SOR iteration)} \tag{8.20}$$

where η denotes the iteration step, α is the relaxation parameter, and the Delassus matrix $\mathbf{G} = \mathbf{L}+\mathbf{D}+\mathbf{U}$ is split into a diagonal matrix $\mathbf{D}$ and a strictly lower and upper triangular matrix $\mathbf{L}$ and $\mathbf{U}$, respectively. By choosing $\mathbf{R}^{-1} = \alpha \mathbf{D}^{-1}$, it is possible to obtain a similar iteration scheme for the proximal point equation in (8.18) compared to (8.19) and (8.20). However, this choice of the projection metric $\mathbf{R}$ determines the metrics $\mathbf{R}_{i,j}$ of all normal cone inclusions j of a contact i and they are in general not positive proportional to the identity matrix and the projections are nontrivial (cf. $\mathbf{R}_{i,T}$ in (8.17)). To simplify the projections, for each normal cone inclusion j of a contact i, one proportional value is chosen, that is, $\mathbf{R}_{i,j} = r_j \mathbf{I}$. A good choice for r_j is to take the maximum of all diagonal entries $\mathbf{D}$ corresponding to the block $\mathbf{R}_{i,j}$. This is reflected in an adapted matrix $\overline{\mathbf{D}}$ and our choice becomes $\mathbf{R}^{-1} = \alpha \overline{\mathbf{D}}^{-1}$. The JOR Prox scheme is then obtained as

$$\mathbf{P}^{\eta+1} = \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\mathbf{P}^{\eta} - \mathbf{R}^{-1}(\mathbf{G}\mathbf{P}^{\eta} + \mathbf{c}) \right) \ , \quad \mathbf{R}^{-1} \coloneqq \alpha \overline{\mathbf{D}} \tag{8.21}$$

$$= \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\mathbf{T}\mathbf{P}^{\eta} + \mathbf{d} \right) \tag{8.22}$$

$$\Rightarrow \left|^{\eta+1}\right. = \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\square \cdot \left|^{\eta}\right. + \left| \right) \ , \tag{8.23}$$

where η denotes the global iteration step with $\mathbf{T} = (\mathbf{I} - \alpha \overline{\mathbf{D}}^{-1}\mathbf{G})$ and $\mathbf{d} = -\alpha \overline{\mathbf{D}}^{-1}\mathbf{c}$.

The SOR Prox scheme on the other hand is obtained as

$$\mathbf{P}^{\eta+1} = \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\mathbf{P}^{\eta} - \mathbf{R}^{-1}(\mathbf{L}\mathbf{P}^{\eta+1} + (\mathbf{U}+\mathbf{D})\mathbf{P}^{\eta} + \mathbf{c}) \right), \quad \mathbf{R}^{-1} \coloneqq \alpha \overline{\mathbf{D}} \tag{8.24}$$

$$= \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\tilde{\mathbf{L}}\mathbf{P}^{\eta+1} + \tilde{\mathbf{U}}\mathbf{P}^{\eta} + \mathbf{d} \right). \tag{8.25}$$

$$\Rightarrow \left|^{\eta+1}\right. = \operatorname{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k} \left(\diagdown \cdot \left|^{\eta+1}\right. + \diagup \cdot \left|^{\eta}\right. + \left| \right) . \tag{8.26}$$

The matrix $\tilde{\mathbf{L}} = -\alpha \overline{\mathbf{D}}^{-1}\mathbf{L}$ is the strictly lower triangular iteration matrix and $\tilde{\mathbf{U}} = \mathbf{I} - \alpha \overline{\mathbf{D}}^{-1}(\mathbf{U}+\mathbf{D})$ is the upper triangular matrix of $\mathbf{T}$. Both schemes are based on the same iteration matrix $\mathbf{T}$ and vector $\mathbf{d}$. The symbolic matrix-vector multiplications are visualized in (8.23) and (8.26). As can be seen in (8.26), the only difference of (8.25) to the JOR Prox scheme is that it subsequently introduces the new calculated values $\mathbf{P}^{\eta+1}$ by matrix $\tilde{\mathbf{L}}$. This *row-wise succession pattern* leads to much faster convergence compared to the JOR Prox scheme. In the literature, the SOR Prox iteration is also called percussion (impulse) propagation method because the succession pattern defines a random order in which the old percussions are updated by the new ones. The succession pattern has influence on the convergence, such as for stacked rigid bodies where a top-down (or bottom-up) pattern is useful. Randomizing the succession pattern during the SOR Prox iteration, at least for Coulomb friction, has a negative effect on the convergence. Despite the fact that both schemes are robust in their convergence behaviour, which merely

depends on the Lipschitz constant of $\mathbf{T}$, it is not trivial to show that they converge to the fixed-point of (8.18). A proof for convergence of the JOR Prox algorithm under the assumption that the convex sets $\mathcal{C}_i$ do not depend on the contact percussion $\mathbf{P}$ and a detailed discussion for these two iteration schemes is given in [170, 108, 181]. The JOR Prox scheme converges if the underlying over-relaxed Jacobi scheme converges for some initial condition. The relaxation parameter should be chosen as $\alpha \in (0, 2)$.

The termination criterion for the JOR Prox and SOR Prox iteration in (8.22) and (8.25) can be chosen for each element in $\mathbf{P}$ as

$$|\mathbf{P}_{(i)}^{\eta+1} - \mathbf{P}_{(i)}^{\eta}| \leqslant |\mathbf{P}_{(i)}^{\eta}| \, T_{rel} + T_{abs} \quad \forall i \ , \tag{8.27}$$

where the subscript $\cdot_{(i)}$ denotes the i^{th} scalar value of the global contact percussion $\mathbf{P}$. The scalars T_{rel} and T_{abs} in (8.27) are the relative and absolute tolerance. The termination criterion (8.27) controls the absolute error if $|\mathbf{P}_{(i)}^{\eta}| \ll 1$ and the relative error otherwise.

Both iteration schems, JOR and SOR Prox, can be evaluated with a sparse representation of $\mathbf{T}$ or $\mathbf{G}$ and for the SOR Prox scheme there is an additional choice in which order to update the entries of the percussion $\mathbf{P}^{\eta+1}$. We assembled the normal cone inclusions per contact and this can also be done differently. For example, by sorting the entries in $\mathbf{P}$ with respect to the type of the normal cone inclusion. For the unilateral contact with Coulomb friction this would mean to update first all unilateral contact forces and then all Coulomb friction forces, which is more in correspondence to the biconvexity of the underlying quasi-optimization problem. The reader should note that the JOR Prox scheme (8.21) can be seen as a gradient projection step given in algorithm 6.1 without the line search and with a step length $\gamma = 1$ as

$$\boldsymbol{\lambda}^{\eta+1} = (1-\gamma)\boldsymbol{\lambda}^{\eta} + \gamma \, \operatorname{prox}_{\mathcal{C}}^{R}\big(\boldsymbol{\lambda}^{\eta} - \mathbf{R}^{-1}(\mathbf{G}\boldsymbol{\lambda}^{\eta} + \mathbf{c})\big) \ . \tag{8.28}$$

Furthermore, applying the line search proposed in algorithm 6.1 can be costly because it involves the evaluation of the objective function given in (7.45) which contains the Delassus matrix $\mathbf{G}$. Note that for unilateral contacts with Coulomb friction, there does not exist an optimization problem and using (7.45) with a line search is regarded from a numerical perspective rather as an experimental method.

As a general note, the SOR Prox scheme should always be preferred to the JOR Prox scheme because it has faster convergence behavior in most cases. The numerical parallelization of both schemes in (8.22) and (8.25) for graphics processing units (GPU) has been conducted in [142].

8.3.2 Velocity JOR and SOR Prox

The inclusion problem in (8.18) can also be rewritten as two equations in the variables $\mathbf{u}^E$ and $\mathbf{P}$ instead of only $\mathbf{P}$, that is,

$$\mathbf{P} = \operatorname{prox}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k}^{R}(\mathbf{P} + \mathbf{R}^{-1}(\mathbf{W}^{\top}\mathbf{u}^E + \mathbf{b})) \ , \tag{8.29}$$

$$\mathbf{u}^E = \mathbf{u}^S + \mathbf{M}^{-1}(\mathbf{h}\Delta t + \mathbf{W}\mathbf{P}) \tag{8.30}$$

$$\text{with: } \mathbf{b} \coloneqq (\mathbf{I} + \boldsymbol{\epsilon})\boldsymbol{\chi} + \boldsymbol{\epsilon}\mathbf{W}^{\top}\mathbf{u}^S. \tag{8.31}$$

The equations above suggest the following JOR iteration from $\mathbf{P}^{\eta} \rightarrow \mathbf{P}^{\eta+1}$ as

$$\begin{aligned} \mathbf{P}^{\eta+1} &= \text{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k}(\mathbf{P}^{\eta} + \mathbf{R}^{-1}(\mathbf{W}^{\top}\mathbf{u}^{E,\eta} + \mathbf{b})) \ , \\ \mathbf{u}^{E,\eta} &:= \mathbf{u}^{S} + \mathbf{M}^{-1}(\mathbf{h}\Delta t + \mathbf{W}\mathbf{P}^{\eta}) \ , \end{aligned} \tag{8.32}$$

which can be expressed as an iteration from $(\mathbf{u}^{E,\eta}, \mathbf{P}^{\eta}) \rightarrow (\mathbf{u}^{E,\eta+1}, \mathbf{P}^{\eta+1})$ as

$$\begin{aligned} \mathbf{P}^{\eta+1} &= \text{prox}^{R}_{\mathcal{C}_1 \times \cdots \times \mathcal{C}_k}(\mathbf{P}^{\eta} + \mathbf{R}^{-1}(\mathbf{W}^{\top}\mathbf{u}^{E,\eta} + \mathbf{b})) \ , \\ \mathbf{u}^{E,\eta+1} &= \mathbf{u}^{E,\eta} + \mathbf{M}^{-1}\mathbf{W}(\mathbf{P}^{\eta+1} - \mathbf{P}^{\eta}) \ . \end{aligned} \tag{8.33}$$

From a numerical and software perspective and with regard to the underlying contact graph defined by the Delassus operator $\mathbf{G}$, it is convenient to formulate (8.33) contact-wise as

$$\mathbf{P}_i^{\eta+1} = \text{prox}^{R}_{\mathcal{C}_i}(\mathbf{P}_i^{\eta} - \mathbf{R}_i{}^{-1}\boldsymbol{\xi}_i^{\eta}) \qquad \forall i \in \mathcal{I} \ , \Big\} \ \forall i \in \mathcal{I} \tag{8.34}$$

$$\mathbf{u}^{E,\eta+1} = \mathbf{u}^{E,\eta} + \mathbf{M}^{-1}\sum_{i \in \mathcal{I}} \mathbf{W}_i(\mathbf{P}_i^{\eta+1} - \mathbf{P}_i^{\eta}) \tag{8.35}$$

$$\text{with:} \quad \left[\begin{aligned} \mathbf{u}^{E,\eta} &:= \mathbf{u}^{S} + \mathbf{M}^{-1}(\mathbf{h}\Delta t + \mathbf{W}\mathbf{P}^{\eta}) \ , \\ \boldsymbol{\xi}_i^{\eta} &:= \mathbf{W}_i{}^{\top}\mathbf{u}^{E,\eta} + \mathbf{b}_i \ , \\ \mathbf{b}_i &:= (\mathbf{I} + \boldsymbol{\epsilon}_i)\boldsymbol{\chi}_i + \boldsymbol{\epsilon}_i\mathbf{W}_i{}^{\top}\mathbf{u}^{S} \end{aligned} \right. \tag{8.36}$$

which is the velocity JOR Prox scheme. The velocity SOR Prox scheme, with a *one-contact* succession pattern, is simply obtained by fully evaluating (8.34) and (8.35) (note the gray bracket) for one single contact i and continuing in this fashion over all contacts. This procedure yields the velocity SOR Prox scheme with a *one-contact* succession pattern as

$$\mathbf{P}_i^{\eta+1} = \text{prox}^{R}_{\mathcal{C}_i}(\mathbf{P}_i^{\eta} - \mathbf{R}_i{}^{-1}\boldsymbol{\xi}_i^{s}), \Big\} \ \forall i \in \mathcal{I} \tag{8.37}$$

$$\mathbf{u}^{E,s+1} = \mathbf{u}^{E,s} + \mathbf{M}^{-1}\mathbf{W}_i(\mathbf{P}_i^{\eta+1} - \mathbf{P}_i^{\eta}) \tag{8.38}$$

$$\text{with:} \quad \left[\begin{aligned} \mathbf{u}^{E,s=0} &:= \mathbf{u}^{S} + \mathbf{M}^{-1}(\mathbf{h}\Delta t + \mathbf{W}\mathbf{P}^{0}), \\ \boldsymbol{\xi}_i^{s} &:= \mathbf{W}_i{}^{\top}\mathbf{u}^{E,s} + \mathbf{b}_i, \\ \mathbf{b}_i &:= (\mathbf{I} + \boldsymbol{\epsilon}_i)\boldsymbol{\chi}_i + \boldsymbol{\epsilon}_i\mathbf{W}_i{}^{\top}\mathbf{u}^{S}. \end{aligned} \right. \tag{8.39}$$

The reader should note that the summation (8.35) simplified to only one term in (8.38), because all other contact percussion deltas $(\mathbf{P}_i^{\eta+1} - \mathbf{P}_i^{\eta})$ are zero when iterating contact-wise over the contact graph. To provide a better understanding of how the loop is performed in (8.37) and (8.38), the velocity SOR Prox procedure is again summarized in algorithm 8.3. As a general note, a contact i corresponds only to two contacting bodies, lets say B_m and B_n where one of them can also be static or excited. The velocity update (8.38) for contact i corresponding to the simulated bodies B_m and B_n will therefore only update the velocities $\mathbf{u}_{B_m}$ and $\mathbf{u}_{B_n}$ of these two bodies. One loop over all contacts in (8.37, 8.38) is denoted as one global SOR Prox iteration with corresponding counter η.

One especially useful termination cirterion is the comparison of the norm of the velocity $\|\mathbf{u}\|_M$ with respect to the metric $\mathbf{M}$, which is the kinetic energy of the mechanical

def `velocitySORProx(...)` **:**
  Data: $\mathcal{I}, \mathbf{M}, \mathbf{h}, \mathbf{u}^S$ and $\mathbf{W}_i, \mathbf{R}_i, \boldsymbol{\epsilon}_i \; \forall i \in \mathcal{I}$ in (8.37, 8.38)
  $converged \leftarrow false$
  $\eta, s \leftarrow 0$, $n_c \leftarrow |\mathcal{I}|$ ▷ initialize counters k, s and number of contacts n_c
  $\mathbf{P}^0 \leftarrow$ `getCachedPercussion(`$default = \mathbf{0}$`)` ▷ init. start percussion from cache
  $\mathbf{u}^{E,0} \leftarrow \mathbf{u}^S + \mathbf{M}^{-1}(\mathbf{h}\Delta t + \mathbf{W}\mathbf{P}^0)$ ▷ initialize end velocity
  while $\neg converged$ **:**
    $\mathbf{u}^{E,c} \leftarrow \mathbf{u}^{E,s}$ ▷ save velocity for convergence check
    for $i \in \mathcal{I}$ **:** ▷ start one global SOR Prox iteration
      $\boldsymbol{\xi}_i^s \leftarrow \mathbf{W}_i{}^\top \mathbf{u}^{E,s} + \mathbf{b}_i$ ▷ might depend on contact law
      $\mathbf{P}_i^{\eta+1} \leftarrow \mathrm{prox}_{\mathcal{C}_i}^R(\mathbf{P}_i^\eta - \mathbf{R}_i{}^{-1}\boldsymbol{\xi}_i^s)$ ▷ update contact percussion of contact i
      $\mathbf{u}^{E,s+1} \leftarrow \mathbf{u}^{E,s} + \mathbf{M}^{-1}\mathbf{W}_i(\mathbf{P}_i^{\eta+1} - \mathbf{P}_i^\eta)$ ▷ update velocity (only two bodies)
      $s \leftarrow s + 1$
    $converged \leftarrow$ `checkConvergence(`$\mathbf{u}^{E,s}, \mathbf{u}^{E,c}, \mathbf{M}, T_{rel}, T_{abs}$`)`
    $\eta \leftarrow \eta + 1$
  return $\mathbf{u}^{E,s}, \mathbf{P}^\eta$

Algorithm 8.3: Basic velocity SOR Prox algorithm.

system. Let the velocity $\mathbf{u}$ consist of each rigid body velocity $\mathbf{u}_{B_i}$ of a body B_i with mass matrix $\mathbf{M}_{B_i}$, then the termination cirterion

$$\left| \|\mathbf{u}_{B_i}^{E,\eta+1}\|_{M_{B_i}} - \|\mathbf{u}_{B_i}^{E,\eta}\|_{M_{B_i}} \right| \leqslant \|\mathbf{u}_{B_i}^{E,\eta}\|_{M_{B_i}} \, T_{rel} + T_{abs} \quad \forall i \tag{8.40}$$

terminates the velocity JOR or SOR Prox algorithm when the change in the kinetic energies of all bodies is below some relative and absolute tolerance T_{abs}, T_{rel}, respectively. A termination cirterion stated in the velocity $\mathbf{u}$ has the advantage of not being influenced by the non-uniqueness of the percussion $\mathbf{P}$ which is likely the case for millions of contacts where the generalized force directions $\mathbf{W}_i$ assembled in $\mathbf{W}$ become linearly dependent and $\mathbf{G}$ positive semi-definite. For the case that all force reservoirs $\mathcal{C}_i$ are convex and not dependent on other forces, the velocities $\mathbf{u}$ are always unique since the existing primal optimization problem (7.48) is strictly convex. For the case of unilateral contacts with friction the velocities $\mathbf{u}$ may be non-unique. The reader should note that the definitions for $\boldsymbol{\xi}_i^\eta$ and consequently also $\mathbf{b}_i$ needs to be replaced when using normal cone inclusions of different type than formulated in (8.1). This includes the unified normal cone inclusion of De Saxé where $\boldsymbol{\xi}_i$ can be replaced with $\tilde{\boldsymbol{\xi}}_i \coloneqq \boldsymbol{\xi}_i + \mu\|\boldsymbol{\xi}_{i,T}\|_2 \mathbf{e}_n^{\mathrm{C}}$ similar to $\tilde{\boldsymbol{\gamma}}$ in (7.33). The velocity SOR Prox scheme is the main iterative technique implemented in the `GRS` framework discussed in chapter 9. Various termination criteria as well as other *succession* patterns, such as the normal-tangential interleaved pattern, are implemented in `ContactGraphVisitors`[35]. Parallel implemenations using graphics processing units have been studied in [17, 183]

8.4 Side Note on the Application

The conducted simulations during this thesis showed that the unified formulation of De Saxé (7.33) does in general not converge faster than Alart-Curnier's separated formulation (7.31). However, De Saxé's variant has a slightly improved convergence behavior for high friction coeffiecients $\mu > 1$ when compared in simulations with identical computations for the metric $\mathbf{R}$ and identical over-relaxation parameter α. The promising idea to add a linear damper in series to the normal cone inclusions on velocity level presented in section 7.1, to be able to fade from a completely soft contact problem to a hard contact problem, or to fade from a frictionless problem to a problem with high friction coefficients during the SOR Prox iteration did not speed up the convergence. In contrary, the manipulation of the damping parameters during the SOR Prox iteration resulted in a restarting behavior of the residual. The idea of using damped contact laws, although not yet successful for the SOR Prox iteration, is considered future work and it is still promising to investigate this idea in combination with a line search either for the dual quasi-objective function in (7.45) or the primal quasi-objective function in (7.48). The reader should note that the velocity JOR Prox scheme consists of a dual feasible update (8.34) and a primal update (8.35) and the velocity JOR/SOR Prox iteration should be extended with a line search in either $\mathbf{P}$ or $\mathbf{u}$ as already included in the basic gradient projection algorithm in (6.30). The line search could be applied contact wise or after one global iteration in the velocity SOR Prox scheme. Future work also focuses on accelerated gradient projection algorithms which have better convergence behavior as shown in the recent promising work [102].

Part II

Software Implementation

"One of the things I really like about programming languages is that it's the perfect excuse to stick your nose into any field. So if you're interested in high energy physics and the structure of the universe, being a programmer is one of the best ways to get in there. It's probably easier than becoming a theoretical physicist."

— Bjarne Stroustrup, *The Essence of C++*, Edinburgh, 2014

This part deals with some of the more difficult software related issues implemented in the Granular Rigid Body Simulation Framework `GRSF` [141] and the Approximate Minimal Volume Bounding Box library `ApproxMVBB` [139] developed in the course of this thesis. In this part, we mainly focus on the parallel capabilities of the `GRS` framework which have been used for the simulation of the granular chuteflow discussed in the last part on the high-performance cluster Euler at the ETH Zurich. The content throughout this part is discussed in a rather conceptual way instead of providing deep insight into the source code and actual implementation intricacies. The reader is therefore only required to have a minimal knowledge in object-oriented programming. Nevertheless, the discussion throughout this part is as specific as needed to empower a reader fluent in an object-oriented programming language and with the knowledge of the theory introduced in part I to understand, reuse, improve and re-implement the source code.

Chapter 9 gives a short overview about the `GRS` framework on a high-level basis and roughly explains the simulation procedure from a user and implementation perspective. Chapter 10 profoundly discusses the requirements and theory of the spatial domain decomposition for the uniform grid and kd-tree decomposition which are two different methods for the parallel load balancing discussed in chapter 11. Chapters 10 and 11 prelude the communication concepts discussed in chapter 12. Chapter 13 explains the theory and algorithms of the mass-splitting method used to parallelize the contact problem in section 8.3. The part concludes with a brief discussion on the visualization and data extraction methods implemented within the `GRS` framework in chapter 14.

Abbreviations such as `RigidBody`[8] or [8]C++ and [3]file used in the remainder of this thesis correspond to source code and file references given in appendix E.

Chapter 9

The GRS Framework

In this chapter, we discuss the software implementations used to simulate large-scale granular multi-body systems. The main software discussed in this chapter is the Granular Rigid Body Simulation Framework (`GRSF`) which encompasses 4 applications: the main granular rigid body application `GRSFSim`, the application `GRSFSimGUI` for the graphical visualization, a parallel version `GRSFSimMPI` which uses the Message Passing Interface (MPI) and the converter tool `GRSFConverter`.

This chapter will rather explain the software framework from a high-level perspective to give the user an overview about the used abstractions and layers instead of focusing on low-level technical aspects. Throughout this section the interested reader is referred to source code locations given in appendix E.1. The source code of the `GRS` framework is available in [141] and is, at the time of writing, not claimed to be a rigid body software development kit but rather a valuable quality-conscious well-structured research tool in the style of the following quote:

> *"Good code is its own best documentation. As you're about to add a comment, ask yourself, 'How can I improve the code so that this comment isn't needed?' Improve the code and then document it to make it even clearer."*
>
> — Steve McConnell, Code Complete, 1993

The source code in [141] has been developed with a special focus on efficiency, memory-safety and the separation of data structures and logic. At time of writing, software frameworks with a focus on granular rigid-body dynamics include the physics engines `pe` [74], `Chrono` [103], `Bullet` [39], the framework `Siconos` [2] and the software project `Solfec` [85]. The software framework `GRSF` has been written in C$^{++11/14}$ and depends mainly on the following libraries `ApproxMVBB`, `Boost`, `Eigen3`, `Ogre`, `pugixml`, `hdf5` and `assimp` (see [141]). Numerical parallel extensions for the GPU in [17, 142] have been written in CUDA C.

9.1 User Perspective

From a user perspective, the execution of a rigid body simulation with the `GRS` framework starts at specifying the mechanical system to be simulated and the various parameters

of the numerical methods to be used for the simulation by means of a scene file in the XML format. The structure of the XML scene description is defined by an XML Schema Definition (XSD). The compact description allows rigid bodies to be defined in groups where each group holds a certain amount of rigid bodies. For each rigid body group, most of the dynamic properties, such as the mass distribution, the initial conditions and the geometries, can be varied over the number of rigid bodies. The scene description also allows to define external forces, in the simplest case, a gravitational force field. A contact parameter map can be specified to define the parameters of contact laws for certain combinations of collision partners. Time-stepper and inclusion solver settings include various parameters for the numerical integration of the dynamical system. The scene description also includes various settings for the load balancing and topology rebuilding techniques explained in chapter 10 for the parallel application `GRSFSimMPI`. The scene file XML description is used at the start of a simulation and also while redistributing bodies during a parallel computation on the cluster to reload the owning rigid bodies by each process. The interpretation of this scene file is performed by the modular implementation in `SceneParser`[15] (see appendix E.1). The simulation can then be launched by any of the three mentioned applications, that is, `GRSFSim`, `GRSFSimGUI` and `GRSFSimMPI`. The graphical version `GRSFSimGUI` allows on-the-fly visualization of the simulation. The output of the simulation consists mainly of a binary file with suffix `.sim` implemented in `MultiBodySimFile`[16] and other various log and data files of the simulation. The output of a simulation can again be visualized with the playback capabilities of `GRSFSimGUI`. Postprocessing, data extraction and data conversion can be performed by the `GRSFConverter` explained in some more detail in chapter 14. Data conversion also includes the generation of Renderman Interface Bytestream (RIB) for rendering a movie from the simulation output. All applications are single-threaded except for the GUI version `GRSFSimGUI` which uses a separate visualization thread aside of the computational thread. All applications, `GRSFSim`, `GRSFSimGUI` and `GRSFSimMPI`, use the same core functionality of the framework and the visualization part in `GRSFSimGUI` and the communication part in `GRSFSimMPI` are decoupled from the core functionality as much as possible.

The next section briefly explains the main components of the implementation and their interaction. Without loss of generality, the following discussion focuses on the parallel implementation for the application `GRSFSimMPI`.

9.2 Implementation Perspective

For the discussion of the core components in the following, we refer to figure 9.1 which shows their simplified dependency graph. The `SimulationManager`[1] is the central part of the framework and controls the input, the integration of the mechanical system and its output. The simulation manager contains some sort of `TimeStepper`[3], a storage class `DynamicsSystem`[5] for the mechanical system, a `StateRecorder`[6] to record the output and for the parallel version also a `BodyCommunicator`[28], a `ProcessCommunicator`[30] and a `TopologyBuilder`[25]. The storage class `DynamicsSystem`[5] is able to construct scene parser modules such that a `SceneParser`[15] directly constructs all objects in-place inside an instance of `DynamicsSystem`[5]. The `DynamicsSystem`[5] storage contains static and

simulated RigidBody[8] objects, global geometries, external forces, initial conditions and other simulation related data. The TimeStepper[3] is responsible for the integration of the stored objects in DynamicsSystem[5] and contains a CollisionSolver[9] which detects all contacts at each time step and an InclusionSolver[11] which solves the contact problem discussed in section 8.3.

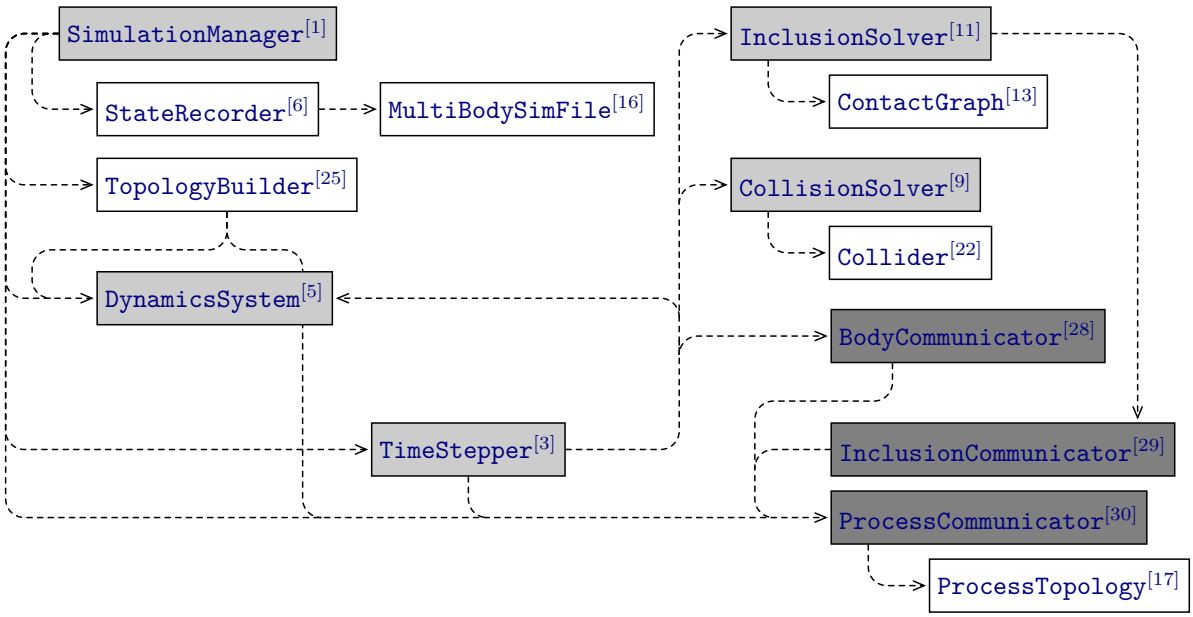


Figure 9.1: A simplified dependency graph of the GRS framework. The core parts are shown in light gray and all communication parts are shown in dark gray.

Each iteration of the simulation loop performed by the SimulationManager[1] mainly consists of a call to the time-stepping routine of the TimeStepper[3] and a successive call to the StateRecorder[6] which writes all rigid body states into the MultiBodySimFile[16]. Each iteration also checks some heuristic to determine if the ProcessTopology[17] needs to be rebuilt to achieve better load balancing. The ProcessTopology[17] contains information about the spatial domain decomposition of the simulation. The notion of a process topology is important for the distribution of the workload during the parallel simulation and is discussed in-depth in chapter 10.

The time step routine of the TimeStepper[3], for the case of Moreau's time-stepping scheme, consists of a single communication step performed by the BodyCommunicator[28] after the first half-time step in (8.6), a successive collision detection step performed by the CollisionSolver[9] at the midpoint as given in (8.7), a contact resolution step for the contact problem in (8.8) performed by the InclusionSolver[11] and a final second half-time step as in (8.9). The contact resolution step performed by the InclusionSolver[11] iteratively solves the contact problem as discussed in section 8.3 and performs the most time consuming computations of the rigid body simulation. In between the iterative solution procedure, the InclusionSolver[11] performs additional communication managed by the InclusionCommunicator[29] for the solution of all mass-split constraints (*split-nodes*) due to the applied mass-splitting method discussed in chapter 13 for the parallelization of the contact problem. The InclusionSolver[11] contains a ContactGraph[13] which is a storage container for all contact and constraint related data and provides an efficient

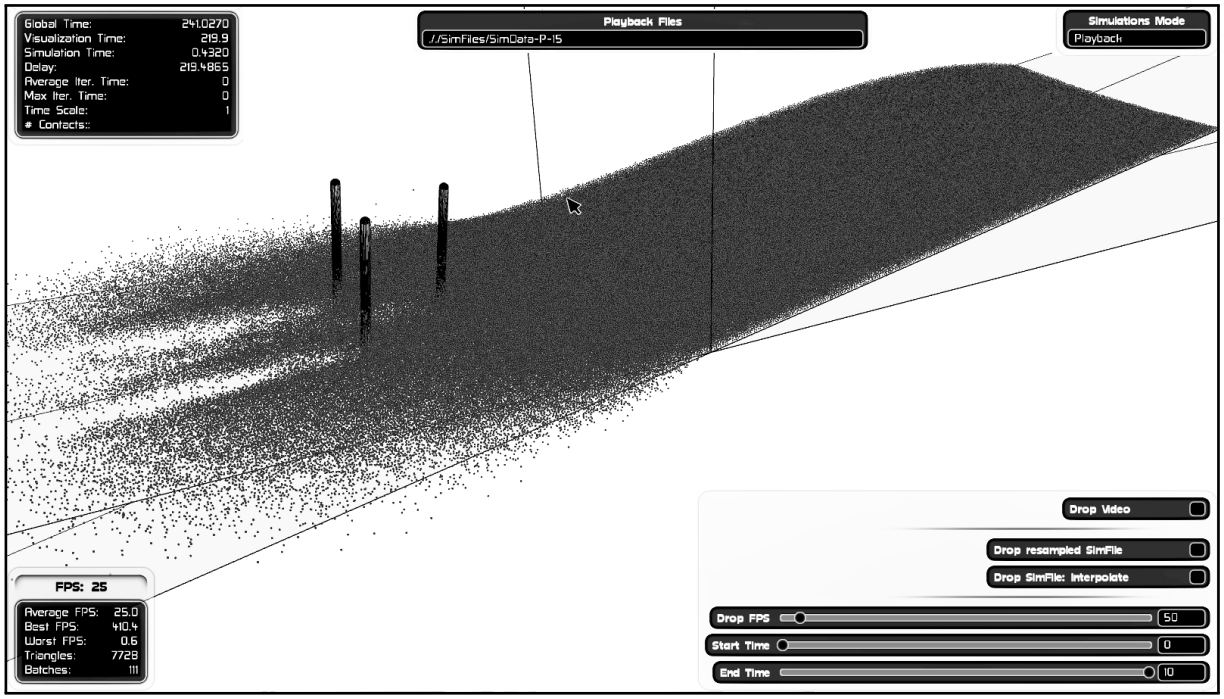


Figure 9.2: The `GRSFSimGUI` application in playback mode to visualize a chute flow experiment of 1 million spheres. The graphical visualization is not real-time due to the read bottleneck of the hard drive.

visitor interface implemented with a static type dispatch system. Different variants of the velocity SOR Prox iteration in section 8.3 have been implemented as visitors[1] in `ContactGraphVisitors`[35] which can be applied efficiently over the `ContactGraph`[13] to iteratively solve the contact problem.

Implementing the communication among processes and keeping the communication interface separated from the numerical computations is one of the most challenging tasks of a parallel implementation. Parallel communication inherently produces a lot of data management and bookkeeping tasks. Maintaining a correct communication pattern for the simulation of large-scale rigid body systems can bring even an experienced programmer to the verge of despair. All communication related bookkeeping tasks are separated from all other parts and implemented in the `BodyCommunicator`[28] and the `InclusionCommunicator`[29]. The `BodyCommunicator`[28] is responsible to correctly communicate the data of shared and non-shared rigid bodies among the neighbors of a process. Although the communication logic implemented in the `BodyCommunicator`[28] is not a mystery once the concept is understood, following the source code is very intricate. For that reason, chapter 12 has been especially devoted to the communication concepts of the `GRS` framework with a focus on the communication logic executed by the `BodyCommunicator`[28]. The sequence diagram in figure 9.3 gives a chronological overview on the time-stepping procedure during a rigid body simulation performed by the application `GRSFSimMPI`.

[1] The visitor design pattern is a method of separating an algorithm from the object on which it operates. A visitor is a function object which accepts an instance of the object on which it operates. For the case of a contact graph, a visitor operates on contacts and constraints in the contact graph.

The next chapter discusses the spatial domain decomposition used to build the `ProcessTopology`[17] which governs the communication logic of the parallel rigid body simulation performed by the `BodyCommunicator`[28] and the `InclusionCommunicator`[29].

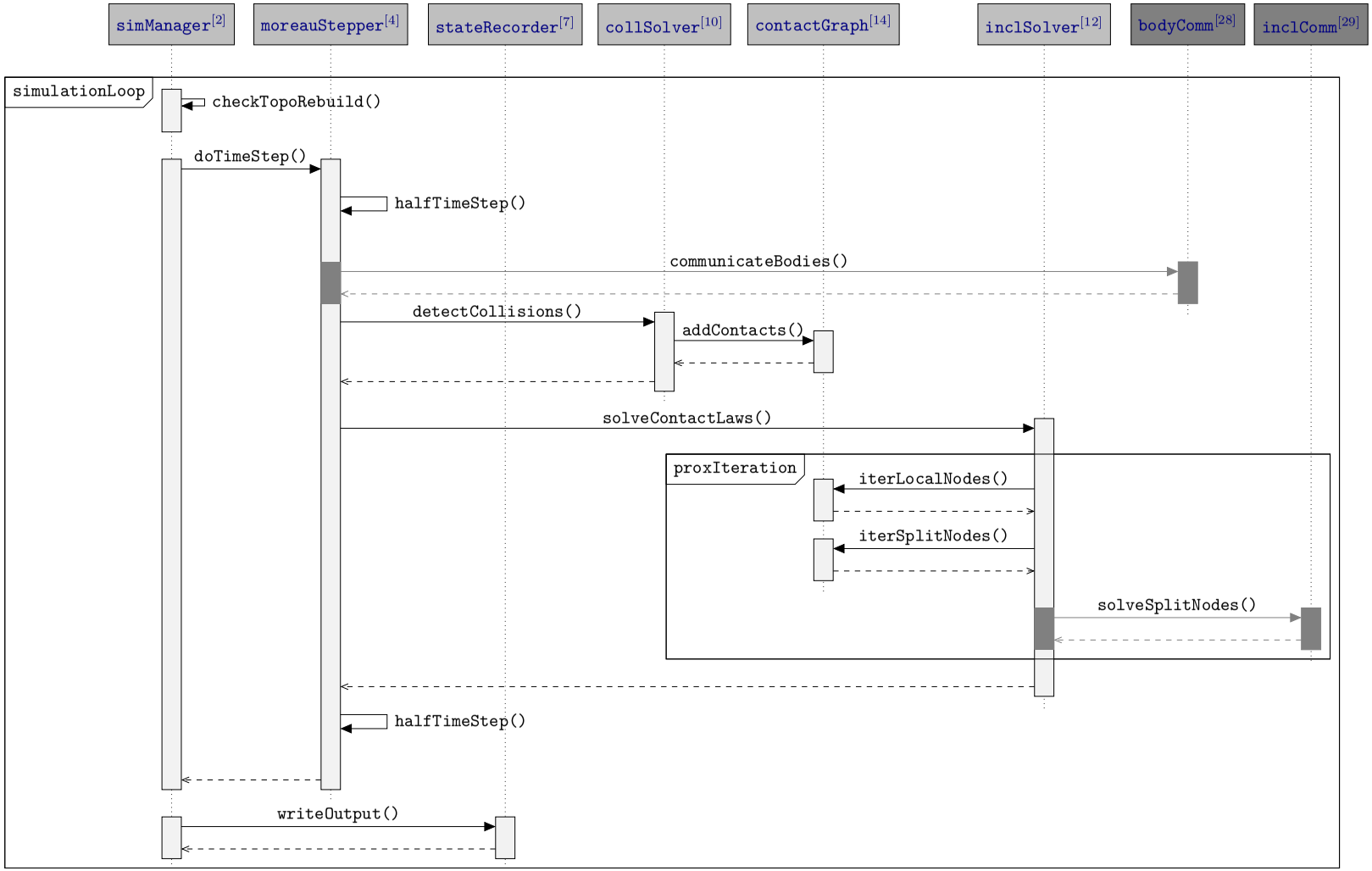


Figure 9.3: Simplified sequence diagram of the time-stepping loop of a rigid body simulation performed by the application `GRSFSimMPI`.

Chapter 10

Spatial Domain Decomposition

When simulating a large-scale rigid multi-body system, such as millions of beads of a granular material, in parallel by using p processes, the computational formulation of the underlying mechanical problem has to be divided into equal shares among the participating processes $i \in [1\,;\,p]$. In this section, we discuss some methods to tessellate the domain covered by a rigid body assembly in $\mathbb{E}^3$ at a fixed time t into p process domains $P_i \subseteq \mathbb{E}^3\ \forall i \in [1\,;\,p]$ which forms a process topology $\mathcal{T} = \bigcup_{i\in[1\,;\,p]} P_i$. The topology $\mathcal{T}$ is used to split the workload of the simulation, namely the numerical time-stepping procedure of all rigid bodies, into p shares. The static approach at a time t is discussed first and more detailed insight into the dynamic approach, including load balancing and topology rebuilding techniques, is given in chapter 11. In the next sections, the notation P_i is used to simultaneously denote the process domain of process i as well as the process i itself.

10.1 Requirements

The requirements of a process topology $\mathcal{T}$ in the context of this work are summarized in the following:

- **Non-overlapping and Unique Owner:** Each process domain P_i has a set $B_{P_i,loc}$ of belonging rigid bodies, also called *local* bodies, given as

$$B_{P_i,loc} \coloneqq \{B_j \mid \mathbf{r}_{S_j} \in P_i\}\ , \tag{10.1}$$

  where all sets $B_{P_i,loc}\ \forall i \in [1\,;\,p]$ are unique and must not overlap. Therefore by (10.1), each body B_j corresponds to a unique process domain determined by the location of its center of gravity $\mathbf{r}_{S_j}$. This implies that also the process domains P_i must not overlap.

- **Coverage:** The topology should cover the maximal possible space occupied by a rigid body assembly. In the context of this work, we assume this to be the space $\mathbb{E}^3$, that is, $\mathcal{T} = \mathbb{E}^3$. This strong requirement is justified by the fact that the location of a body outside of $\mathcal{T}$ with no belonging process domain is unacceptable.

- **Convexity:** The process domains P_i of a process topology $\mathcal{T}$ should be convex.

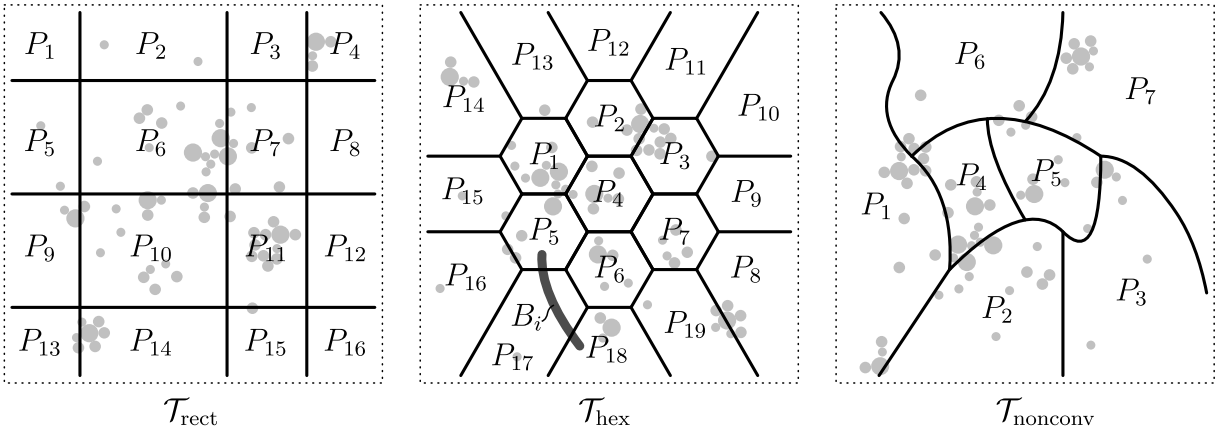


Figure 10.1: Three domain decompositions of a rigid body assembly (gray dots): a rectlinear and a hexagonal-like decomposition, $\mathcal{T}_{\text{rect}}$ and $\mathcal{T}_{\text{hex}}$, respectively, and a non-convex generic decomposition $\mathcal{T}_{\text{nonconv}}$. The boundary cells extend to $\pm\infty$ such that the whole space $\mathbb{E}^2$ is covered.

This requirement allows for more efficient search queries, for example, membership or collision queries of a body.

Figure 10.1 shows a rectlinear decomposition $\mathcal{T}_{\text{rect}}$, a hexagonal-like decomposition $\mathcal{T}_{\text{hex}}$ and a generic decomposition $\mathcal{T}_{\text{nonconv}}$ in $\mathbb{E}^2$, where the latter does not fulfill the requirements above.

In order to use a process topology $\mathcal{T}$ for the simulation, the following two queries have emerged to be useful (see the interface in `ProcessTopology`[17]):

- **Owner Query:** A function `belongsBodyToProcess(`b`)` which returns the unique process domain index/reference to which the center of gravity $\mathbf{r}_S$ of body b belongs as well as a boolean flag telling if the calling process is the direct owner or not.
- **Overlap Query:** A function `checkOverlap(`b`)` which reports the indices/references of all process domains which the geometry of body b overlaps.

It is important to mention that the overlap query `checkOverlap` is efficiently implemented such that when evaluated by a process P_i, it only reports indices/references of process domains in a certain neighborhood around process domain P_i for a body b and is thus named `checkOverlapLocal`.

The neighborhood N_i of a process P_i is important from a communication perspective. The neighborhood N_i defines the set of processes to which a process P_i can communicate which implicitly determines the knowledge a process P_i can obtain during a simulation. It is most natural, and also implemented in this way, to define the neighborhood N_i of a process P_i as the set of all adjacent process domains to P_i. The definition of adjacency is dependent on the domain decomposition and will be explained for each decomposition individually.

Our definition of the neighborhood immediately leads to the strong limitation that a body B_i with owning process P_j must not overlap processes not contained in N_j, since

otherwise an overlap cannot be communicated. For a given topology class and a simulation setup of rigid bodies with a too large disparity in their geometry, this limitation may inhibit an efficient tessellation of the assembly. This again leads to performance issues during the parallel simulation. An example is a simulation of small and very large spheres. For another example, consider the dark gray body B_i in $\mathcal{T}_{\text{hex}}$ in figure 10.1: if its belonging process is P_5, the overlap query `checkOverlapLocal(`$b = B_i$`)` evaluated in process P_5 would only return P_{17} and falsely neglect the non-adjacent domain P_{18} and process P_5 has no knowledge about body B_i overlapping P_{18}.

This locality restriction for a given process decomposition $\mathcal{T}$ can be weakened by either expanding the neighborhood N_i of a process P_i by incorporating more process domains than just all adjacent neighbors. In the worst case this leads to a neighborhood consisting of all processes for each process which leads to inefficient communication. Or, by computing the tessellation such that all rigid bodies are fully contained in their associated neighborhood, that is, the neighborhood of their belonging process.

Two domain decomposition approaches have been implemented `GRS` framework: A simple uniform grid decomposition implemented in `GridTopology`[18] and a more elaborate kd-tree decomposition implemented in `KdTreeTopology`[20]. Both topologies are accompanied by their corresponding topology builder `GridTopologyBuilder`[19] and `KdTreeTopology-Builder`[21], repsectively, and explained in some more detail in the following sections. We will especially elaborate on the aforementioned topology requirements for both the grid and kd-tree decomposition.

10.2 Uniform Grid Decomposition

A grid decomposition is a tessellation of an n-dimensional euclidean space $\mathbb{E}^n$ into parallelotopes (rectangular cuboids). In the context of this thesis, the uniform grid where all parallelotopes are congruent, also called regular grid, in $\mathbb{E}^3$ is of most interest. A uniform grid $\mathcal{T}_{\text{grid}} \subset \mathbb{E}^n$ is specified by four parameters: the orientation given by a transformation matrix $\mathbf{A}_{\text{IK}} \in \mathbb{R}^{n\times n}$ of a coordinate system K together with its displacement $\mathbf{r}_{OK} \in \mathbb{E}^n$, a tuple $\mathbf{g} \in \mathbb{N}_+^n$ for the subdivisions in the direction of each basis vector and a tuple $\mathbf{d} \in \mathbb{R}_+^n$ for the uniform cell extents in each direction. Without loss of generality, the transformation matrix $\mathbf{A}_{\text{IK}}$ is assumed to be orthogonal and maps coordinates from the grid coordinate system K to an inertial coordinate system I. Figure 10.2 visualizes a grid in $\mathbb{E}^3$ with $\mathbf{g} = [3, 3, 3]^\top$.

10.2.1 Properties

All derivations in the following are conducted for the n-dimensional case and simplified results for the three-dimensional case are provided by using $\mathbf{d} = [d_x, d_y, d_z]^\top$, $\mathbf{g} = [g_x, g_y, g_z]^\top$ and $\mathbf{i} = [i_x, i_y, i_z]^\top$ which denotes the index of an arbitrary grid cell. The set of all feasible grid cell indices is defined as

$$\mathcal{I}_{\mathbf{g}} \coloneqq \{\mathbf{i} \in \mathbb{N}_0^n \mid \mathbf{i}_{(k)} \in [0; \mathbf{g}_{(k)} - 1]\} \ . \tag{10.2}$$

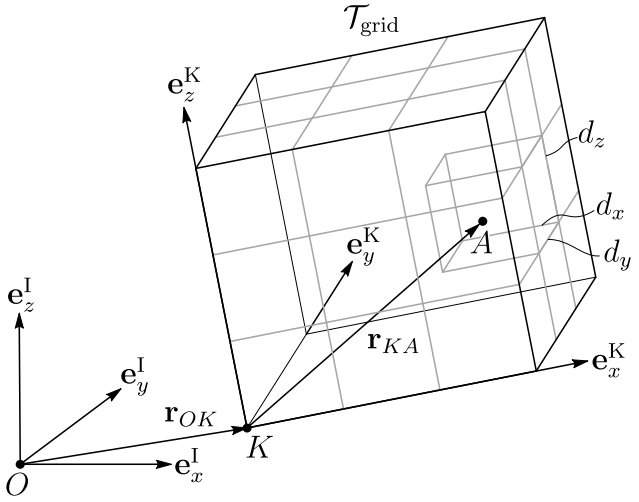


Figure 10.2: A uniform grid $\mathcal{T}_{\text{grid}}$ in $\mathbb{E}^3$ with an orthogonal coordinate system K at the position $\mathbf{r}_{OK}$, a subdivision $\mathbf{g} = [3,3,3]^\top$ and a cell extent $\mathbf{d} = [d_x, d_y, d_z]^\top$. The grid cell which contains point A has the zero-indexed index $\mathbf{i} = [2,1,1]^\top$.

The subscript $\mathbf{x}_{(i)}$ denotes the i^{th} component of the tuple $\mathbf{x}$. The zero-indexed process index p corresponding to a grid cell with index tuple $\mathbf{i} \in \mathcal{I}_\mathbf{g}$ is defined as

$$p = \text{idx}_\text{P}(\mathbf{i}, \mathbf{g}) \coloneqq \sum_{k=1}^{n} \mathbf{i}_{(k)} (\mathbf{g}_{(1)} \cdot \mathbf{g}_{(2)} \cdots \mathbf{g}_{(k-1)}) = i_x + i_y\ g_x + i_z\ g_x\ g_y \in \mathbb{N}_0^+ \ . \tag{10.3}$$

The inverse function $\mathbf{i} \coloneqq \text{idx}_\text{P}^{-1}(p, \mathbf{g})$ is given by the inverse recursion from $k = n \to k = 1$ as

$$\mathbf{i}_{(k)} \coloneqq \left\lfloor \frac{p - \sum_{j=k+1}^{n} \mathbf{i}_{(j)} (\mathbf{g}_{(1)} \cdot \mathbf{g}_{(2)} \cdots \mathbf{g}_{(j-1)})}{\mathbf{g}_{(1)} \cdot \mathbf{g}_{(2)} \cdots \mathbf{g}_{(k-1)}} \right\rfloor \ . \tag{10.4}$$

The grid cell index $\mathbf{i} \in \mathcal{I}_\mathbf{g}$ of the cell containing a point A with displacement ${}_\text{K}\mathbf{r}_{KA} \in \mathbb{R}^n$ in coordinate system K (see figure 10.2) can be computed as

$$\mathbf{i} = \text{idx}_\text{C}({}_\text{K}\mathbf{r}_{KA}, \mathbf{g}) \coloneqq \text{clamp}_{\mathcal{I}_\mathbf{g}} \left(\left\lfloor \frac{{}_\text{K}\mathbf{r}_{KA,(k)}}{\mathbf{g}_{(k)}} \right\rfloor \right) \ , \tag{10.5}$$

where the function clamp operates component-wise and clamps each index $\mathbf{v}_{(k)}$ onto the feasible range $[0\,;\, \mathbf{g}_{(k)} - 1]$. This exactly returns the index $\mathbf{i}$ of the closest cell in $\mathcal{T}_{\text{grid}}$ also for points $\mathbf{r}_{KA} \notin \mathcal{T}_{\text{grid}}$.

The index set $\text{AdjNbs}(\mathbf{i})$ of all adjacent neighbor cells (set of strict 1-neighbors or Moore neighborhood, see definitions in [125]) to a cell with index $\mathbf{i} \in \mathcal{I}_\mathbf{g}$ can be defined as

$$\text{AdjNbs}(\mathbf{i}) \coloneqq \{\mathbf{j} \in \mathcal{I}_\mathbf{g} \mid \max(|\mathbf{i} - \mathbf{j}|) = 1\} \ . \tag{10.6}$$

The neighborhood N_i for process domain P_i with cell index $\mathbf{i} = \text{idx}_\text{P}^{-1}(i, \mathbf{g})$ is then given as

$$N_i \coloneqq \{P_j \mid \mathbf{j} = \text{idx}_\text{P}^{-1}(j, \mathbf{g}) \in \text{AdjNbs}(\mathbf{i})\} \ . \tag{10.7}$$

It is useful to define a k-boundary cell as a boundary cell where k indices are chosen at the possible limit of the range $[0\,;\,\mathbf{g}_{(k)}-1]$. This definition will later be useful when elaborating on the communication aspects of a uniform grid. For a k-boundary cell with grid index $\mathbf{i}$, the size of the adjacent neighbor set $|\text{AdjNbs}(\mathbf{i})|$ can be computed as

$$\text{adjNbs}(k) \coloneqq 3^{n-k}2^k - 1 \ . \tag{10.8}$$

The combinatoric result in (10.8) follows from the fact that for a k-boundary cell with index $\mathbf{i}$, the values of k indices in $\mathbf{i}$ can be chosen at the possible minimum or maximum limit, that is, the term 2^k, and $n-k$ non-limited indices can be shifted by $+1, -1$ or 0, that is, the term 3^{n-k}, and subtracting one for the exclusion of the zero shift.

From (10.8), the case for the non-boundary cell (0-boundary cell) follows directly as

$$\text{adjNbs}(0) = 3^n - 1 = \sum_{m=0}^{n-1} E_{n,m} \ , \qquad E_{n,m} \coloneqq 2^{n-m}C_m^n \ . \tag{10.9}$$

Equation (10.9) results in the maximum number of 26 adjacent neighbors or a grid in $\mathbb{E}^3$. The scalar $E_{n,m}$ in (10.9) represents the number of m-dimensional faces, also called m-faces, of a hypercube with dimension n. The reason for the summation is that each m-face of a 0-boundary cell shares this m-face with one neighbor cell. A 4-dimensional hypercube for example, representing a grid cell in $\mathbb{E}^4$, has $E_{4,0} = 16$ vertices (0-faces), $E_{4,1} = 32$ edges (1-faces), $E_{4,2} = 24$ faces (2-faces) and $E_{4,3} = 8$ 3-faces which totals up to $80 = 3^4 - 1$ four-dimensional neighboring hypercubes[1]. For a grid cell in $\mathbb{E}^3$, this yields $E_{3,0}+E_{3,1}+E_{3,2} = 8+12+6 = 26$ neighbor cells. The second last equality in (10.9) can be shown by using

$$\sum_{m=0}^{n} a^m C_m^n = (1+a)^n \ , \tag{10.10}$$

where C_m^n denotes the binomial coefficient with property $C_m^n = C_{n-m}^n$.

Furthermore, the total amount of k-boundary cells in a grid is given by

$$\text{nBndr}(k, \mathbf{g}) \coloneqq \sum_{t \in I_k} 2^k \prod_{\substack{i=1 \\ i \notin t}}^{n} \max(\mathbf{g}_{(i)} - 2, 0) \ , \tag{10.11}$$

where I_k represents the collection of all k-subsets of the range $\{1, \ldots, n\}$, for example, $I_2 = \{(1,2),(1,3),(2,3)\}$ for $n = 3$. An element $t \in I_k$ in (10.11) represents one combination of index positions of a grid index which all have a possible minimum or maximum value, and for each t there are 2^k possibilities. The other possibilities for the remaining $n-k$ non-limited indices, meaning that a value of an index $\mathbf{i}$ at position $i \notin t$ is in the range $[1; \mathbf{g}_{(i)}-2]$, is taken account by by the product in (10.11). The product in (10.11) evaluates to zero combinations when the maximum function returns zero for some $i \notin t$ with $0 < \mathbf{g}_{(i)} < 2$, which means that the index $i \notin t$ is necessarily at the limit.

[1] See Wikipedia: https://de.wikipedia.org/wiki/Hypercube or [27] for a corresponding table.

Summing up all amounts of k-boundary cells for $k \in [0\,;\,n]$ gives the total number of cells in the grid, that is,

$$\sum_{k=0}^{n} \mathrm{nBndr}(k, \mathbf{g}) = \prod_{i=1}^{n} \mathbf{g}_{(i)} \ . \tag{10.12}$$

The above relation holds true and is non-trivial to show for the general case but is tractable if $\mathbf{g} = [g, \ldots, g]^\top \in \mathbb{N}_+^n$ and $g > 2$, which yields

$$\sum_{k=0}^{n} \mathrm{nBndr}(k, \mathbf{g}) = \sum_{k=0}^{n} 2^k (g-2)^{n-k} C_k^n = (g-2)^n \sum_{k=0}^{n} \frac{2^k}{(g-2)^k} C_k^n \tag{10.13}$$

$$\overset{(10.10)}{=} (g-2)^n \left(1 + \frac{2}{g-2}\right)^n = g^n \ . \tag{10.14}$$

The grid in figure 10.2 with $\mathbf{g} = [3, 3, 3]$ has exactly $\mathrm{nBndr}(2, \mathbf{g}) = 12$ two-boundary cells.

10.2.2 Construction

Assumed that $\mathbf{A}_{\mathrm{IK}}$, $\mathbf{r}_{OK}$, and $\mathbf{g}$ is given, there is no need to construct anything else as these values already define the grid itself. However, finding good parameters highly depends on the requirements discussed in detail in chapter 11 and is a non-trivial task.

So far, we defined the uniform grid $\mathcal{T}_{\mathrm{grid}}$ to be bounded in the space $\mathbb{E}^n$. It is possible to extend the limit faces of all boundary cells to infinity and denote it by $\mathcal{T}_{\mathrm{grid}}^{\infty}$ to fullfil the total coverage requirement, that is, $\mathcal{T}_{\mathrm{grid}}^{\infty} = \mathbb{E}^n$.

10.2.3 Search Queries

The overlap query `checkOverlapLocal(`b`)` in $\mathbb{E}^3$ evaluated in a 0-boundary grid cell with index $\mathbf{i}$ reduces to a total of $\mathrm{adjNbs}(0) = 26$ collision tests between each neighbor cell's axis aligned bounding box in coordinate system K and body b. Although the overlap check is not a full collision query where other information is returned, such as the contact point or the contact normal, its evaluation is non-trivial for arbitrary geometries. An efficient sphere-box overlap test presented in [91] has been implemented in `ColliderAABB`[24] and `ColliderOOBB`[23]. For other combinations of geometries, the reader is referred to standard literature [50]. For the implementation of the search queries, it is also useful to compute the neighborhood N_i for a process P_i. Computing it as in (10.7) is simple and involves merely some combinatorial index computations.

The grid index function in (10.5) is the main ingredient of the required search query `belongsBodyToProcess` which by construction returns the same result for a bounded grid $\mathcal{T}_{\mathrm{grid}}$ as well as for an extended grid $\mathcal{T}_{\mathrm{grid}}^{\infty}$, namely the closest cell to the center of gravity $\mathbf{r}_S$ of a body b.

The implementation of the search queries can be found in `GridTopology`[18].

10.2.4 Communication

In the following, we evaluate the total number of messages sent by all processes if they communicate with their neighbors. For that, it is necessary to sum up the total amount of neighbors for all k-boundary cells which yields

$$\text{nMessages}(\mathbf{g}) \coloneqq \sum_{k=0}^{n} \text{nBndr}(k, \mathbf{g})\text{adjNbs}(k). \tag{10.15}$$

Simplifying this would involve a lot of mathematical intricacies. However, evaluating the uniform split case with $\mathbf{g} = [g, \ldots, g]^\top$ and $g > 2$ yields

$$\begin{aligned} \text{nMessages}(\mathbf{g}) &= (g-2)^n \sum_{k=0}^{n} \frac{2^k}{(g-2)^k} C_k^n (3^{n-k} 2^k - 1) && (10.16) \\ &\overset{(10.10)}{=} (3g-2)^n - g^n \in \mathcal{O}\left(g^n\right). && (10.17) \end{aligned}$$

The presented grid topology is one of the simplest topologies for a parallel simulation. Table 10.1 lists some of its strengths and weaknesses.

Strengths	**Weaknesses**
• No effort for construction given $\mathbf{A}_{\text{IK}}$ and $\mathbf{r}_{OK}$. • Simple implementation of the search queries. • Simple computation of the neighborhood N_i for a process P_i. • Small efficient data structure.	• Number of neighbors is exponential to the dimension of the space $\mathbb{E}^n$. Maximal 26 neighbors in $\mathbb{E}^3$. • The upper bound of total messages sent is $\mathcal{O}\left(\max(\mathbf{g})^n\right)$. • Uneven data distribution over the cells.

Table 10.1: Advantages and disadvantages of using a uniform grid decomposition for the parallel simulation.

10.3 *K*d-Tree Decomposition

A kd-tree is a space-partitioning data structure in a k-dimensional linear space, in the following assumed to be $\mathbb{E}^n$. For a given input data set in $\mathbb{E}^n$, a kd-tree subsequently divides the data set in two new data sets by a $(n-1)$-dimensional hyperplane $H(\mathbf{n}, \mathbf{p}) = \{\mathbf{x} \in \mathbb{E}^n \mid (\mathbf{n} \,|\, \mathbf{x} - \mathbf{p}) = 0\} \subseteq \mathbb{E}^n$ with normal $\mathbf{n} \in \mathbb{E}^n$ and displacement $\mathbf{p} \in \mathbb{E}^n$. One data set lies in the upper closed half-space $H_0^+(\mathbf{n}, \mathbf{p}) = \{\mathbf{x} \in \mathbb{E}^n \mid (\mathbf{n} \,|\, \mathbf{x} - \mathbf{p}) \geqslant 0\}$ whereas the other lies in the open lower half-space $H^-(\mathbf{n}, \mathbf{p}) = \{\mathbf{x} \in \mathbb{E}^n \mid (\mathbf{n} \,|\, \mathbf{x} - \mathbf{p}) < 0\}$. At first glance, kd-trees may appear to be more theoretical than practical in nature. However,

they are extremely useful in a lot of applications ranging from ray tracing [70] and collision detection [168] in computer graphics through data clustering in the field of machine learning [111] to applications in astronomy [56]. Not to forget the efficient computation of the measure (area for $n = 2$ or volume $n = 3$) of a union of multidimensional rectangles, also known as Klee's measure problem. The space partitioning of a kd-tree is also called *coordinate bisection* in the literature.

Kd-trees are efficient data structures for a lot of algorithms in computer science, such as range searches or the archetypical k-nearest neighbor search query (cf. section 11.1). A kd-tree is a special case of a binary space partition tree in $\mathbb{E}^n$, in short BSP tree. Binary space partition is a general method of recursively dividing a data set or space in two. If hyperplanes are used for constructing a BSP tree, they can have in general arbitrary orientation. This is not the case for kd-trees where the splitting hyperplanes are aligned with a coordinate system, in this context denoted as K.

A kd-tree is described by a binary tree which can be unbalanced or balanced depending on its construction. Figure 10.3 shows a kd-tree decomposition in a coordinate system K in $\mathbb{E}^3$ together with its binary graph representation. The description of a kd-tree, as visualized in figure 10.3, is given by the following parameters: the orientation given by a transformation matrix $\mathbf{A}_{\mathrm{IK}} \in \mathbb{R}^{n \times n}$ for the coordinate system K together with its displacement $\mathbf{r}_{OK} \in \mathbb{E}^n$, a list of all nodes $S = \{\dots, n_i, \dots\}$ and an optional tuple $\mathbf{e}$ for the extent in each direction of the basis K. Each non-leaf node $n_i \in S$ stores its two child nodes, the index of the splitting axis $s_{idx} \in [1\,;\, n]$, and the location s of the splitting hyperplane measured in the direction of the splitting axis and other useful information. The union of all disjoint bounding boxes of all leaf nodes in the kd-tree forms the process topology $\mathcal{T}_{k\text{d-tree}} \subset \mathbb{E}^n$.

10.3.1 Construction

Given $\mathbf{A}_{\mathrm{IK}}$ and $\mathbf{r}_{OK}$, the construction process of a kd-tree consists of finding the splitting positions $\{\dots, s_i, \dots\}$ for all non-leaf nodes. A kd-tree can be built recursively in top-down, breadth-first fashion. There exist a variety of methods to obtain the splitting positions and an optimal method highly depends on the used context. In the application of ray tracing in computer graphics where huge triangle meshes are subdivided, surface area heuristics are used to choose the optimal splitting positions. The surface area heuristic uses a cost function based on probabilistic considerations which optimizes ray-triangle intersection tests. In our context, the data set from which the kd-tree topology $\mathcal{T}_{k\text{d-tree}}$ is constructed, is a point cloud of the center of gravities of all rigid bodies used for the simulation. The following three splitting methods are useful:

- **Midpoint Split:** At each level during construction, the splitting position of the hyperplane is located at the midpoint of the bounding box to be split in the direction of the splitting axis.
- **Median Split:** At each level during construction, the splitting position is located at the median of the point cloud in the direction of the splitting axis.

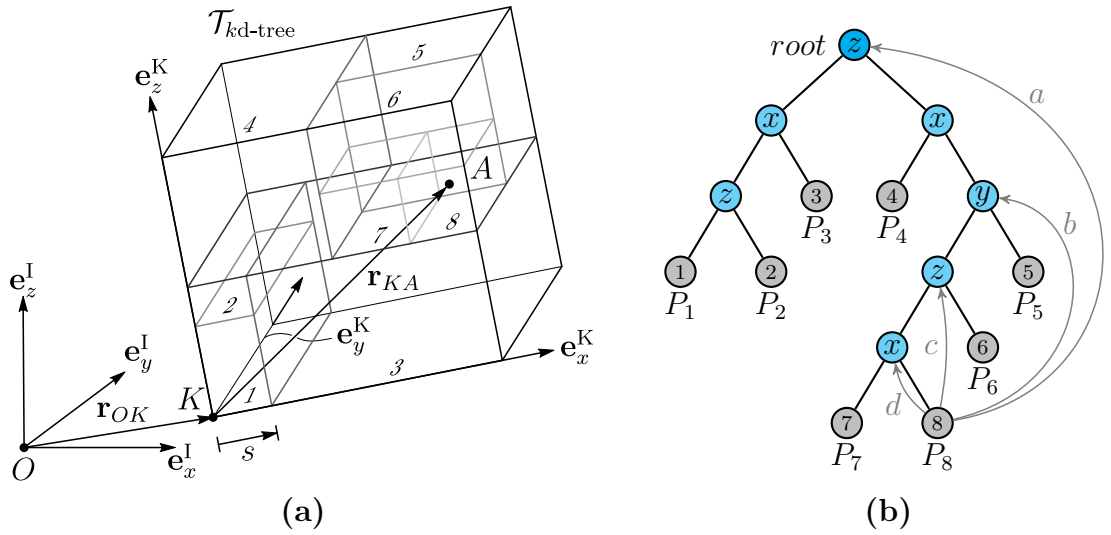


Figure 10.3: A $\mathcal{T}_{k\text{d-tree}}$ in $\mathbb{E}^3$ with an orthogonal coordinate system K in (a) and its corresponding tree structure in (b). The letter inside the non-leaf nodes in (b) denote the used splitting axis at this node. The number inside the leaf nodes corresponds to the numbered bounding box in (a). The left child corresponds always to the lower open half-space. The gray arrows leaving leaf node 8 point to the root nodes of its four boundary subtrees.

- **Geometric Mean Split:** At each level during construction, the splitting position is located at the geometric mean of the point cloud in the direction of the splitting axis.

A simplified recursive breadth-first construction method is shown in algorithm 10.1. At each level during construction (cf. line 7 in algorithm 10.1), independent of the chosen splitting method described above, a splitting heuristic choses the splitting position s and axis s_{idx} according to a quality function. The splitting heuristic implemented in `KdTree`[27] is fully customizable and the default implementation for our application contains a linear quality evaluator which evaluates the splitting quality function

$$\text{splitQ}(r_s, r_p, r_e) \coloneqq 2w_s \cdot r_s + 2w_p \cdot r_p + w_e \cdot r_e, \quad w_s, w_p, w_e \in \mathbb{R}_0^+ \tag{10.18}$$

with weighting parameters w_s, w_p and w_e. The parameter $r_s \in (0, 0.5]$ denotes the splitting ratio of a potential split along the current axis, that is, a split ratio $r_s = 0.5$ is an exact midpoint split. The second parameter

$$r_p = \frac{min(n_{H^-}, n_{H_0^+})}{n_{H^-} + n_{H_0^+}} \in [0, 0.5] \tag{10.19}$$

is the point ratio of a potential split of the point cloud contained in the current node where n_{H^-} and $n_{H_0^+}$ represent the point count in the lower and upper half space. A point ratio $r_p = 0.5$ means that the lower half space contains the same amount of points as the upper half space which is exactly obtained by a median split. The third important

parameter $r_e \in (0, 1]$ denotes the lowest extent ratio of the two new resulting bounding boxes of a potential split. The extent ratio of a bounding box is defined as the quotient of its minimal extent divided by its maximal extent.

The implemented splitting heuristic maximizes the quality function (10.18) over all possible directions and all specified splitting methods, that is, the midpoint, median and geometric mean split. The splitting recursion is continued till the tree has reached its maximum number of leaf nodes or reached its maximum allowed depth. The splitting heuristic also rejects splits for which the ratios r_s, r_p, r_e are below specified minimal values. Thus, it can happen that the heuristic does not find a feasible split among all specified splitting methods which results in a new leaf node (cf. line 10 in algorithm 10.1).

The two parameters r_p and r_e are of special importance. Keeping the point ratio r_p around 0.5 is essential since it directly influences the workload distribution during the parallel simulation. Keeping the extent ratio r_e at the same time as high as possible leads to less degenerated bounding boxes and counteracts the situation where huge long-drawn bounding boxes have lots of neighboring cells which is bad for communication. Figure 10.4 shows two splitting results of a kd-tree construction. Figure 10.4a shows a tessellation where all methods, the midpoint, median and geometric mean split, are used to maximize (10.18). This results in a high average split, point and extent ratio $\bar{r}_s, \bar{r}_p$ and $\bar{r}_e$, respectively, and a low maximal neighbor count compared to the splitting result in figure 10.4b, where only the midpoint split is used. From a performance viewpoint, figure 10.4a is a better tessellation since the average point count per leaf node and maximal neighbor count is lower which are two determining factors for the overall performance of the parallel simulation.

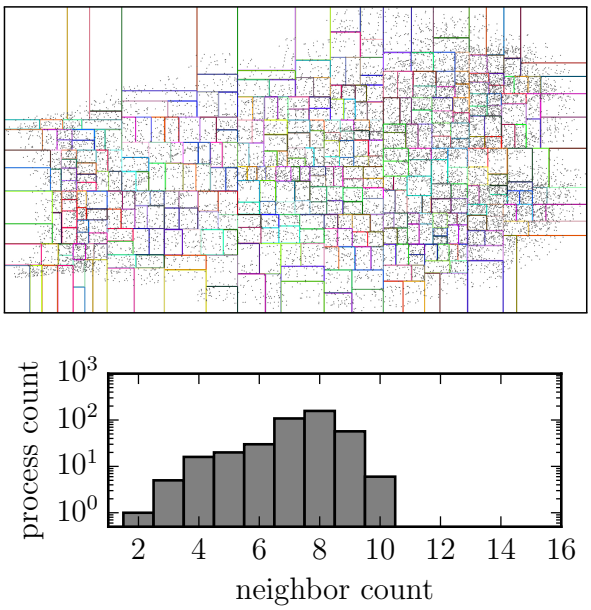
The complexity for building a kd-tree, as shown in algorithm 10, is dependent on the used splitting method. The median, midpoint and geometric mean methods have best case build complexity $\mathcal{O}(n \log n), \mathcal{O}(\log n)$ and $\mathcal{O}(n \log n)$, respectively, for n points[1]. Using all methods over all axes results in a best case build complexity $\mathcal{O}(dn \log n)$ in a d-dimensional space.

Although the kd-tree construction subdivides a start bounding box given by the extent $\mathbf{e}$, it also recursively divides the space $\mathbb{E}^n$ in two which results in a binary tree as shown in figure 10.3b: the resulting tessellation already covers the whole space $\mathbb{E}^3$ and every point $\mathbf{p} \in \mathbb{E}^3$ belongs to a unique leaf node which is found by traversing the tree from the root. In this way, the coverage requirement is fulfilled, that is, $\mathcal{T}_{k\text{d-tree}} = \mathbb{E}^n$. For the implementation, it is useful to extend the boundary faces of the bounding boxes of the leaf nodes to infinity.

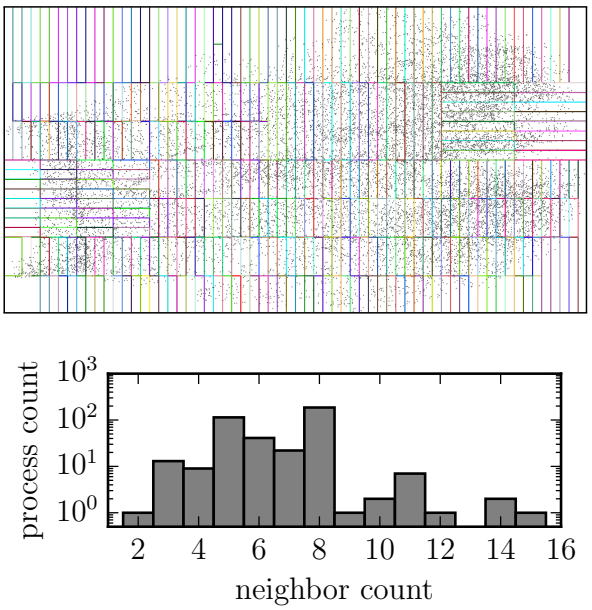
10.3.2 Search Queries

The search query `belongsBodyToProcess` to obtain the belonging process P_i for a body b with center of gravity $\mathbf{r}_S$ is simple and is implemented with a top-down traversal of the binary tree starting at the root node. This simple procedure is shown in algorithm 10.2 and it is worth mentioning this works also for points outside of $\mathcal{T}_{k\text{d-tree}}$. The query `check-`

[1] Confer [197] and the master theorem in [38].



(a) Good splitting by using all methods: midpoint, median, geometric mean.



(b) Bad splitting by using only the midpoint method.

	$\overline{r}_s$	$\overline{r}_p$	$\overline{r}_e$	∅ neighbor count	max. neighbors	∅ point count
(a)	0.40	0.49	0.23	7.3	10	115
(b)	0.5	0.31	0.11	6.74	15	447

(c) Properties of the splits (a) and (b).

Figure 10.4: Two kd-tree decompositions with 400 leafs of 29429 points in $\mathbb{E}^2$ with splitting heuristic weights $w_s = 0, w_p = 1$ and $w_e = 1$ in (10.18).

`OverlapLocal` is similar to the function `belongsBodyToProcess` except that it returns all overlapping neighbors and takes the geometry of the rigid body into account, that is, `isInUpperHs`$(b.\mathbf{r}_S, n_t)$ in algorithm 10.2 needs to be replaced by more advanced collision detection queries. The query `checkOverlapLocal` can be implemented efficiently for a process P_i by starting not at the root node of the kd-tree but at the common ancestor of all boundary subtrees. The boundary subtree of a leaf node i is the subtree starting at the node whose splitting hyperplane is a boundary face of the bounding box of node i. In figure 10.3, the gray arrows a, b, c and d point to the root nodes of the four boundary subtrees of leaf node 8. The implementation of the search queries can be found in `KdTreeTopology`[20].

10.3.3 Communication

Determining the exact number of sent message between neighbors in a kd-tree is not deducible since it is highly dependent on the resulting tessellation for a certain data set. Furthermore, the number of neighbors for a certain leaf node can not be bounded from above like for the grid decomposition in section 10.2. One can imagine extremely bad

```
1  def builKdTree(aabb, points):
       Data: aabb is an axis aligned box around points in the list points.
2      n_r ←createNewNode(aabb, points)                    ▷ create new root node.
3      splitList.push(n_r)                  ▷ push root node onto FIFO queue splitList.
4      while ¬splitList.empty():
5          if treeDepth < maxTreeDepth and leafs.size() < maxLeafs:
6              f ←splitList.front()
7              if f.split(h):                          ▷ try to split node with heuristic h.
8                  splitList.push(f.leftChild(), f.rightChild())
9              else:
10                 leafs.add(f)                                    ▷ this node is a leaf.
11         else:
12             leafs.add(f)                                  ▷ no more splitting allowed.
13         splitList.pop()                                    ▷ pop node at the front.
```

Algorithm 10.1: Principle of a kd-tree construction in top-down, breadth-first fashion.

```
1  def belongsBodyToProcess(b):
       Data: T_kd-tree with root node n_r and leaf node n_i for a process P_i.
2      n_t ← n_r                                  ▷ start the traversal at the root node.
3      while n_t ≠ null:
4          if isInUpperHs(b.r_S, n_t):          ▷ check if point r_S of body b is in H_0^+.
5              n_t ← n_t.rightChild
6          else:
7              n_t ← n_t.leftChild
8      return (n_t, n_t = n_i)
```

Algorithm 10.2: Principle of `belongsBodyToProcess(`b`)` for the kd-tree topology $\mathcal{T}_{k\text{d-tree}}$.

subdivisions which result in an enormous neighbor count for certain leaf nodes. Nevertheless, this is rarely a problem in practice and using the described splitting heuristic with quality function (10.18) for the construction especially counteracts such cases. As long as artificially constructed data sets are excluded, the kd-tree decomposition results mostly in less than 26 neighbors compared to the grid decomposition. Reducing the maximal neighbor count improves the performance for the communication a lot.

Building the neighborhood set N_i as in (10.7) is not simple since the data structure of a leaf node does not encode neighbor information a priori as trivially given for case of the grid decomposition. To obtain this information, additional boundary information has to be propagated during the top-down construction. The boundary information for each node contains all links to boundary subtrees (see gray arrows in figure 10.3) and is continuously updated during the splitting procedure. At the end of the construction, the

boundary information is then available at the leaf nodes which helps in finding all adjacent neighbors. Obtaining the neighbor leaf nodes of a leaf node n_l is achieved by expanding the bounding box of n_l by 99% of the minimal extent of all leaf nodes encountered during construction in all directions. Intersecting the faces of the expanded bounding box of n_l ($2n$ hypersurfaces in $\mathbb{E}^{n-1}$, 6 faces in $\mathbb{E}^3$) with the closest leaf nodes in the $2n$ boundary subtrees. In figure 10.3, expanding the bounding box of leaf node 8 by a small amount and intersecting with the closest nodes in the boundary subtrees given by the arrows a, b, c and d results in the neighborhood $N_8 = \{P_3, P_5, P_6, P_7\}$. The neighborhood N_i in $\mathbb{E}^3$ incorporates all domains which share an edge,vertex or face with domain P_i and is similar to the neighborhood for the grid decomposition.

The following table summarizes some of the strengths and weaknesses of using a kd-tree decomposition.

Strengths	**Weaknesses**
• Obtaining an approximate uniform distribution of the data set over all leaf nodes is possible. • Number of neighbors is not bounded from above but in general less than 26 as for the grid decomposition in $\mathbb{E}^3$.	• Best case build complexity $\mathcal{O}(n \log n)$. • Implementation of search queries is non-trivial. • Neighbor extraction for a process domain is non-trivial. • Data structure handling is more complex (trees).

Table 10.2: Advantages and disadvantages of using a kd-tree decomposition for the parallel simulation.

Chapter 11

Load Balancing

In this chapter, load balancing strategies and techniques for the parallel simulation of large-scale rigid body systems are discussed in more detail. Load balancing in computer science, as the name suggests, is the distribution of a workload onto several processes. This can be done statically or dynamically. Statical load balancing distributes the workload once and keeps the distribution fixed whereas dynamic load balancing constantly redistributes the workload in between the computations. Static load balancing is mainly useful for embarrassingly parallel tasks where it is guaranteed that the load is balanced among all processes during computation. In the context of this thesis, dynamical load balancing is of great importance since the simulations conducted in part III include a fast moving granular material, where a fixed-time domain decomposition as described in chapter 10 is not going to leverage the parallel processing power. The reason is that an approximate uniform distribution of rigid bodies given by an initial domain decomposition only holds for a small time period and the body distribution quickly becomes non-uniform over the processes topology. This slows down the parallel simulation because the process with the largest workload, that is, the one with the largest amount of bodies, is the bottleneck because of the communication and computation overhead for the time-stepping and inclusion solving procedure in each time step (cf. chapter 8). From a high-level perspective, a dynamic load balancing strategy is basically a control loop where a load balancer uses profiling information of the simulation, such as run times of certain expensive routines of the simulation and other heuristics, to optimally balance the total work among all processors. Our method is closely related to [174] where a kd-tree domain decomposition is applied for a simulation of a granular material consisting of up to 1×10^6 two-dimensional discs.

The two important requirements for a dynamic load balancing technique in the context of this thesis are

- **Uniform Workload Distribution:** The computational time for one time step used by a process is dependent on a lot of parameters, such as the number of bodies and the size of the inclusion problem in the process domain and the size of the adjacent neighborhood N_i. One can define a workload heuristic W_i depending on these and other factors which approximately determines each process's workload. This heuristic can then be used together with profiling information to uniformly

balance the workload among all processes. For reasons of simplicity, we mainly considered this parameter W_i to be the total number of bodies located in a process domain P_i.

- **Small Overhead:** The load balancing should only take a fraction of the time of the overall rigid body simulation and its communication overhead should be small.

There exist a lot of dynamic load balancing techniques (e.g. [37, 25]) which could be studied for a parallel rigid body simulation. One technique is to continuously update the domain decomposition depending on a simple heuristic, for example, the distribution of the bodies over the process topology. Together with a pool of available processes, which is managed by a master process, one can imagine a setup where processes are continuously assigned and re-assigned to a group of non-empty process domains in the process topology. This thought experiment quickly raises some very difficult questions as: how should the communication between the processes work since every process needs to know its exact neighbors which constantly change due to the dynamic assignment of the processes? Even though it is certainly possible to implement such a system with a master-slave approach, its intricate communication logic and overhead is significant. A simpler approach is chosen in the context of this work and briefly outlined in the remainder of this chapter.

11.1 Simple Topology Rebuilding

The approach described in the following is the simplest one can imagine. It is a master-slave method where the master process rebuilds the decomposition and shares its results with all other processes. The decision to rebuild the decomposition is determined by communication with the master process which decides based on the workload heuristic W_i, for example, the distribution of the number of bodies or the computational time spent for the last iterations. At time of writing, the topology building process in `GridTopologyBuilder`[19] and `KdTreeTopologyBuilder`[21] has not yet been parallelized. The parallelization of certain subtasks, such as point cloud filtering or rigid body prediction explained in more detail in this section, is straight forward and left for future work. For the following discussion, the time points during a simulation at which a topology rebuilding takes place are denoted as $\{t_k, t_{k+1}, \dots\}$.

The rebuilding procedure, implemented in `TopologyBuilder`[25], at a time t_k performed by the master process can be cast into the six steps shown in algorithm 11.1 independent of the used domain decomposition method. Steps 2,3 and 4 are explained in more detail in the sequel of this section.

Step 2: Outlier Removal

The outlier removal method has been added later in the development process of the `GRS` framework. This step is extremely necessary especially for the uniform grid decomposition which despite the bounding box fit can still result in unfavorable distribution of the workload. This is experienced especially for simulations where the body distribution

Algorithm 11.1 *(Simple Topology Rebuilding)*:

Step 1 : **Send/Receive Information:** The master process receives all necessary data from all other processes (cf. [33]C++). This includes mainly all center of gravities $\mathbf{r}_{S_i}$ and masses m_i of all bodies B_i and optional local computed data depending on the bounding box fit in step 3.

Step 2 : **Outlier Removal:** The received point cloud P consisting of all centers of gravity is filtered by a statistical outlier removal method with the help of a kd-tree and a k-nearest neighbor search algorithm.

Step 3 : **Point Mass Prediction:** The filtered point cloud P_f consisting of all relevant point masses is predicted over time which results in a bigger final point cloud P_p.

Step 4 : **Bounding Box Fitting:** A bounding box $B(P_f) \subset \mathbb{E}^3$ is fitted to the filtered point cloud P_f. For this task, three methods have been implemented:

- **Approximate MVBB:** An approximate minimal volume bounding box fitting described in section 11.2. Some variations of this algorithm include user provided hints such as one or two predefined directions of the resulting bounding box.
- **AABB:** An axis aligned bounding box fitting.
- **PCA:** The principle component analysis (PCA) described in appendix B is based on an Eigenvalue decomposition of the binet tensor of the point cloud where the Eigenvectors define the bounding box directions.
- **Specified OBB:** The user specifies the object oriented bounding box (OBB) or two direction vectors.

Step 5 : **Topology Construction:** One of the decompositions explained in chapter 10, either a uniform grid or a kd-tree decomposition, is computed by the master process for the point cloud P_p (cf. [19, 21]C++).

Step 6 : **Broadcast:** The data structure of the computed process topology is broadcast (cf. [34]C++) to all participating processes. This process topology is used for the next time range $[t_k, t_{k+1})$ of the simulation.

becomes inhomogeneous over time. This is also the case for the conducted simulations in part III. The k-nearest neighbor search for the statistical outlier removal is a non-trivial standard algorithm which efficiently operates recursively on a kd-tree data structure in bottom-up fashion by using a priority queue data structure with maximal size k which sorts the points according to their distance. The algorithm returns the k nearest neighbor points of a point cloud for an input point $\mathbf{p}$ with respect to some metric. The k-nearest neighbor search is shown in algorithm 11.3. The implementation of the outlier removal

```
1  def getKNearestNeighbors(p):
       Data: T_kd-tree with root node n_r, priority queue kNpQ with maximal size k
             for k nearest points, a LIFO node stack nStack to trace parent nodes.
2      nStack ← getNodePathOfPoint(n_r,p)   ▷ obtain node stack nStack such that
         node nStack.front() is a leaf node containing p
3      maxDistSq ← 0   ▷ initialize the maximal squared distance of the found points
4      while ¬nStack.empty():
5          n_c ← nStack.front()   ▷ get current parent node
6          if ¬n_c.isLeaf():   ▷ if current node is not a leaf
7              if ¬n_c.isleftVisited():   ▷ visit left subtree (H^-) if not done
8                  n_c.setLeftVisited()
9                  if kNpQ.full():   ▷ if we have k points already
10                     i ← n_c.splitAxis()
11                     d ← n_c.splitPos() − p_(i)   ▷ distance to splitting hyperplane
12                     if d ⩽ 0 ∧ d² ⩾ maxDistSq:
13                         continue   ▷ p ∈ H_0^+, no overlap of current max ball with H^-
14                 nStack.push(n_c.leftChild)   ▷ go to left child
15                 continue
16             elif: ¬ n_c.isRightVisited():   ▷ visit right subtree (H_0^+) if not done
17                 n_c.setRightVisited()
18                 if kNpQ.full():   ▷ if we have k points already
19                     i ← n_c.splitAxis()
20                     d ← n_c.splitPos() − p_(i)   ▷ distance to splitting hyperplane
21                     if d > 0 ∧ d² > maxDistSq:
22                         continue   ▷ p ∈ H^-, no overlap of current max ball with H_0^+
23                 nStack.push(n_c.rightChild)   ▷ go to right child
24                 continue
25         else:   ▷ if current node is a leaf, add all points to the priority queue
26             if n_c.size() > 0:
27                 kNpQ.push(n_c.getPoints())   ▷ add all points of n_c to the prio.
                     queue
28                 maxDistSq ← ‖kNpQ.top() − p‖₂²   ▷ update max. squared distance
29         nStack.pop()   ▷ go to parent node by removing n_c
30     return kNpQ
```

Algorithm 11.3: Principle of obtaining the k nearest neighbor points of a point cloud to an input point $\mathbf{p}$.

in `NearestNeighbourFilter`[26] computes the mean of the distances of the k-nearest neighbors for every point in the point cloud. These means result in a histogram from which the standard deviation and mean can be used in a second step to classify outlier points in the point cloud.

Step 3: Point Mass Prediction

By predicting the time evolution of the filtered point masses in P_f, the decomposition in the last step is prepared such that the uniform distribution requirement for the next simulation interval $[t_k, t_{k+1})$ is better fulfilled. The two important parameters for this step are the duration and the accuracy of the prediction. Finding the optimal duration is not straight forward. The process decomposition built from a point cloud predicted over a too long interval may have worse performance over the next time interval $[t_k, t_{k+1})$ due to the imbalance of the body distribution. Thus, the duration is manually adjusted depending on the underlying mechanical problem. The accuracy of the prediction is in direct relation to its computational complexity. For the simulation of a granular material, the explicit standard Euler method is used for the time-stepping of the point masses in P_f. All external forces, except contact forces, acting on the original bodies are reduced to the point masses such that their movement is close to the original bodies. In this context, only the gravitational force is of relevance and already acts at the center of gravity. Collisions during the prediction are only detected between point masses and static geometries in the scene, for example, walls and floors. As soon as a point mass collides, it is projected onto the surface of the colliding object and the prediction for this point mass is stopped. Collision detection is necessary to prevent mass points from being located outside of the feasible space of the given scene. The collision detection has been implemented with point-to-object collision routines which work for all objects which have a well defined inside, for example, half spaces, spheres or capsules, and do not work well for open triangle meshes, for example. Further work focuses on the implementation of a more robust collision detection by finding the collision between a point mass trajectory and the objects of interest. This results in ray-object collision tests as used in ray tracing in computer graphics. The time step for the prediction is typically chosen to be a multiple of the real time step of the simulation.

Step 4: Bounding Box Fitting

It is worth noting that the computation of a bounding box $B(P)$ of an arbitrary point cloud $P \subset \mathbb{E}^n$ with k points is only dependent on the convex hull $cHull\,(P)$ of P. This is true since $P \subseteq cHull\,(P) \subseteq B(P) \subset \mathbb{E}^n$. Finding the exact minimal bounding box $B(P)$ of a point cloud P in $\mathbb{E}^n$ is a very difficult task. The algorithm presented in [194] for a point cloud in $\mathbb{E}^2$, also called the rotating calipers method, requires $\mathcal{O}\,(k \log k)$ time, and $\mathcal{O}\,(k)$ if $cHull\,(P)$ is known. In $\mathbb{E}^3$, the problem is more intricate and the current best-known exact algorithm, published by O'Rourke [143], requires $\mathcal{O}\,(k^3)$ time and $\mathcal{O}\,(k)$ space. O'Rourke's algorithm is known to be extremely difficult to implement and is non-applicable for large point clouds with millions of points. The state of the art is to use approximation algorithms which can be classified into two categories, the *direct* methods which only compute one approximation of the bounding box, the *enumerative* algorithms which choose the best bounding box among a computed set of boxes and the *iterative* ones which iteratively improve the approximation.

The principal component analysis (PCA) as described in appendix B belongs to the class

$\kappa_{n,i} \in$	$\mathbb{E}^n$		
	$n = 2$	$n = 3$	$n = 4$
$i = 0$	$\{\infty\}$	$\{\infty\}$	$\{\infty\}$
$i = 1$	$[2, 2.654]$	$\{\infty\}$	$\{\infty\}$
$i = 2$	$[2, 2.104]$	$[4, ?]$	$\{\infty\}$
$i = 3$		$[4, 7.81]$	$[16, ?]$
$i = 4$			$[16, ?]$

Table 11.1: An extraction from [46] of the approximation factors $\kappa_{n,i}$ in (11.1) for the PCA bounding boxes of a point cloud P for different dimensions n and dimensions i of the considered faces of $cHull\,(P)$. Question marks denote currently unknown bounds.

of *direct* algorithms. This algorithm computes the eigenvectors of the Binet inertia tensor of P which form the directions of the resulting bounding box. The classical PCA approach is performed over all points in P. The PCA method in general has the deficiency of being extremely sensitive to the distribution of the point cloud due to the Eigenvalue decomposition. Symmetric point clouds are the archetypical bad examples where the PCA method fails to provide a good fitting. Improved adaptations exist where the analysis is performed only over the points on the boundary of the convex hull $cHull\,(P)$ for example or where the associated 2-dimensional minimum area rectangle problem of the projected point cloud onto a single principal direction, that is, the eigenvector to the smallest or largest eigenvalue, is solved. In the thesis in [46], lower and upper bounds on the approximation ratio of PCA bounding boxes are derived. The approximation factor $\kappa_{n,i}$ in $\mathbb{E}^n$ for a PCA bounding box is defined in [46] as

$$\kappa_{n,i} \coloneqq \sup\left(\frac{vol\,(B_{pca,i}(P))}{vol\,(B_{opt}(P))} \mid P \in \mathbb{E}^n,\ vol\,(cHull\,(P)) > 0\right), \tag{11.1}$$

where $B_{opt}(P)$ denotes the exact minimal volume bounding box of point cloud P and $B_{pca,i}(P)$ is the bounding box obtained from the principal component analysis of the i-dimensional faces of $cHull\,(P)$. A 0-,1-, and 2-dimensional face of $cHull\,(P)$ in $\mathbb{E}^3$ is a vertex, edge or a bounding surface respectively. The approximation factor for the whole convex hull is $\kappa_{n,n}$. Table 11.1 gives an overview of the lower and upper bounds on the above approximation ratio $\kappa_{n,i}$ extracted from the extensive work [46]. Table 11.1 shows that, in the worst case, the PCA bounding box computed from $cHull\,(P) \in \mathbb{E}^3$ can still be 7.81 times bigger than the optimal one. At the time of developing the `GRS` framework, the quality of available efficient implementations of certain approximate bounding box algorithms was extremely poor and some implementations were not written in the preferred language C++. As a consequence the open-source library `ApproxMVBB` [139] has been developed alongside the `GRS` framework to provide a robust, efficient and safe implementation of an approximate bounding box algorithm. This library and its *enumerative* algorithm is briefly explained in the next section.

There exist several more *enumerative* methods for obtaining an approximate bounding box. They are also called *brute-force* methods, which sample the space of rota-

tions $SO(3)$ or choose sample directions $\mathbf{d}_i$ in a unit sphere to obtain several approximations $B_{opt}(P, \mathbf{d}_i)$. These methods perform in general worse than the exhaustive grid search method described in section 11.2. An excellent review on this topic and the so far mentioned algorithms can be found in [35] where an *iterative* optimization method is proposed. In [35], the minimal volume bounding box problem is simply cast into a volume optimization problem as

$$\min_{{}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}} \in SO(3)} f({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}}) \ , \tag{11.2}$$

where the objective function $f({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}})$ is simply the volume of the axis-aligned bounding box in coordinate system K, that is,

$$f({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}}) \coloneqq \left(\begin{array}{ll} \min\limits_{{}_{\mathrm{I}}\mathbf{r}_{OK},\ {}_{\mathrm{K}}\mathbf{c} \in \mathbb{R}^3} & \mathbf{e}_{(1)}\mathbf{e}_{(2)}\mathbf{e}_{(3)} \\ \textit{subject to:} & -\dfrac{\mathbf{e}}{2} \leqslant ({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}})^{\top}{}_{\mathrm{I}}\mathbf{p}_i - {}_{\mathrm{K}}\mathbf{c} \leqslant \dfrac{\mathbf{e}}{2} \quad \forall \mathbf{p}_i \in P \end{array} \right). \tag{11.3}$$

The vector ${}_{\mathrm{K}}\mathbf{c}$ denotes the center of the bounding box in the coordinate system K and ${}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}} \in \mathbb{R}^{3\times3}$ is the rotation matrix rotating an inertial coordinate system I to a coodinate system K expressed in system I and $\mathbf{e}$ is the extent of the bounding box (note that ${}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}} = {}_{\mathrm{K}}\mathbf{R}_{\mathrm{KI}}$). The objective function $f({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}})$ cannot be differentiated at all points for certain rotations ${}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}}$ which align at least one face of the bounding box to be flush with one edge of the convex hull $cHull\,(P)$, and is thus C^0. Evaluating the function $f({}_{\mathrm{I}}\mathbf{R}_{\mathrm{KI}})$ is expensive and can be done in $\mathcal{O}\,(k)$ time. Nevertheless, the optimization algorithm in [35] applied to the above optimization problem tends to be even faster than our implemented approach in [139]. An efficient implementation of the presented approach which uses the proposed genetic optimization method, that is, the Nelder-Mead simplex algorithm in [128, 89], is left for further work. Applying it to a representation of (11.2) using quaternions discussed in chapter 3 is even more interesting. The work in [47] can be considered for a good reference about the application of the Nelder-Mead algorithm to Riemannian manifolds such as the special orthogonal group $SO(3)$.

The following combinations of step 3 and 4 for the uniform grid and kd-tree decomposition have been proven useful in practice:

- **Uniform Grid:** Fitting a bounding box is an essential step for this decomposition to improve the uniform distribution of the point cloud. All box fitting methods can be used in step 3. It is however strongly recommended to use the ApproxMVBB procedure because it almost always gives superior results than the other two methods. The ApproxMVBB method, although being more accurate, is significantly slower than the AABB and PCA approach. The AABB or user-provided OOB method are only useful for simple simulation setups where the change of the displacement and orientation of the granular material over time is very low and can be predicted beforehand.
- ***K*d-Tree:** This decomposition already fulfills the uniform distribution requirement if the splitting heuristic is chosen adequately, independent of the given transformation matrix $\mathbf{A}_{\mathrm{IK}}$ (cf. section 10.3). Therefore, no expensive bounding box fitting

is required. Since the most expensive method, that is, the ApproxMVBB fitting, can be adjusted to have a reasonable run time to accuracy ratio, its application in combination with the kd-tree does not harm at all and is recommended.

11.2 The ApproxMVBB Library

The library `ApproxMVBB` [139] is a modern C$^{++11/14}$ library developed in the context of this thesis whose focus lies in computing an approximation of the oriented minimal volume bounding box of a given input point cloud in $\mathbb{E}^3$ consisting of k points.

The *enumerative* algorithm for computing a minimal volume bounding box approximation is based on the work [19] and [99]. The algorithm presented in [19] provides for a value $\epsilon \in (0, 1)$ an approximation in $\mathcal{O}\left(k \log k + k/\epsilon^3\right)$ time whose volume is at most $(1 + \epsilon)$ times the optimal volume. The algorithm can be cast into the following five steps:

Algorithm 11.4 *(Approximation of a Minimal Volume Bounding Box)*:

Step 1 : **Approximate Diameter**: An approximation of the diameter d and its direction $\mathbf{x}$ of the point cloud P is computed as described in [99].

Step 2 : **Initial Bounding Box:** The points P are projected onto the plane perpendicular to the direction $\mathbf{x}$ and the diameter $\mathbf{y}$ of the projected point cloud is computed. The initial bounding box $A \coloneqq B(P, \mathbf{x}, \mathbf{y}, \mathbf{z})$ is computed from the orthogonal basis vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$.

Step 3 : **First Optimization Loop:** An optimization loop successively improves A by cycling through all three basis vectors, that is, $\mathbf{d} \in \{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$, and computing the exact optimal bounding box for the projected point cloud in the direction $\mathbf{d}$. The exact optimal bounding box given a direction $\mathbf{d}$ is denoted as $B_{opt}(P, \mathbf{d})$. This procedure is demonstrated in algorithm 11.5.

Step 4 : **Sample Point Cloud:** A representative sample of the point cloud P is chosen by snapping all points to a uniform grid defined by the initial box A and keeping only the extreme points.

Step 5 : **Exhaustive Grid Search:** The optimal bounding box is searched among all boxes $B_{opt}(P, \mathbf{d})$ with direction $\mathbf{d}$ being collinear to the grid cell positions in a uniform grid defined by box A and subdivisions $\mathbf{g} = [g_x, g_y, g_z]^\top$.

The optimal bounding box computation for $B_{opt}(P, \mathbf{d})$ in step 3 and 5 involves geometric algorithms such as the convex hull and minimal area rectangle of a two-dimensional point cloud, where the latter is dependent on the first one. The fundamental building blocks of such geometric algorithms, such as the computation of a convex hull in two or three dimensions, are called *geometric predicates*. A *geometric predicate* is mostly a two- or three-way decision algorithm, for example, the routine which determines if a point is left, right or on a straight line in two dimensions. If the implementation of such *geometric predicates* are numerically fragile, geometric algorithms relying on them may provide

completely wrong results if given bad conditioned input data. Exact arithmetics, that is, by using higher precision floating point arithmetics in combination with floating-point filters, can be necessary for geometric algorithms designed to only return approximate results. An excellent in-depth discussion about geometric robustness can be found in the report by Shewchuk [173]. The `ApproxMVBB` library uses the source code of the geometric predicates provided along with the work in [173]. Special care has been taken to ensure the implementation in [173] works with extended precision internal floating-point registers by using the source code in [171]. Also worth mentioning at this point is the famous Computational Geometry Algorithms Library `CGAL` [186] , which has the goal of providing efficient and reliable geometric algorithms in the form of a C^{++} library. Unfortunately at time of writing, the bounding volumes module in `CGAL` [53] is lacking an implementation of an approximate minimal volume bounding box algorithm in $\mathbb{E}^3$.

The library `ApproxMVBB` provides the following additional features:

- Convex hull of a two-dimensional point cloud (see the graham scan algorithm in [144]).
- Minimal area rectangle of a two-dimensional point cloud (see the rotating calipers algorithm in [144]).
- Efficient kd-tree (n-dimensional, generic) implementation with sophisticated splitting technique which optimizes the quality criterion in (10.18) (cf. section 10.3). It can also support other data structures than just n-dimensional points.
- Efficient k-nearest neighbor search in a point cloud in $\mathbb{E}^n$ by the help of a kd-tree (cf. algorithm 11.3).
- Fast statistical outlier removal in a point cloud in $\mathbb{E}^n$ with the help of a k-nearest neighbor search.

The Point Cloud Library `PCL` [163] is also worth mentioning. It provides various algorithms for processing point clouds arising in the field of robotics. Future work includes the integration of certain parts of the `ApproxMVBB` library into `PCL`.

The `ApproxMVBB` library provides its own kd-tree implementation, although there exists a wide variety of kd-tree implementations, such as the implementation in the Computational Geometry Algorithms Library `CGAL` [186], the Approximate Nearest Neighbor library `ANN` [121] or the Fast Library for Approximate Nearest Neighbor Search `FLANN` [124, 123, 122]. The reason for this is that our implementation on the one hand provides support for the domain decomposition by the flexible splitting process during construction (see section 10.3) and on the other hand provides efficient support for the outlier removal step discussed in section 11.1. Furthermore, it is difficult to define efficient interfaces to the aforementioned libraries such that existing data structures, such as the point cloud data structure, can be used directly without the need for inefficient copies. The library `nanoflann` [24] is very promising because its fast nearest neighbor queries can operate directly on user-defined data structures and is considered as a further improvement of the `ApproxMVBB` library. The kd-tree implementation in the library `ApproxMVBB` pursues similar paradigms and its genericness allows the direct use of user-provided data structures.

```
1  def optimizeBB(A,P,nLoops):
       Data: Input bounding box A of point cloud P in E^3, a list dirCache to save
             the last three directions and epsilon values ε_v, ε_d ≪ 1.
2      cacheIdx ← 0
3      for loops in range(0,nLoops):
4          d ← A.getDirection(0)                          ▷ get first normalized direction
5          for i in range(0,min(3, nLoops)):              ▷ compare to cache
6              dotp ← |d^⊤(dirCache.get(i))|
7              if |dotp − 1| ⩽ ε_d :                      ▷ direction almost the same as in cache
8                  d ← A.getDirection(1)                  ▷ choose another direction
9          dirCache.get(cacheIdx) ← d                     ▷ save to cache
10         cacheIdx ← (cacheIdx + 1)  mod 3               ▷ move cache write index
11         O ← computeMVBB(P, A, d)      ▷ compute B_opt(P, d), 3rd direction of O is d
12         if ε_v · A.volume() < O.volume() < A.volume():
13             A ← O                                      ▷ overwrite OBB with smaller one
14     return A
```

Algorithm 11.5: Principle of a simple minimal volume bounding box optimization by successively computing the optimal bounding box $B_{opt}(P, \mathbf{d})$ of a point cloud P for different directions $\mathbf{d} \in \mathbb{E}^3$.

The library `ApproxMVBB` has only a few dependencies, namely, the matrix-vector library `Eigen` [68] (at least version 3), the library `Meta` by Niebler [136] for sophisticated functional metaprogramming in C++11/14, the library `pugixml` by Kapoulkine [82] for XML export functionalities and the programming language `python` (at least version 3) for visualizing the examples and tests.

Chapter 12

Communication during Parallel Execution

This chapter is dedicated to the heart of the parallelization of large-scale multi-body systems, namely the communication among all processes in a process topology $\mathcal{T}$.

The concept and underlying implementation of exchanging information among processes is dependent on the parallelization technique used for a certain parallel application. If the parallelization of the simulation uses several threads or processes on a multi-core architecture, the exchange of information can be realized by exploiting the shared memory architecture (SMA) with certain synchronization mechanisms. Synchronization mechanisms, such as locks ,also called mutexes, or semaphores, ensure that processes follow a certain concurrency policy such that the integrity of the shared memory is guaranteed, for example, two processes may not write to the same shared data block at the same time. Similar synchronization paradigms also apply to the parallelization on graphic processing units (GPUs), which is at time of writting achieved by the CUDA or OpenCL technology. However, if the application runs on a distributed system, such as a high-performance cluster, the de facto standard communication technique is the so called Message Passing Interface MPI [106]. The Message Passing Interface is a standardized and portable message-passing system to function on a wide variety of parallel computers. The `GRS` framework mainly uses MPI as a top-level parallelization technique on a distributed system where multiple compute nodes consisting of several processors are connected over a high-speed network. Additional lower level parallelization techniques, especially GPU parallelization, can be added to increase the performance even more. Extensive work on GPU parallelization techniques, related to large-scale rigid body dynamics, has been conducted in [142, 183, 40]. In the following, we describe the communication during a parallel multi-body simulation with focus on the parallelization technique MPI.

The communication during a parallel rigid body simulation is directly coupled to the discretized integration scheme used for the equations of motion. In the case of the explicit Moreau time-stepping scheme described in section 8.1, the communication can be divided into two parts: the *body communication* and the *inclusion communication*. This separation is also possible in particular for related explicit time-stepping schemes where the integration on displacement level is clearly separated from the integration on velocity

level. The body communication operates on displacement level and is explained in more detail in the remaining part of this chapter. The second part, the inclusion communication, is responsible for the communication during the solution process of the contact problem and is described in chapter 13.

12.1 Sending and Receiving Messages

In the design of MPI's message passing model, there is the notation of a *communicator*. A *communicator* is a group of processes identified by their id, called *rank*, which have the ability to communicate with each other. The foundation of the message passing model is built upon send and receive operations. The process of sending and receiving messages with MPI is fully described in the weighty standard [106] or in more educational references such as [145] and [67]. To familiarize the reader to the MPI functionality, the resource [84] is highly recommended. We will in the following only describe some of the concepts of the actual send and receive implementations of the `ProcessCommunicator`[30], which provides a layer around several MPI functionalities. MPI's send and receive operations are classified into *collective* and *non-collective* routines. We will briefly elaborate on these two types in the following.

Non-collective MPI routines are point-to-point communications, where for example a process P_i sends a message to a process P_j. Each sent message is identified by a *tag*, that is, an integral value. Some process may submit a send operation with the *blocking* function call `MPI_Send` and another receiving process posts its matching receive operation with a *blocking* function call `MPI_Recv`. A matching receive means that in order to successfully receive the message, the sender *rank*, the message *tag*, the message type and the used *communicator* specified for the receive call needs to match with the specified values for the send call (exceptions apply, see [106]). A *blocking* send operation will not return till the message data has been stored away such that the sender can again freely access and overwrite the sent data. A *blocking* point-to-point send operation can be used in *standard*-, *synchronous*-, *buffered*- or *ready*-mode, that is, `MPI_Send`, `MPI_Ssend`, `MPI_Bsend` and `MPI_Rsend`, respectively. A *blocking* send operation in *standard* mode, that is, routine `MPI_Send`, is a *non-local* routine: successful completion may depend on an occurrence of a matching receive posted on another process. However, a *blocking* send operation in *standard* mode may return early if the message has been copied into an internal send buffer even if no matching receive has been posted. Internal buffering is dependent on the MPI implementation and not directly specified by the MPI standard. A *synchronous* send operation, that is, routine `MPI_Ssend`, is always *non-local* and can only successfully complete if a matching receive on the receiving process has been posted. For the rather exotic *ready*-mode, the reader is referred to [106]. The point-to-point communication functions `MPI_Send` and `MPI_Recv` also have a *non-blocking*, also called *asynchronous*, counterpart, that is, routine `MPI_Isend` and `MPI_Irecv`, which when called will return as soon as possible to allow for other computations while the sending or receiving operations are being processed in the background by light-weight threads. Of course the user should not access or modify the sent message data until the *non-blocking* operation `MPI_Isend` has finished.

Process P_1

```
1 MPI_Send(m1, P2)      ▷ may deadlock!
2 MPI_Recv(m2, P2)
```

⇓

```
1 MPI_Isend(m2,P2)   ▷ non-block. send
2 MPI_Irecv(m1,P2)   ▷ non-block. send
3 MPI_Waitall()      ▷ wait for completion
```

Process P_2

```
1 MPI_Send(m2,P1)
2 MPI_Recv(m1,P1)
```

⇓

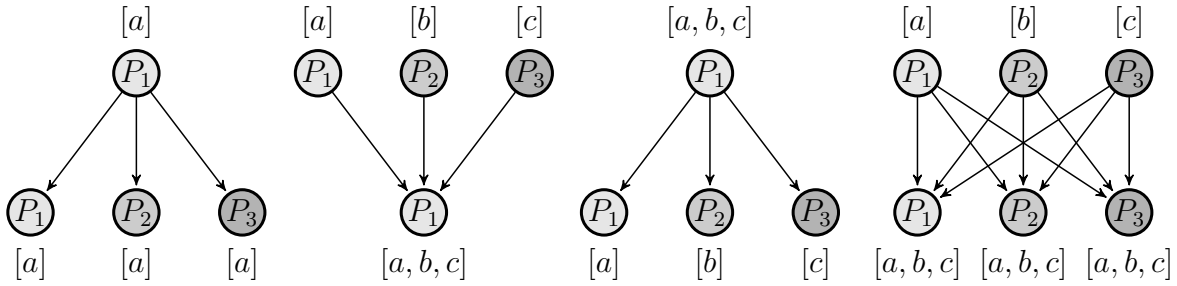
```
1 MPI_Isend(m2,P1)   ▷ non-block. send
2 MPI_Irecv(m1,P1)   ▷ non-block. send
3 MPI_Waitall()      ▷ wait for completion
```

Algorithm 12.0: An *unsafe* MPI program at the top with its *safe* counterpart at the bottom. If the MPI implementation dores not use internal buffering or the internal buffer is to small for receiving message m_1 on process P_2, the send operation on process P_1 will never return and thus the application deadlocks.

This can be achieved by waiting for completion by using the routine `MPI_Wait`.

As mentioned above, an MPI conform implementation may use internal buffers for send routines in *standard* mode. If an MPI application relies on internal buffering, it is called *unsafe*. An *unsafe* MPI application may run correctly if the MPI implementation uses internal buffering but may eventually deadlock if the implementation does not so. If a deadlock occurs or not is not Turing computable due to the Halting problem[1] and thus it is crucial to prevent deadlocks from the beginning. This can be done by writing *safe* MPI applications in the first place and by using deadlock-free parallel synchronization paradigms as discussed in [100, 66]. An example of a *safe* versus *unsafe* MPI program is shown in algorithm 12.0. The *safe* and deadlock-free implementation in algorithm 12.0 uses the non-blocking function calls `MPI_Isend` and `MPI_Irecv` in combination with a *blocking* call to `MPI_Waitall` to wait for the termination of the send and receive operations. A *collective* MPI routine, as the name suggests, is a communication routine in which all processes of a given communicator are involved. A *collective* routine combines sending and receiving operations into a high-level communication function. A common *collective* function is the broadcast function `MPI_Bcast` which distributes a message from a root process to all other processes in the *communicator*. Other functions include the *gathering* and *scattering* of information on and to a set of processes, which is achieved by the routines `MPI_Gather` and `MPI_Scatter`, respectively. Figure 12.1 visualizes these routines together with routine `MPI_Allgather`. The MPI standard defines several message data types, such as `MPI_INT`, `MPI_DOUBLE`, `MPI_BYTE` etc., to be used with the send and receive routines. MPI also provides functionalities to define aggregates of these data types by routines such as `MPI_Type_contignuous` or `MPI_Type_struct`, for example. Sending a message at the user-level in general needs one copy to encode the message data, encompassing data from different data structures in an application, into a user-defined buffer of some MPI data type, such as an aggregate of types `MPI_INT` and `MPI_DOUBLE`. Receiving a message might even take two copies: one to copy the received data into the user-provided buffer (e.g., `MPI_BYTE`) and another copy to decode the message back into the data structures used in the application. If a message of arbitrary length needs to

[1] See Wikipedia: https://en.wikipedia.org/wiki/Halting_problem



(a) `MPI_Bcast` **(b)** `MPI_Gather` **(c)** `MPI_Scatter` **(d)** `MPI_Allgather`

Figure 12.1: Behavior of four *collective* MPI routines for 3 processes P_1, P_2 and P_3 using messages a, b and c, respectively.

be sent multiple times and the shape of the data stays the same, then the *skeleton & content* approach implemented in [26] is beneficial. Sending and receiving dynamic-sized messages, where the shape and content of the message highly varies, is realized with a *serialization & deserialization* approach in the `GRS` framework. Serialization is the process of translating data structures into a continuous data buffer, for example a memory buffer or a file, such that it can be transmitted over the network. Deserialization is the opposite process, namely the reconstruction of the data structure from the continuous data buffer. Serialization and deserialization of messages in the `GRS` framework is implemented by using the `Boost` library [26, 167]. The implementation in `MPIMessages`[31] encompasses serialization/deserialization wrappers for all messages sent during the execution of the application `GRSFSimMPI`. The major advantage of serializing a message into a continuous memory block is to be able to send messages of arbitrary size and shape over the network. The serialization implementation in the Boost C^{++} library is extremely versatile and also deals with byte ordering, called *endianness*. At the time of writing, byte ordering is not considered for the serialization and deserialization as the parallel application `GRSFSimMPI` is assumed to run on a cluster with a homogeneous architecture. The next section covers the aforementioned body communication in detail.

12.2 Body Communication

As mentioned in section 10.1, a body B_i is assigned to its belonging process domain according to its center of gravity $\mathbf{r}_{S_i}$. During the simulation of a multi-body system, rigid bodies may change their belonging process domain multiple times. Therefore, during the whole simulation, all processes need to maintain consistent information on which bodies they do or do not own and on which bodies do or do not overlap their process domain. This is crucial for a correct collisions detection and to correctly solve the corresponding contact problem in a later step. The body communicator of a process P_i in `BodyCommunicator`[28] is responsible for maintaining this information by communicating with its neighborhood N_i, that is, all adjacent neighbors of process P_i. The communication for the case of Moreau's time-stepping scheme is performed by the `BodyCommunicator`[28] after the first half-time step in step 1 and before the collision detection in step 2 in

algorithm (8.1).

The consistency of information among all processes is of tremendous importance since missing or wrong information quickly leads to wrong numerical computations. Consistency at the bottom-level includes the geometric correctness of the required routines `belongsBodyToProcess` and `checkOverlapLocal` described in section 10.1. Situations such as bodies suddenly belonging to two process domains or falsely removed bodies in processes should not occur in all circumstances. Achieving consistency of all exchanged information therefore requires the implementation of a sophisticated book-keeping procedure and the design of corresponding data structures. The following explanations cover some of the combinatoric issues involved in the book-keeping procedure in the body communication part and should support the understanding of the implementation in `BodyCommunicator`[28]. The body communication was greatly influenced by the extensive work [75] for the rigid body engine *pe* [74] developed at the university of Erlangen. Differences to their implementation will be pointed out in the remaining part of this section.

Each process P_i maintains a set of *local* bodies $B_{P_i,loc}$ and a set of *remote* bodies $B_{P_i,rem}$. A body B_i is said to be *local*, that is, $B_i \in B_{P_i,loc}$), if it is owned by process P_i, meaning that its center of gravity $\mathbf{r}_{S_i}$ is contained in the process domain P_i. A *remote* body $B_i \in B_{P_i,rem}$ is the exact opposite, namely, it is not owned by process P_i and its geometry overlaps the process domain P_i. Only local bodies, that is, bodies in the set $B_{P_i,loc}$, take part in the time-stepping procedure and are updated exclusively by process P_i. In contrary, all remote bodies, that is, bodies in the set $B_{P_i,rem}$, are updated exclusively by received messages during the communication stage from other neighboring processes owning these bodies. The reader should note that $\bigcup_{i=1}^{p} B_{P_i,loc}$ yields the set containing all simulated bodies by the unique owner requirement in section 10.1. In this way, the task of a body communicator in process P_i at time t_k is to send and receive messages from and to the neighboring processes $P_n \in N_i$ to guarantee that its own set of remote bodies $B_{P_i,rem}$ and the sets $B_{P_n,rem}$ of all neighbors $P_n \in N_i$ are consistent and their contained body data is up-to-date. A message sent by process P_i to a neighbor $P_n \in N_i$ contains update information for local bodies in $B_{P_i,loc}$ which are classified as remote bodies in the neighbor process P_n. Vice versa, a message received by process P_i sent from a neighbor $P_n \in N_i$ contains update information for remote bodies in the set $B_{P_i,rem}$ of process P_i, which are owned by neighbor process P_n and thus in its set $B_{P_n,loc}$.

One single body communication message sent by the `BodyCommunicator`[28] of process P_i to a neighbor P_n contains basically a number of sub-messages, where each sub-message corresponds to one body and is either a *body notification*, a *body update* or a *body removal* message. These three types of sub-messages encoded in a single body communication message sent at time t_k are explained briefly in the following:

- **Notification Message:** If a body B_i, owned by process P_i at time t_k, starts overlapping a neighbor process P_n at time t_k, process P_i sends a notification message to process P_n.
- **Update Message:** If a body B_i, owned by process P_i at time t_k, is at time t_k overlapping a neighbor process P_n which has already been informed by a notification message about body B_i at the previous communication time point t_{k-1}, process P_i

sends an update message to process P_n.

- **Removal Message:** If a body B_i, owned by process P_i at time t_k, becomes a non-remote body in neighbor process P_n at time t_k, process P_i sends a removal message to process P_n.

For a better understanding of the communication procedure, figure 12.2 visualizes all possible communication patterns between two processes P_1 and P_2 demonstrated with only one single body B_i at an arbitrary time t_k. Figure 12.2 shows three possible situations for sending a notification message, denoted by $A(t_k)$, $B(t_k)$ and $C(t_k)$, three possible situations for sending an update message, denoted by $D(t_k)$, $E(t_k)$ and $F(t_k)$, and one possible situation for sending a removal message, denoted by $G(t_k)$. The arrows in figure 12.2 visualize the fictitious displacement of body B_i from time t_{k-1} to time t_k. The description at the bottom of all seven situations in figure 12.2 describes the classification of body B_i in process P_1 before and after sending and in process P_2 before and after receiving the message. In situation $F(t_k)$ in figure 12.2, for example, process P_1 sends an update message for body B_i to process P_2 since body B_i was already overlapping at time t_{k-1} and is thus classified as a remote body in process P_2 and classified as a local body in process P_1. After sending and receiving the message, process P_1 needs to delete body B_i from its local set $B_{P_1,loc}$ because body B_i is no more overlapping process P_1 and is solely owned by process P_2 which consequently moves body B_i from its set of remote bodies $B_{P_2,rem}$ to the its set of local bodies $B_{P_2,loc}$. All other situations in figure 12.2 can be understood analogously.

The transition diagram shown in figure 12.3 for all sending states of process P_1 can be obtained by tracking the communication states $A(t_k)$ to $G(t_k)$ of body B_i over a time interval. All black transitions in figure 12.3 are time step updates where time t_k is updated by the new time t_{k+1}. Note that the sending state transitions in figure 12.3 applies to all local bodies in any process and are used for the implementation of the message `NMessageWrapperBodies`[32] used by the `BodyCommunicator`[28].

The communication algorithm for exchanging information among processes in the neighborhood N_i of a process P_i is demonstrated in algorithm 12.2 and is based on the work [75]. The reason for explicitly mentioning algorithm 12.2, is that a correct deadlock-free implementation of such an information exchange is not simple for users being new to MPI. The algorithm 12.2 is generic and *safe* in MPI terminology and thus does not rely on buffering. Not relying on buffering is achieved in algorithm 12.2 by first posting all send operations (line 2) of all messages by the *non-blocking* call `MPI_Isend`. After that, the algorithm directly proceeds to receiving all message by a polling loop over all processes from which a message has not yet been received (line 9). The function `MPI_Iprobe` checks if a message with tag *tag* from process with rank n in communicator *comm* is available. If so, the size of the message is determined (line 13) and the receiving archive *rar* is resized (line 14) and prepared to receive the message from rank n (line 15). The message is then received and written into the binary archive *rar* by the standard *blocking* function call `MPI_Recv` in line 17. The binary archive *rar* is then deserialized in line 18. After all messages have been received, all send operations may still be incomplete and need to be waited for by using the function `MPI_Waitall` in line 21, which when complete marks the end of the

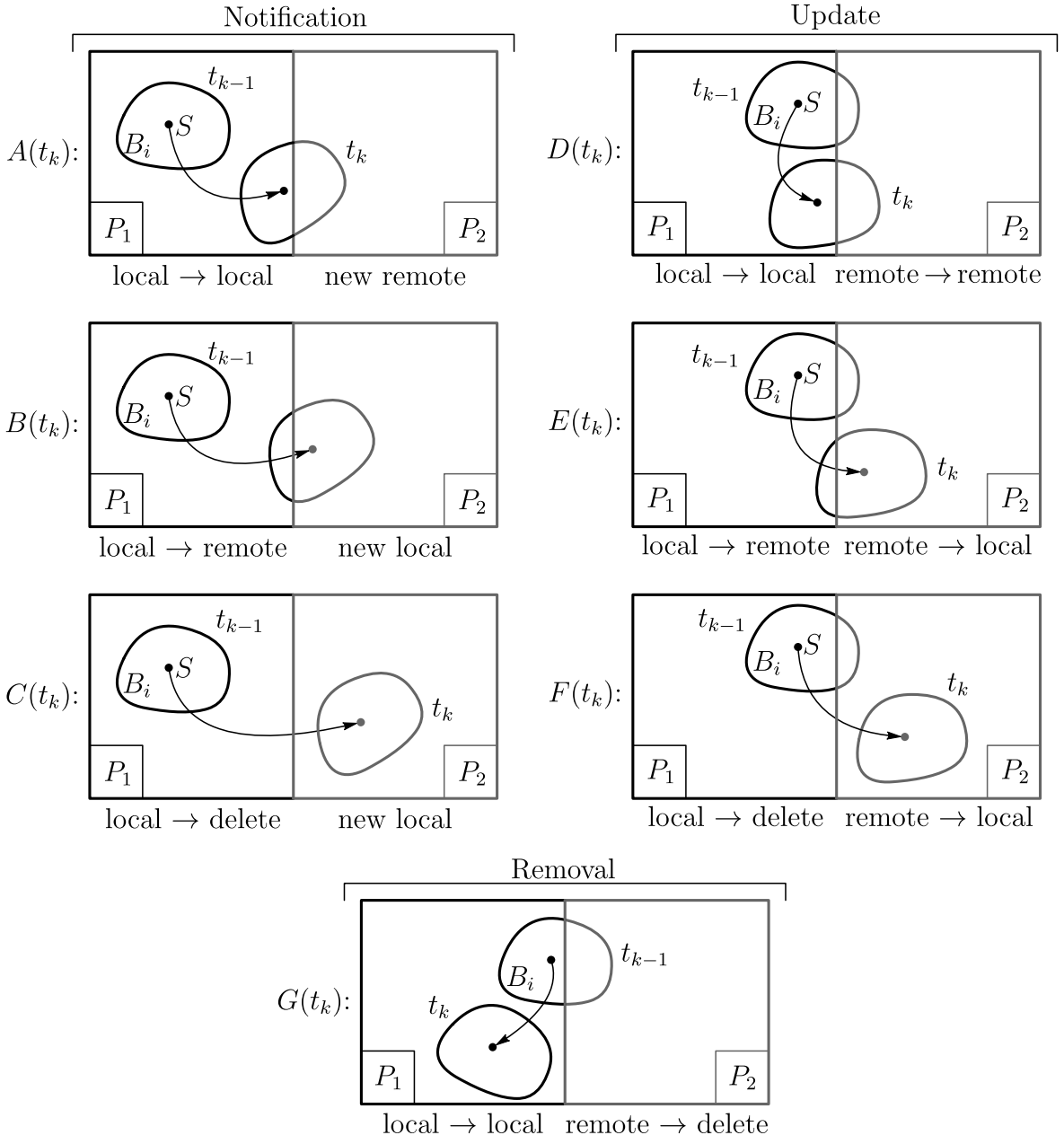


Figure 12.2: A visualization of all communication types between process P_1 (black rectangular domain) and P_2 (gray rectangular domain) caused by the displacement of a body B_i from time t_{k-1} to time t_k. In the situations A, B and C, a notification message is sent to process P_2. In the situations D, E and F, an update message is sent to process P_2. The removal message is denoted in situation G. The description below each situation denotes the change in the classification of body B_i before and after sending/receiving the message.

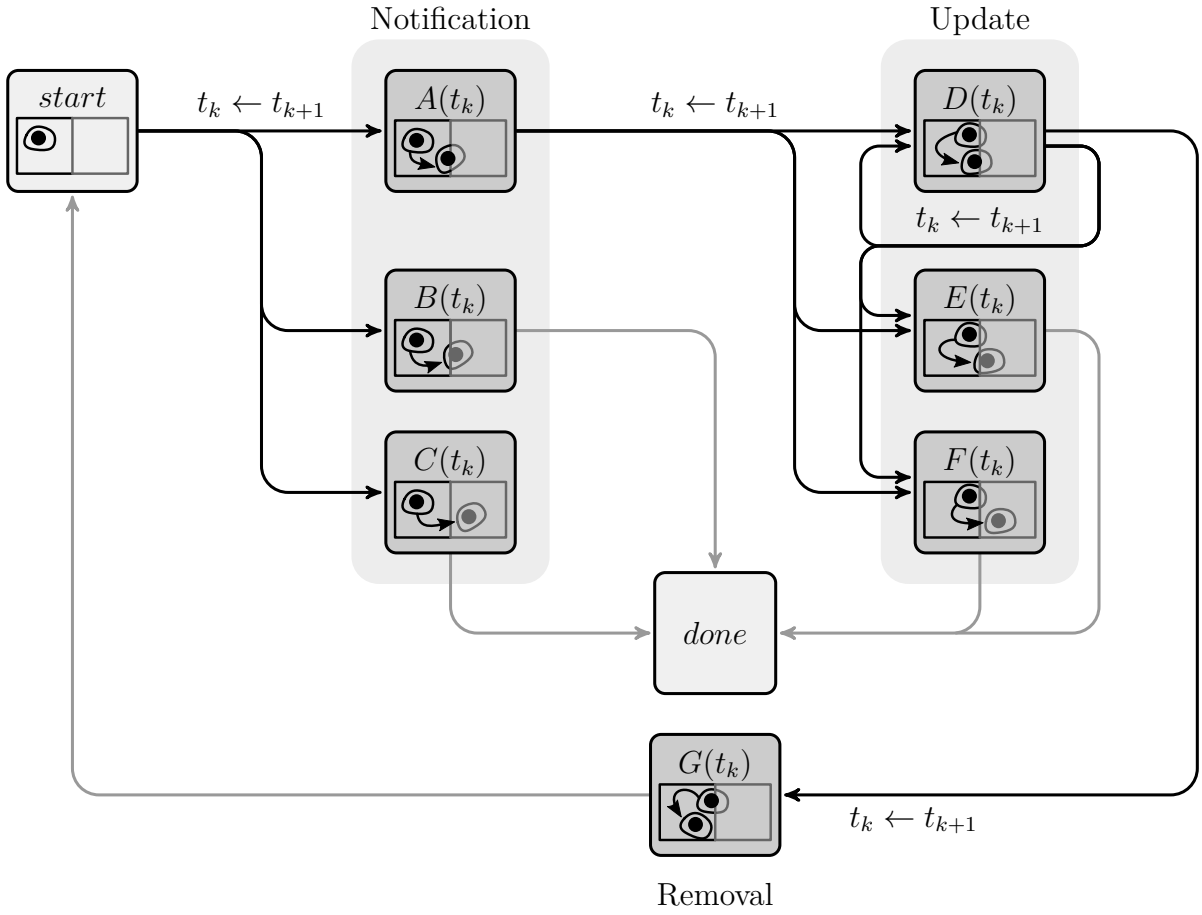


Figure 12.3: Visualization of all possible transitions between the different communication states shown in figure 12.2 for a process P_1 and a body B_i. The state *start* marks the situation at time t_k where body B_i is solely contained in process P_1 and does not overlap any process. The state *done* marks the termination of sending submessage for B_i to P_2. All black transitions $t_k \leftarrow t_{k+1}$ update the time t_k to the new time t_{k+1}.

information exchange. Its worth noting that the message object *msg* in algorithm 12.2 is indentical to `NMessageWrapperBodies`[32] and that the MPI datatype of the archives *sar* and *rar* is `MPI_BYTE`.

Algorithm 12.2 builds the central part of the communication in the `GRS` framework and is used not only in the `BodyCommunicator`[28], but also in the `InclusionCommunicator`[29].

The main difference between the rigid body engine *pe* [74] is that the `GRS` framework does not use any kind of overlap region as presented in [75] where the separating boundary faces between processes are thickened up by a value dx. The *pe* rigid body engine starts sending notification or update messages to a neighbor P_j for a body B_k owned by P_i when B_k enters the overlap region between P_i and P_j. Such an overlap region introduces an additional vague parameter dx whose value needs to be chosen in reasonable bounds in accordance to the given extents of the simulated rigid body geometries. However, it is sufficient to consider only the sole boundary faces between a process and its neighbors,

```
1  def exchangeInformationAsync(msg,tag,comm):
       Data: A list sArchs of binary archives for all sent neighbor messages. A set N_i
             of all neighbor processes of process P_i with rank i. A binary archive rar
             for the received messages.
2      for P_n in N_i:
3          sar ← sArchs.getArchive(n)
4          msg.setRank(n)              ▷ prepare the message to be serialized for process P_n
5          sar ≪ msg                   ▷ serialize message msg into the binary archive sar
6          MPI_Isend(sar.data(), sar.size(), sar.mpiDataType(),
7                    i, tag, comm, sar.request)     ▷ send binary archive to process P_n
8      pL ← [ P_n for P_n in N_i ]
9      while ¬pL.empty():
10         for P_n in pL:
11             flag, status ← MPI_Iprobe(n, tag, comm)    ▷ status of message from P_n.
12             if flag:
13                 size ← MPI_Get_count(status, rar.mpiDataType())
14                 rar.resize(size)          ▷ resize the archive to receive the binary data
15                 msg.setRank(status.source())          ▷ prepare deserialization for P_n
16                 MPI_Recv(rar.data(),rar.size(),rar.mpiDataType(),
17                          status.source, tag, comm)
18                 rar ≫ msg                 ▷ deserialize the binary archive rar by msg
19                 pL.remove(P_i)
20     requests ← [ sar.request for sar in sArchs ]      ▷ put all send requests into a list
21     MPI_Waitall(requests.size(), requests, ...)        ▷ wait for all send operations
```

Algorithm 12.2: Sending and receiving a message *msg* with tag *tag* to and from the neighborhood N_i of process P_i in a given communicator *comm*.

exactly as shown and explained in this section, to successfully implement a correct body communication. Therefore, an additional overlap region appeared to be impracticable and has not been used in the GRS framework.

Chapter 13

The Mass Splitting Method

In this chapter, we review the mass-splitting procedure to solve the non-smooth contact problem of a domain-decomposed assembly of rigid bodies. The mass-splitting procedure is independent of the used process topology and any domain decomposition approach in $\mathbb{E}^3$ as the ones described in chapter 10 can be used.

In each time step of a rigid body simulation, the collision detection results in a contact graph G in $\mathbb{E}^3$ which involves all contacting bodies B_i. The spatial domain of a contact graph G does generally overlap the domains of the process topology $\mathcal{T}$. Splitting a body overlapping several process domains at a fixed time t into multiple virtual geometrically identical copies with a smaller mass than the original body is the simple yet powerful key component of the mass-splitting procedure. Additionally, bilateral constraint forces then literally glue all copies together such that the coalesced assembly of all virtual bodies still represents the original body. In other words, bilateral constraints enforce that all copies have the same displacement and velocity as the original body. This mass-splitting results in replacing all domain overlapping edges on the split body in G with several bilateral constraint nodes. This is convenient for dividing G into distinct subgraphs G_k within each process domain of the process topology $\mathcal{T}$ which is essential to efficiently compute the solution of the inclusion problem in (8.1).

The remainder of this chapter explains the consequences of the mass-splitting procedure more mathematically by often referring to figure 13.1 which visualizes a contact graph of a rigid body assembly in $\mathbb{E}^2$ where the process topology $\mathcal{T}$ consists of 3 rectangular process domains.

13.1 Splitting a Domain-Overlapping Body

The equations of motion for one rigid body B_i with generalized displacement $\mathbf{q}_{B_i} \in \mathbb{R}^{n_q}$ and velocity $\mathbf{u}_{B_i} \in \mathbb{R}^{n_u}$ is given by the kinetic and kinematic part in equations (5.42)

and (5.43). Together with set-valued contact laws, this yields

$$\begin{aligned} \dot{\mathbf{q}}_{B_i} &= \mathbf{F}(\mathbf{q}_{B_i}, t)\ \mathbf{u}_{B_i} + \boldsymbol{\beta}(\mathbf{q}_{B_i}, t)\ , \\ \mathbf{M}_{B_i}\ \dot{\mathbf{u}}_{B_i} - \mathbf{h}_{B_i}(\mathbf{q}_{B_i}, \mathbf{u}_{B_i}, t) &- \sum_{k \in C_{B_i}} \mathbf{W}_k(\mathbf{q}_{B_i}, t)\boldsymbol{\lambda}_k = \mathbf{0}\ , \end{aligned} \tag{13.1}$$

$$\left.\begin{aligned} \boldsymbol{\gamma}_k &= \mathbf{W}_k^\top \mathbf{u}_{B_i} + \boldsymbol{\chi}_k \\ (\boldsymbol{\gamma}_k, \boldsymbol{\lambda}_k) &\in \mathcal{S}_k \end{aligned}\right\} \quad \forall k \in C_{B_i} \subseteq \mathcal{I}(\mathbf{q}, t)\ . \tag{13.2}$$

The index set C_{B_i} in (13.2) refers to all closed contacts k on body B_i, which is a subset of all closed contacts $\mathcal{I}(\mathbf{q}, t)$ in the mechanical system. Without loss of generality, it is assumed that body B_i overlaps p process domains $\{P_1, \ldots, P_p\}$ at time t and belongs to domain P_1.

At time t, body B_i is now split into p virtual copies $\{V_1, \ldots, V_p\}$, each with the exact same geometry such that each domain P_j, which body B_i overlaps, has its own virtual copy V_j. Body B_i is then said to have a *multiplicity factor* of p. Body B_i in figure 13.1, for example, belongs to domain P_1 and overlaps the neighbor process domains P_1 and P_2 and therefore has multiplicity factor 3.

Each virtual copy V_j has the same displacement as the orginal body, that is, $\mathbf{q}_{V_j} = \mathbf{q}_{B_i}\ \forall j \in [1\,;\,p]$, but different velocity $\mathbf{u}_{V_i}$, different constant mass matrix $\mathbf{M}_{V_j} \in \mathbb{R}^{n_u \times n_u}$ and different nonlinear term $\mathbf{h}_{V_j}(\mathbf{q}_{B_i}, \mathbf{u}_{V_i}) \in \mathbb{R}^{n_u}$, where the latter two are not yet known and conditions on these quantities are developed in the following.

The next step is to find a unique distribution of all contacts k on the original body B_i onto the p virtual copies $\{V_1, \ldots, V_p\}$. This choice is arbitrary and from a collision solving perspective, it makes most sense to assign a contact k to the copy V_j in process domain P_j where it has been detected, such that $C_{B_i} = \bigcup_{j=0}^{p} C_{V_j}$. In figure 13.1, for example, body V_1 contains the set of all contacts detected in domain P_1, that is, $C_{V_1} = \{C_1, C_2\}$.

Consequently, each virtual copy V_j has its own equation of motion written as

$$\mathbf{M}_{V_j}\ \dot{\mathbf{u}}_{V_j} - \mathbf{h}_{V_j}(\mathbf{q}_{B_i}, \mathbf{u}_{V_j}, t) - \sum_{k \in C_{V_j}} \mathbf{W}_k(\mathbf{q}_{B_i}, t)\boldsymbol{\lambda}_k = \mathbf{0} \qquad \forall j \in [1\,;\,p]\ . \tag{13.3}$$

To make the description of the new mechanical system including the p virtual bodies consistent with the original equation of motion (13.1), bilateral constraints on velocity level are introduced to enforce the equality of velocity of all virtual rigid bodies at time t, that is,

$$\begin{aligned} \mathbf{s}_m \in \mathbb{R}^{n_u} \quad \wedge \quad \boldsymbol{\gamma}_{B_i,m}(\mathbf{u}_{V_m}, \mathbf{u}_{V_n}) &= \mathbf{u}_{V_m} - \mathbf{u}_{V_n} \overset{!}{=} \mathbf{0} \\ &= \underbrace{\begin{bmatrix} \mathbf{I} & -\mathbf{I} \end{bmatrix}}_{\overline{\mathbf{W}}_m{}^\top} \begin{bmatrix} \mathbf{u}_{V_m} \\ \mathbf{u}_{V_n} \end{bmatrix}, \quad \forall m \in [1\,;\,p-1] : n = m+1, \end{aligned} \tag{13.4}$$

where the vector $\mathbf{s}_m$ represents the bilateral force of constraint m. All linearly independent force directions $\overline{\mathbf{W}}_m \in \mathbb{R}^{2n_u \times n_u}$ are assembled as $\overline{\mathbf{W}} \coloneqq \left[\overline{\mathbf{W}}_1, \ldots, \overline{\mathbf{W}}_{p-1}\right]$, the same for the relative velocities, that is, $\boldsymbol{\gamma}_{B_i} = \left[\boldsymbol{\gamma}_{B_i,1}{}^\top, \ldots, \boldsymbol{\gamma}_{B_i,p-1}{}^\top\right]^\top$. If another bilateral constraint is

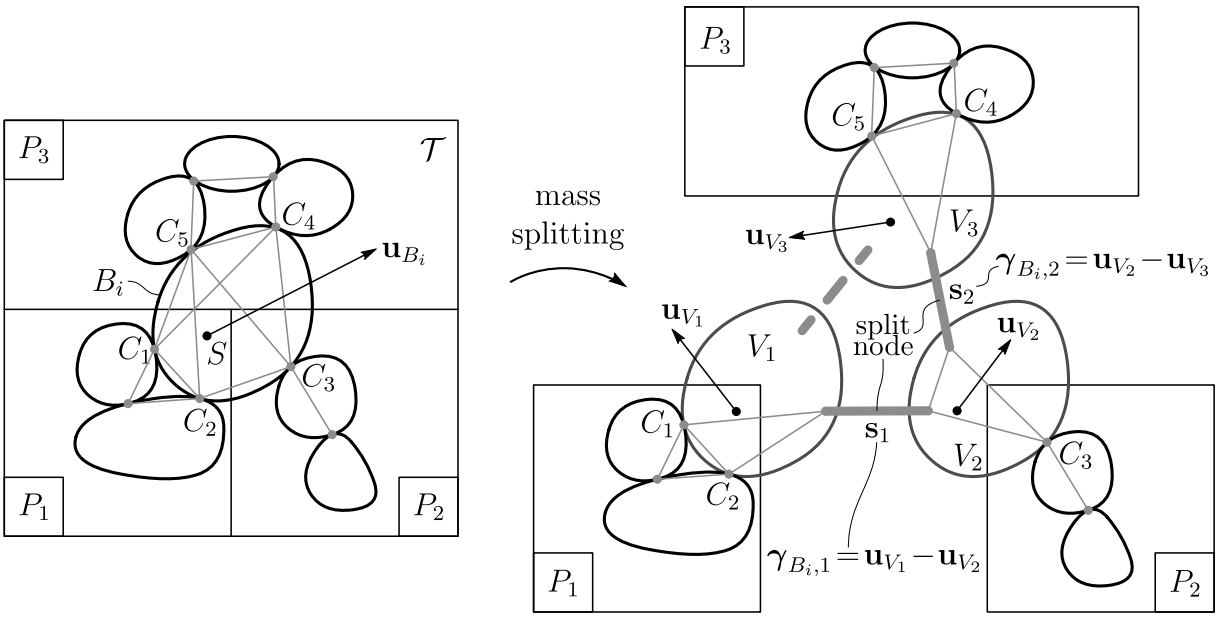


Figure 13.1: Visualization of the mass-splitting procedure. Left: The process topology consists of three process domains P_1, P_2 and P_3 over which several bodies are spread and form a contact problem with 5 contacts $C_1, C_2, \ldots, C_5$ on the body of interest B_i. Right: The same problem is shown but with three virtual bodies V_1, V_2 and V_3 with the same geometry and position but different velocities. Two bilateral constraints (long gray bars) form one *split-node* which enforces the equality of velocity among all virtual bodies. The contact graph is nicely split into subgraphs for each process domain.

included such that $\mathbf{u}_{V_1} - \mathbf{u}_{V_p} = \mathbf{0}$, then the directions $\{\mathbf{W}_1, \ldots, \mathbf{W}_{p-1}, \mathbf{W}_p\}$ will become linearly dependent. In figure 13.1, that additional redundant constraint is shown as a long dashed gray bar. The combined equation of motion of the p virtual bodies V_j expressed with the principle of virtual power in (5.63) yields

$$\underline{\delta} P = \delta \mathbf{u}^\top \left(\mathbf{M}\, \dot{\mathbf{u}} - \mathbf{h}(\mathbf{q}, \mathbf{u}, t) - \mathbf{f}_C(\mathbf{q}, t) - \overline{\mathbf{W}}\mathbf{s} \right) = 0 \quad \forall \delta \mathbf{u} \;, \tag{13.5}$$

$$\boldsymbol{\gamma}_{B_i} \in \mathcal{N}_{\mathbb{R}^{(p-1)n_u}}(\mathbf{s}) \;, \tag{13.6}$$

$$\left.\begin{array}{c} \boldsymbol{\gamma}_k = \mathbf{W}_k^\top \mathbf{u}_{V_l} + \boldsymbol{\chi}_k \\ (\boldsymbol{\gamma}_k, \boldsymbol{\lambda}_k) \in \mathcal{S}_k \end{array}\right\} \quad \forall k \in C_{V_l} \quad \forall l \in [1\,;\, p], \tag{13.7}$$

where the following definitions are used throughout the remainder of this chapter:

$$\mathbf{q} \coloneqq \begin{bmatrix} \mathbf{q}_{B_i} \\ \vdots \\ \mathbf{q}_{B_i} \end{bmatrix}, \; \mathbf{u} \coloneqq \begin{bmatrix} \mathbf{u}_{V_1} \\ \vdots \\ \mathbf{u}_{V_p} \end{bmatrix}, \; \mathbf{s} \coloneqq \begin{bmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_{p-1} \end{bmatrix}, \; \mathbf{M} \coloneqq \begin{bmatrix} \mathbf{M}_{V_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{V_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{M}_{V_p} \end{bmatrix}, \tag{13.8}$$

$$\mathbf{h}(\mathbf{q}, \mathbf{u}, t) \coloneqq \begin{bmatrix} \mathbf{h}_{V_1}(\mathbf{q}_{B_i}, \mathbf{u}_{V_1}, t) \\ \vdots \\ \mathbf{h}_{V_p}(\mathbf{q}_{B_i}, \mathbf{u}_{V_p}, t) \end{bmatrix}, \; \mathbf{f}_C(\mathbf{q}, t) \coloneqq \begin{bmatrix} \sum_{k \in C_{V_1}} \mathbf{W}_k(\mathbf{q}_{B_i}, t) \boldsymbol{\lambda}_k \\ \vdots \\ \sum_{k \in C_{V_p}} \mathbf{W}_k(\mathbf{q}_{B_i}, t) \boldsymbol{\lambda}_k \end{bmatrix}.$$

The $p-1$ linearly independent constraints in (13.4) with constraint forces $\mathbf{s}$ eliminate $p-1$ velocities out of $\{\mathbf{u}_{V_1}, \ldots, \mathbf{u}_{V_p}\}$. This is a well known reduction step to a new set of minimal velocities with respect to the *perfect* constraints in (13.4) consisting only of the velocity $\mathbf{u}_{B_i}$ and the transformation yields

$$\mathbf{u} = \begin{bmatrix} \mathbf{I} & \cdots & \mathbf{I} \end{bmatrix}^\top \mathbf{u}_{B_i} = \mathbf{Q}\mathbf{u}_{B_i} \ , \quad \Rightarrow \quad \delta\mathbf{u} = \mathbf{Q}\delta\mathbf{u}_{B_i} \ . \tag{13.9}$$

Transforming the variational formulation in (13.5) with (13.9) yields for all $\delta\mathbf{u}_{B_i}$

$$\delta\mathbf{u}_{B_i}{}^\top \left[\sum_{l=1}^{p} \mathbf{M}_{V_l}\dot{\mathbf{u}}_{B_i} - \sum_{l=1}^{p} \mathbf{h}_{V_l}(\mathbf{q}_{B_i}, \mathbf{u}_{B_i}, t) - \underbrace{\sum_{l=1}^{p}\sum_{k\in C_{V_l}} \mathbf{W}_k(\mathbf{q}_{B_i}, t)\boldsymbol{\lambda}_k}_{\sum_{k\in C_{B_i}}} - \underbrace{\mathbf{Q}^\top\overline{\mathbf{W}}}_{=\mathbf{0}}\mathbf{s} \right] = 0 \ . \tag{13.10}$$

The evaluation of the variational formulation (13.10) and the equation of motions in (13.1) are identical if the following conditions are fulfilled:

$$\mathbf{M}_{B_i} \stackrel{!}{=} \sum_{l=1}^{p} \mathbf{M}_{V_l} \ , \qquad \mathbf{h}_{B_i}(\mathbf{q}_{B_i}, \mathbf{u}_{B_i}, t) \stackrel{!}{=} \sum_{l=1}^{p} \mathbf{h}_{V_l}(\mathbf{q}_{B_i}, \mathbf{u}_{B_i}, t) \ . \tag{13.11}$$

This shows that the mass-splitting is completely arbitrary as long as the above simple conditions are fulfilled. In the following, we choose a simple scalar convex combination of the original terms given as

$$\boxed{\begin{gathered} \mathbf{M}_{V_l} \coloneqq \alpha_l \mathbf{M}_{B_i}, \quad \mathbf{h}_{V_l}(\mathbf{q}_{B_i}, \mathbf{u}_{V_l}, t) \coloneqq \alpha_l \mathbf{h}_{B_i}(\mathbf{q}_{B_i}, \mathbf{u}_{V_l}, t) \\ \text{with:} \ \sum_{l=1}^{p} \alpha_l = 1, \quad \alpha_l \geqslant 0 \quad \forall l \in [1\,;\,p] \ . \end{gathered}} \tag{13.12}$$

As can be seen in figure 13.1, the mass-splitting procedure results in 3 bilateral constraints which act as in-between nodes to couple all local contact graphs in each process domain together. The collection of all bilateral constraints belonging to one split body B_i in the contact graph is called *split-node*. The next section explains the strategy to solve a split-node analytically.

13.2 Direct Iterative Solution of a Split-Node

In the following, we are interested how the velocity $\mathbf{v}_{B_i}$ is determined from all virtual body velocities $\mathbf{v}_{V_j}$ once the constraint (13.4) is enforced and all other contact forces are known. The question arises if the procedure is as simple as averaging all virtual velocities, that is, $\mathbf{v}_{B_i} = \frac{1}{p}\sum_{i=1}^{p} \mathbf{v}_{V_i}$? To forestall the derivation in the next section, it turns out that, at least for the simplest splitting choice, averaging the velocities is the intuitive correct solution. In the following, the analytical solution for one split-node which belongs to an arbitrary body B_i is described within the framework of Moreau's time-stepping method in algorithm 8.1.

As mentioned before, a split-node for a body B_i with multiplicity p consists of $p-1$ bilateral constraints written in the form:

$$\boldsymbol{\gamma}_{B_i} = \overline{\mathbf{W}}^{\top}\mathbf{u} \in \mathcal{N}_{\mathbb{R}^{(p-1)n_u}}(\mathbf{s}) \quad \Leftrightarrow \quad \boldsymbol{\gamma}_{B_i} = \mathbf{0} \quad \wedge \quad \mathbf{s} \in \mathbb{R}^{(p-1)n_u} \ . \tag{13.13}$$

During a time-stepping procedure at time t, all constructed split-nodes are used in the inclusion solving step together with all other contact nodes from detected collisions. For Moreau's time-stepping scheme explained in section 8.1, the bilateral constraint in (13.13) is formulated at the end time t_E which results in

$$\boldsymbol{\gamma}^{E}_{B_i} = \overline{\mathbf{W}}^{\top}\mathbf{u}_E = \overline{\mathbf{W}}^{\top}\big(\mathbf{u}^S + \Delta t\mathbf{M}^{-1}\mathbf{h} + \mathbf{M}^{-1}\mathbf{f}_C + \mathbf{M}^{-1}\overline{\mathbf{W}}\mathbf{S}\big) = \mathbf{0}, \tag{13.14}$$

where the terms $\mathbf{h}$ and $\mathbf{f}_C$ are evaluated at the midpoint t^M with $\mathbf{q}^M$ and $\mathbf{u}^S$, the vector $\mathbf{S}$ denotes the bilateral percussion corresponding to the generalized force $\mathbf{s}$ and $\mathbf{f}_C(\mathbf{q}^M, t)$ is identical to definition (13.8) except that the contact forces $\boldsymbol{\lambda}_k$ are now understood as percussions $\boldsymbol{\Lambda}_k$.

If the iterative velocity SOR Prox scheme in (8.37) and (8.38) is applied to the inclusion (13.13) then (13.14) becomes

$$\boldsymbol{\gamma}^{E,s}_{B_i} = \overline{\mathbf{W}}^{\top}\mathbf{u}^{E,s} = \mathbf{0} \tag{13.15}$$

$$\text{with:} \begin{bmatrix} \mathbf{u}^{E,s} \coloneqq \mathbf{u}^{E,s}_C + \mathbf{M}^{-1}\overline{\mathbf{W}}\mathbf{S}^{\eta}, \\ \mathbf{u}^{E,s}_C \coloneqq \mathbf{u}^S + \Delta t\mathbf{M}^{-1}\mathbf{h}(\mathbf{q}^M, \mathbf{u}^S, t^M) + \mathbf{M}^{-1}\mathbf{f}^{\eta}_C(\mathbf{q}^M, t^M) \ , \end{bmatrix} \tag{13.16}$$

where η and s denote the percussion and velocity update counters, respectively (see (8.37) and (8.38)). The end time velocity $\mathbf{u}^{E,s}_{V_j,C}$ at step s of body V_j contains only the percussive contribution

$$\sum_{k \in C_{V_j}} \mathbf{W}_k(\mathbf{q}^M_{B_i}, t^M)\boldsymbol{\Lambda}^{\eta}_k \tag{13.17}$$

of the contacts in $C_{V_j} \subseteq G_j$ on body V_j, which belongs to process domain P_j.

Supposed that $\mathbf{u}^{E,s}_C$ is known at iteration (η, s) of the velocity SOR Prox iteration in (8.37) and (8.38), equation (13.15) suggests to directly solve for the percussion $\mathbf{S}^{\eta}$, that is,

$$\mathbf{S}^{\eta} = -\overline{\mathbf{G}}^{-1}\overline{\mathbf{W}}^{\top}\mathbf{u}^{E,s}_C, \quad \overline{\mathbf{G}} \coloneqq \overline{\mathbf{W}}^{\top}\mathbf{M}^{-1}\overline{\mathbf{W}} \in \mathbb{R}^{(p-1)n_u \times (p-1)n_u}, \quad \overline{\mathbf{G}}^{\top} = \overline{\mathbf{G}} \tag{13.18}$$

and substituting back into (13.15) to get

$$\mathbf{u}^{E,s} = \left(\mathbf{I} - \mathbf{M}^{-1}\overline{\mathbf{W}}\,\overline{\mathbf{G}}^{-1}\overline{\mathbf{W}}^{\top}\right)\mathbf{u}^{E,s}_C. \tag{13.19}$$

Equation (13.19) is meaningful since the symmetric Delassus matrix $\overline{\mathbf{G}}$ is positive definite and thus invertible since $\overline{\mathbf{W}}$ only contains linearly independent columns. The matrix $\overline{\mathbf{G}}$ and $\overline{\mathbf{W}}$ can be written as

$$\overline{\mathbf{W}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ -\mathbf{I} & \mathbf{I} & \ddots & \vdots \\ \mathbf{0} & -\mathbf{I} & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \mathbf{I} \\ \mathbf{0} & \cdots & \mathbf{0} & -\mathbf{I} \end{bmatrix} \in \mathbb{R}^{pn_u \times (p-1)n_u}, \tag{13.20}$$

$$\overline{\mathbf{G}} = \begin{bmatrix} \mathbf{M}_{V_1}+\mathbf{M}_{V_2} & -\mathbf{M}_{V_2} & \mathbf{0} & \cdots & \mathbf{0} \\ -\mathbf{M}_{V_2} & \mathbf{M}_{V_2}+\mathbf{M}_{V_3} & -\mathbf{M}_{V_3} & \ddots & \vdots \\ \mathbf{0} & \ddots & \ddots & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \ddots & -\mathbf{M}_{V_{p-1}} \\ \mathbf{0} & \cdots & \mathbf{0} & -\mathbf{M}_{V_{p-1}} & \mathbf{M}_{V_{p-1}}+\mathbf{M}_{V_p} \end{bmatrix} . \tag{13.21}$$

The matrix $\overline{\mathbf{G}}$ in (13.21) is a tridiagonal symmetric block matrix which depends on the chosen mass matrices of the virtual bodies. An analytical inverse of a general tridiagonal symmetric block matrix exists, nevertheless its derivation is mathematically very intricate and can be found in [107]. Applying the result in (13.18) naively results in computing the inverse for each splitting configuration and multiplicity p which is not efficient from a computational perspective. However, it is possible to simplify the above structure of $\overline{\mathbf{G}}$ a great deal, if (13.12) is considered. By doing so, the matrix $\overline{\mathbf{G}}$ then only depends on the splitting factors $\{\alpha_1, \ldots, \alpha_p\}$ and on the mass matrix $\mathbf{M}_{B_i}$. By the choice in (13.12), it is simple to factor out a diagonal matrix $\boldsymbol{\alpha} \in \mathbb{R}^{pn_u \times pn_u}$ consisting of all splitting ratios from the mass matrix $\mathbf{M}$ as

$$\mathbf{M} = \boldsymbol{\alpha}\mathbf{M}_p \quad \Rightarrow \quad \mathbf{M}^{-1} = \boldsymbol{\alpha}^{-1}\mathbf{M}_p{}^{-1} \tag{13.22}$$

$$\text{with:} \quad \boldsymbol{\alpha} \coloneqq \operatorname{diag}(\alpha_1\mathbf{I}, \ldots, \alpha_p\mathbf{I}) , \quad \mathbf{M}_p \coloneqq \operatorname{diag}(\mathbf{M}_{B_i}, \ldots, \mathbf{M}_{B_i}) \in \mathbb{R}^{pn_u \times pn_u} , \tag{13.23}$$

where the latter equality in (13.22) follows from $\mathbf{M}$ and $\mathbf{M}_p$ being symmetric. The bilateral Delassus operator $\overline{\mathbf{G}}$ becomes

$$\overline{\mathbf{G}} = \overline{\mathbf{W}}^{\top}\mathbf{M}^{-1}\overline{\mathbf{W}} = \overline{\mathbf{W}}^{\top}\boldsymbol{\alpha}^{-1}\mathbf{M}_p{}^{-1}\overline{\mathbf{W}} = \mathbf{L}\mathbf{M}_{p-1}{}^{-1} \tag{13.24}$$

$$\text{with:} \quad \mathbf{L} \coloneqq \overline{\mathbf{W}}^{\top}\boldsymbol{\alpha}^{-1}\overline{\mathbf{W}} \in \mathbb{R}^{(p-1)n_u \times (p-1)n_u} . \tag{13.25}$$

The last equality in (13.24) follows from the property[1] $\mathbf{M}_p{}^{-1}\overline{\mathbf{W}} = \overline{\mathbf{W}}\mathbf{M}_{p-1}{}^{-1}$, where $\mathbf{M}_{p-1}$ only contains $p-1$ diagonal block entries of value $\mathbf{M}_{B_i}$.

The inverse Delassus matrix $\overline{\mathbf{G}}^{-1}$ is now obtained as

$$\overline{\mathbf{G}}^{-1} = \mathbf{M}_{p-1}\mathbf{L}^{-1} = \mathbf{L}^{-1}\mathbf{M}_{p-1} , \tag{13.26}$$

where the latter is true due to the symmetry of $\overline{\mathbf{G}}$. The velocity update in (13.19) simplifies to

$$\mathbf{u}^{E,s} = \left(\mathbf{I} - \mathbf{M}^{-1}\overline{\mathbf{W}}\mathbf{M}_{p-1}\mathbf{L}^{-1}\overline{\mathbf{W}}^{\top}\right)\mathbf{u}_C^{E,s} \tag{13.27}$$

$$= \left(\mathbf{I} - \boldsymbol{\alpha}^{-1}\mathbf{M}_p{}^{-1}\overline{\mathbf{W}}\mathbf{M}_{p-1}\mathbf{L}^{-1}\overline{\mathbf{W}}^{\top}\right)\mathbf{u}_C^{E,s} \tag{13.28}$$

$$= \left(\mathbf{I} - \boldsymbol{\alpha}^{-1}\overline{\mathbf{W}}\mathbf{L}^{-1}\overline{\mathbf{W}}^{\top}\right)\mathbf{u}_C^{E,s} , \tag{13.29}$$

where we used again the property $\overline{\mathbf{W}}\mathbf{M}_{p-1} = \mathbf{M}_p\overline{\mathbf{W}}$.

[1] This result can be checked by multiplying out the left- and right-hand side or by the idea of double contracting the second and forth slot of the 4^{th} order tensor $\mathbf{M}_p{}^{-1} = \delta^i{}_j m^l{}_k \mathbf{e}_i \otimes \boldsymbol{\epsilon}^j \otimes \mathbf{e}_l \otimes \boldsymbol{\epsilon}^k$ with the first and third slot of the tensor $\overline{\mathbf{W}} = s^i{}_j \delta^l{}_k \mathbf{e}_i \otimes \boldsymbol{\epsilon}^j \otimes \mathbf{e}_l \otimes \boldsymbol{\epsilon}^k$, where $m^l{}_k,\ s^i{}_j \in \mathbb{R}$.

Result (13.26) is of special interest since the real effort of computing $\overline{\mathbf{G}}^{-1}$ lies merely in computing the inverse $\mathbf{L}^{-1}$ which is only dependent on the chosen splitting factors $\{\alpha_1, \ldots, \alpha_p\}$. The matrix $\mathbf{L}$ is still a tridiagonal block matrix in the form:

$$\mathbf{L} = \begin{bmatrix} (\frac{1}{\alpha_1}+\frac{1}{\alpha_2})\mathbf{I} & -\frac{1}{\alpha_2}\mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ -\frac{1}{\alpha_2}\mathbf{I} & (\frac{1}{\alpha_2}+\frac{1}{\alpha_3})\mathbf{I} & -\frac{1}{\alpha_3}\mathbf{I} & \ddots & \vdots \\ \mathbf{0} & \ddots & \ddots & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \ddots & -\frac{1}{\alpha_{p-1}}\mathbf{I} \\ \mathbf{0} & \cdots & \mathbf{0} & -\frac{1}{\alpha_{p-1}}\mathbf{I} & (\frac{1}{\alpha_{p-1}}+\frac{1}{\alpha_p})\mathbf{I} \end{bmatrix}, \quad \mathbf{I} \in \mathbb{R}^{n_u \times n_u} \ . \tag{13.30}$$

The values for the symmetric inverse $\mathbf{L}^{-1}$ can be found with a symbolic algebra program or by applying the analytical solution for a simple tridiagonal matrix since the sub-matrices $\mathbf{I}$ can be neglected. The inverse $\mathbf{L}^{-1}$ yields

$$\mathbf{L}^{-1} = \begin{bmatrix} l_{1,1}\mathbf{I} & l_{1,2}\mathbf{I} & \cdots & l_{1,p-1}\mathbf{I} \\ l_{2,1}\mathbf{I} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & l_{p-2,p-1}\mathbf{I} \\ l_{p-1,1}\mathbf{I} & \cdots & l_{p-1,p-2}\mathbf{I} & l_{p-1,p-1}\mathbf{I} \end{bmatrix}, \tag{13.31}$$

where the scalars $l_{m,n}$ are given as

$$l_{m,n} \coloneqq \begin{cases} (\alpha_1 + \cdots + \alpha_m)(\alpha_{n+1} + \cdots + \alpha_p) & \text{if } n \geqslant m \in [1; p-1] \\ l_{n,m} & \text{if } n < m \in [1; p-1] \end{cases} \tag{13.32}$$

The inverse $\mathbf{L}^{-1}$ is analogue to the inverse obtained for the example of a p-linear ball chain in chapter 8 in [5].

The non-symmetric term $\mathbf{A} \coloneqq \mathbf{I} - \boldsymbol{\alpha}^{-1}\overline{\mathbf{W}}\mathbf{L}^{-1}\overline{\mathbf{W}}^{\top} \in \mathbb{R}^{pn_u \times pn_u}$ in (13.29) has the same block matrix structure as $\mathbf{L}^{-1}$ in (13.31) but with different indices denoted as $a_{m,n}$. The scalar multipliers $a_{m,n}$ for the identity sub-matrices $\mathbf{I}$ in $\mathbf{A}$ yield

$$a_{m,n} \coloneqq \delta_{m,n} - \frac{1}{\alpha_m}\big((l_{m,n} - l_{m,n-1}) - (l_{m-1,n} - l_{m-1,n-1})\big), \quad \forall m, n \in [1\,;\, p]. \tag{13.33}$$

To make (13.33) a proper definition, we define $l_{m,p} = 0,\ \forall m \leqslant p \Rightarrow l_{p,m} = 0$ and note that $l_{0,n} = 0 \Rightarrow l_{n,0} = 0$. Furthermore from (13.32) follows

$$l_{m,n} - l_{m,n-1} = -\alpha_n(\alpha_1 + \cdots + \alpha_m) \qquad \forall n \geqslant m+1 \tag{13.34}$$

$$l_{n,m} - l_{n-1,m} = +\alpha_n(\alpha_{m+1} + \cdots + \alpha_p) \qquad \forall n \leqslant m \ . \tag{13.35}$$

Equation (13.33) needs to be evaluated for three cases: the diagonal part, the strictly lower and strictly upper diagonal part:

- Strictly upper diagonal part with $n \geqslant m+1$ and $m \in [1\,;\, p-1]$:

$$a_{m,n} \overset{(13.33,13.34)}{=} -\frac{1}{\alpha_m}\big(-\alpha_n(\alpha_1 + \cdots + \alpha_m) + \alpha_n(\alpha_1 + \cdots + \alpha_{m-1})\big) \tag{13.36}$$

$$= -\frac{1}{\alpha_m}\big(-\alpha_n\alpha_m\big) = \alpha_n \ . \tag{13.37}$$

- Strictly lower diagonal part with $n \leqslant m-1$ and $m \in [2\,;\,p]$: in this case, the indices of all l-values in (13.33) need to be switched when applying (13.32), that is,

$$a_{m,n} \overset{(13.33,13.32)}{=} -\frac{1}{\alpha_m}\big((l_{n,m} - l_{n-1,m}) - (l_{n,m-1} - l_{n-1,m-1})\big) \tag{13.38}$$

$$\overset{(13.35)}{=} -\frac{1}{\alpha_m}\big(-\alpha_n\alpha_m\big) = \alpha_n \ . \tag{13.39}$$

- Diagonal part with $n = m$ and $m \in [1\,;\,p]$: switching the second l-value in (13.33) yields

$$a_{m,m} \overset{(13.33,13.32)}{=} 1 - \frac{1}{\alpha_m}\big((l_{m,m} - l_{m-1,m}) - (l_{m-1,m} - l_{m-1,m-1})\big) \tag{13.40}$$

$$\overset{(13.35,13.34)}{=} 1 - \frac{1}{\alpha_m}\big(\alpha_m(\alpha_1 + \cdots + \alpha_{m-1} + \alpha_{m+1} + \cdots + \alpha_p)\big) \tag{13.41}$$

$$= \quad \alpha_m \ . \tag{13.42}$$

The velocity update in (13.29) drastically simplifies to

$$\begin{aligned} \mathbf{u}^{E,s} &= \mathbf{A}\ \mathbf{u}_C^{E,s} \ , \\ \begin{bmatrix} \mathbf{u}_{V_1}^{E,s} \\ \vdots \\ \mathbf{u}_{V_p}^{E,s} \end{bmatrix} &= \begin{bmatrix} \alpha_1\mathbf{I} & \cdots & \alpha_p\mathbf{I} \\ \vdots & \vdots & \vdots \\ \alpha_1\mathbf{I} & \cdots & \alpha_p\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{V_1,C}^{E,s} \\ \vdots \\ \mathbf{u}_{V_p,C}^{E,s} \end{bmatrix} . \end{aligned} \tag{13.43}$$

The above states that each end velocity update $\mathbf{u}_{V_1}^{E,s}$ is the affine combination of all virtual body velocities $\mathbf{u}_{V_j,C}^{E,s}$ with the splitting factors $\alpha_j\ \forall j \in [1;p]$. If the splitting factors are chosen uniformly, that is, $\alpha_j = 1/p\ \forall j \in [1;p]$, the intuitive averaging of all velocities suspected at the beginning of this chapter immediately follows from (13.43).

The mass-splitting method is not restricted to certain set-valued force laws between bodies. For the sake of simplicity, set-valued contact laws have been used in this section to explain the mechanism of this method. However, other constraints such as bilateral constraints, that is, translational or rotational joints between bodies, or other set-valued force laws can be added. This only results in additional nodes in the contact graph and does not change the mathematical derivation explained in this section.

13.3 Parallel Considerations

Supposed that the total contact problem at a time instance t is solved iteratively in parallel by all processes. Then in each global iteration, a process P_i computes one velocity SOR Prox iteration over the nodes in its local contact graph G_i, that includes contacts on virtual bodies as well, and afterwards collaborates with all other processes to compute an update of all currently existing split-nodes by (13.43). In figure 13.1, for example,

this would mean that process P_1, P_2 and P_3 iterate first over their local contact nodes, adding up to $3, 2$ and 4 contact nodes, respectively, and afterwards collaborate to compute one iteration over the single split-node (consisting of two bilateral constraints) to update velocities of the virtual bodies V_1, V_2 and V_3 by (13.43).

Collaboration always involves communication among processes and the two common methods for computing the update of a split-node are presented in the following:

- **Gather on Master Process:** One process acts as the master process for one split-node corresponding to one split body B_i and is responsible for evaluating the update (13.43). For this approach, all processes associated with this split-node need to send their virtual body velocities to the master process which then computes the bilateral constraint update (13.43). In figure 13.1, for example, process P_1 takes the role of the master because it also owns body B_i since its center of gravity is contained in its domain. The master process P_1 receives $\mathbf{u}_{V_2,C}^{E,s}$ from process P_2 and $\mathbf{u}_{V_3,C}^{E,s}$ from P_3 among other data to successfully compute the update (13.43). Of course, the master process is also responsible to scatter the result of (13.43) back to the processes P_2 and P_3 for the start of a next iteration over the contact graph. This totals up to $2(p-1)$ sent messages for a split-node with multiplicity p.
- **Gather on All Processes:** All processes send their virtual body velocity $\mathbf{u}_{V_i,C}^{E,s}$ to all other associated processes for a particular split-node. After receiving, all associated processes compute the update (13.43). This totals up to $p(p-1)$ sent messages for a split-node with multiplicity p. Since this is quadratic in p, the master method is the preferred solution.

The last question which remains is how to communicate the multiplicity factor of a split body. Since each body B_i is managed by a master process, for example P_m, determined by is center of gravity, a simple way to achieve this is that each neighbor process P_k maintaining a remote body of B_i informs the master process about the external forces, contact forces and possible other constraints on the remote body B_i determined in its process domain. It is important to note that a remote body B_i in some neighbor process P_k might be free of any interaction, and therefore P_k does not need to be involved in a potential split of body B_i. The master process determines if a body split is necessary and communicates all split-node multiplicities to the involved processes which then prepared for the local contact graph iteration. This means each process, including the master process, splits the mass $\mathbf{M}_{B_i}$ and nonlinear terms $\mathbf{h}_{B_i}$ of body B_i according to (13.12).

The procedure of solving the contact problem (8.8) in parallel on k processes by using the mass-splitting procedure is summarized in algorithm 13.1 and implemented in `InclusionSolverMPI`[12].

```
1  parfor process P_i ∈ {P_1, ..., P_k} :
2  |  Determine all domain overlapping bodies in domain P_i.
3  nbcomm among processes {P_1, ..., P_k} :
4  |  Determine the multiplicity factor p of all split bodies by their master process.
5  parfor process P_i ∈ {P_1, ..., P_k} :
6  |  Split remote and local domain overlapping bodies in P_i according to their
   |   multiplicity (cf. (13.12)).
7  for each global contact graph iteration η :
8  |  parfor process P_i ∈ {P_1, ..., P_k} :
9  |  |  Iterate over the local contact graph G_i and perform a velocity SOR Prox
   |  |   update.
10 |  nbcomm among processes {P_1, ..., P_k} :
11 |  |  Collaborate to solve all split-nodes on their corresponding master
   |  |   processes.
```

Algorithm 13.1: Basic SOR Prox iteration for solving the contact problem in parallel on k processes in (8.8) by using the mass-splitting method . Abbrevations: **parfor**: parallel for, **nbcomm**: neighborhood communication.

13.4 Related Work

The mass-splitting procedure explained in this section has mostly been inspired by [196, 7] at time of implementing it into the `GRS` framework discussed in chapter 9. In [196], the mass-splitting method is also called *non-smooth contact domain decomposition* and the minor difference to our method lies in the resolution of the bilateral constraints. In [196], the bilateral forces $\mathbf{S}^{\eta}$ are directly computed by (13.18) and substituted back into (13.16) to get the velocity update. This procedure results in computing the inverse $\mathbf{L}^{-1}$ or iteratively solving the implicit proximal equation of (13.13). With the help of the analytical solution in (13.43), this is not necessary for a chosen mass distribution in (13.12). The mass-splitting approach is also discussed in the field of computer graphics in [192] where it is used to simulate large multi-body systems. In [192], the focus lies more on real-time simulation, where the accuracy of the contact resolution is sacrificed by an early termination of the contact iteration. In contrast to the mass-splitting method being used as a contact graph decomposition approach presented here and in [196, 7], the ulterior motive in [192] is the parallelization of a JOR Prox algorithm (cf. section 8.3.1). Unfortunately, the proof of (13.43) in the additional supplemental material in [192] is not clear.

The parallelization effort in [174] is of much interest for the discussed mass-splitting method. The work in [174] showed that the parallelization of the contact problem does not influence the solution of the contact forces in a statistical sense. For the kd-tree domain decomposition in [174] also an opaque overlap region is introduced similar to [75] (cf. section 12.2). The iterative scheme is very much comparable to the mass-splitting method used in this work. The parallel scheme in [174] does not split the bodies but rather splits the contact graph into local and remote contacts defined by the overlap region. Instead

of communicating velocities, as in the case of the mass-splitting method for solving the bilateral constraints, contact forces on remote bodies are communicated to neighboring processes. Therefore, the method in [174] can be seen as a dual analogue to the mass-splitting method. From an implementation perspective, the contact force communication in [174] is supposed to be more complex compared to velocity communication, especially when considering different contact laws.

Chapter 14

Visualization and Data Extraction

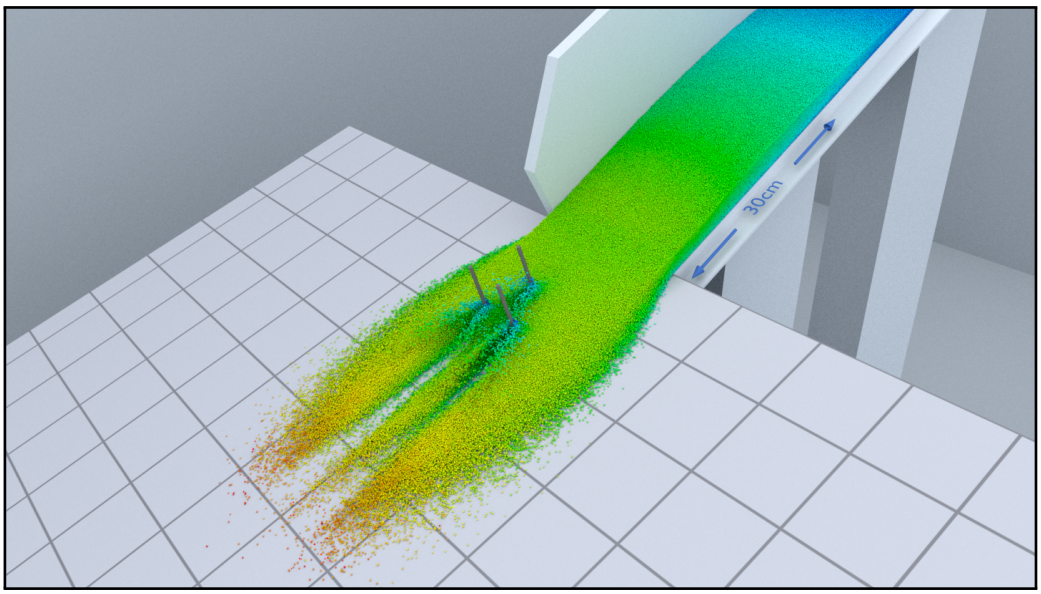
Several converter tools integrated in the command-line application `GRSFConverter` have been developed to help visualize and analyze the simulation output of one of the rigid body applications `GRSFSim`, `GRSFSimGUI` or `GRSFSimMPI`. The main input file used by these tools is the binary simulation file with suffix `.sim` which contains all rigid body states of all simulated bodies of the provided XML scene file. A certain rigid body simulation mostly consists of several large `.sim` files and therefore all tools can handle a list of these files. The `GRSFConverter` provides the following functionalities:

- **SimConverter**: This application resamples, joins or splits a set of `.sim` files.
- **SimInfo**: This small application shows information of a set of `.sim` files and also provides state index information for certain striding patterns over the specified files to be used with the SimConverter tool.
- **Analyzer**: This tool feeds several `.sim` files to an execution graph network which contains several execution nodes specified in an XML file. This concept is essentially similar to the one used in the application *Matlab Simulink* or in certain audio tools used in the music industry. More is said in the further course of this chapter.
- **Renderer**: This application produces render input data which can be handed to a ray tracing application to render images of the simulated scene. Currently, the application supports the generation of Renderman Interface Bytestream *RIB* output. The Renderman Interface Specification *RISpec* [146] specification, also described in more detail in [178], is a three-dimensional scene description language understood by a *RISpec*-compliant renderer to render an image of the scene. The *RISpec* is the de facto standard for rendering and supported by a lot of commercial and non-commercial renderers. This includes the famous photorealistic renderer *Renderman* developed at Pixar Animation Studios. This tool is also based on an execution graph provided by an XML file and analogue to the analyzer.
- **Gridder**: This tool is used to collect certain data inside a specified Cartesian grid, such as positions, velocities, process ids, collision information etc., provided by a set of `.sim` files . The output is a HDF5 file which can be easily read and manipulated in `python` for further data visualization and validation tasks.

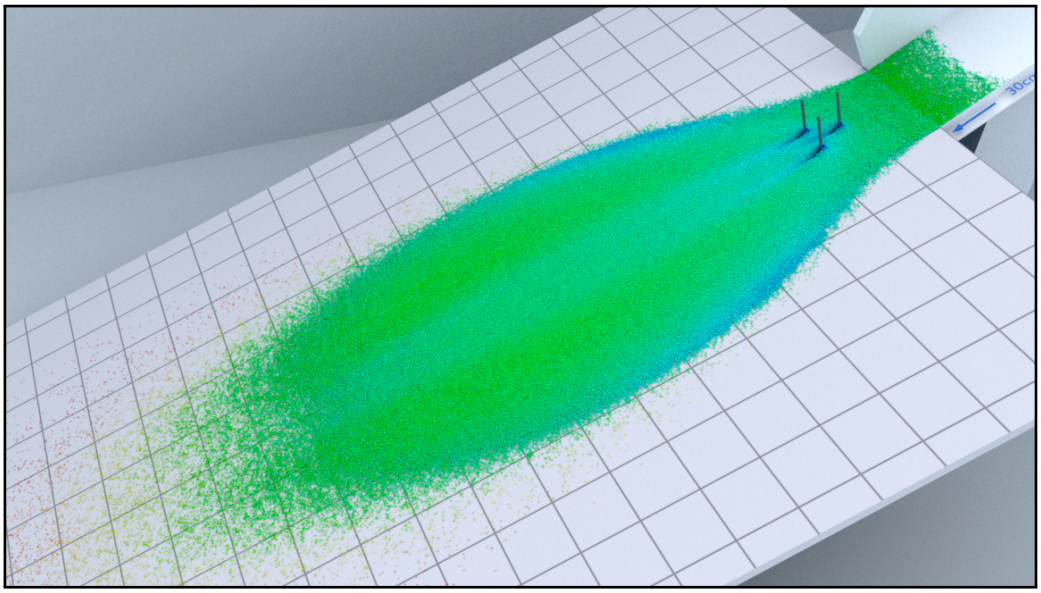
In the following, we briefly describe the sophisticated execution graph implementation used for both the analyzer and the render tool of the `GRSFConverter` application in more detail.

14.1 Execution Graph

The execution graph implemented in `ExecutionTree`[39] is a directed acyclic graph consisting of several connected logic nodes derived from `LogicNode`[36] which define a simple input/output control flow. Each logic node in the execution graph contains several input/output sockets (`LogicSocket`[38]) with a certain type out of the predefined types defined in `LogicSocketTypes`[37]. An execution graph works in the way that each logic node contains a specific `compute` routine which provides values for the output sockets by using the values from the input sockets. Each output of a logic node can be linked to an input of the same type of another logic node. This means an output socket of the arithmetic type `double` cannot be linked to an input socket of integral type `int` for example. Each logic node can be assigned to one or more execution groups which are collections of logic nodes and form directed acyclic subgraphs. For each execution group, an execution order is computed such that the data flow defined by the input/output links in the group is respected. An execution order of an execution group is called a *topological ordering* in computer science, and such an ordering always exists for a directed acyclic graph, despite being non-unique. A topological ordering of an execution group is an ordering of all logic nodes such that for all connections from a logic node A to B, A precedes B in the ordering. Each execution graph network consists of several input logic nodes whose output sockets are initialized before executing the network. The implementation in `LogicSocket`[38] allows two types of directed links between an input and output socket, namely a *get* and a *write* connection. A *write* link is a link from an output socket i of a node A to an input socket j of some node B, denoted as $\{A, i\} \rightarrow \{j, B\}$. A *write* link basically duplicates a write request to the output socket i of A also to an additional write request to the input socket j of B. A *get* link is the exact opposite and is a link from an input socket j of a node B to an output socket i of a node A, denoted as $\{A, i\} \leftarrow \{j, B\}$. A *get* link basically forwards any read access on the input socket j of B to a read access on the input socket i of A. Most of the time only *get* nodes are necessary but as soon as the execution graph becomes more complex and certain switching behavior should be reproduced, the additional *write* links are a convenient tool to realize this. Cyclic paths between logic nodes are detected and result in an error when building the execution network. The write and read access of input and output sockets is implemented using a fast static type dispatch system in `LogicSocket`[38] which is based on the famous CRTP pattern [36] and analogue to the dispatch used in the `boost::variant` class in [26]. Static type dispatching avoids the use of virtual calls when using polymorphic objects in object-oriented programming languages. At time of writing, the related work to be used in further work is the open-source library `TensorFlow`™ [1]. Figure 14.2 visualizes a simplification of an execution graph used for the simulation studies discussed in part III. It basically counts the number of bodies inside an object oriented box and writes the result into an XML file. The execution graph was especially useful for rendering tasks. It gives the user the ability to setup



(a)



(b)

Figure 14.1: Two renderings of the chute flow simulation B-7 in figure 17.1b at time $t = 0.5\,\mathrm{s}$ in (a) and $t = 1.3\,\mathrm{s}$ in (b). The Renderman Interface Byte stream for the bodies was generated by an execution graph with the render tool implemented in `GRSFConverter`. The frame was rendered with Pixar Renderman on the Euler cluster at ETH Zurich. The color map (blue-green-red) indicates the velocity magnitude.

an input/output work flow which converts bodies and their geometry into render specific input such as a Renderman Interface Bytestream by providing render-specific logic nodes ([40, 41]C++). This allows the user to easily colorize the rigid bodies of a simulation according to their velocity magnitude or for example by their accumulated gap penetration depth.

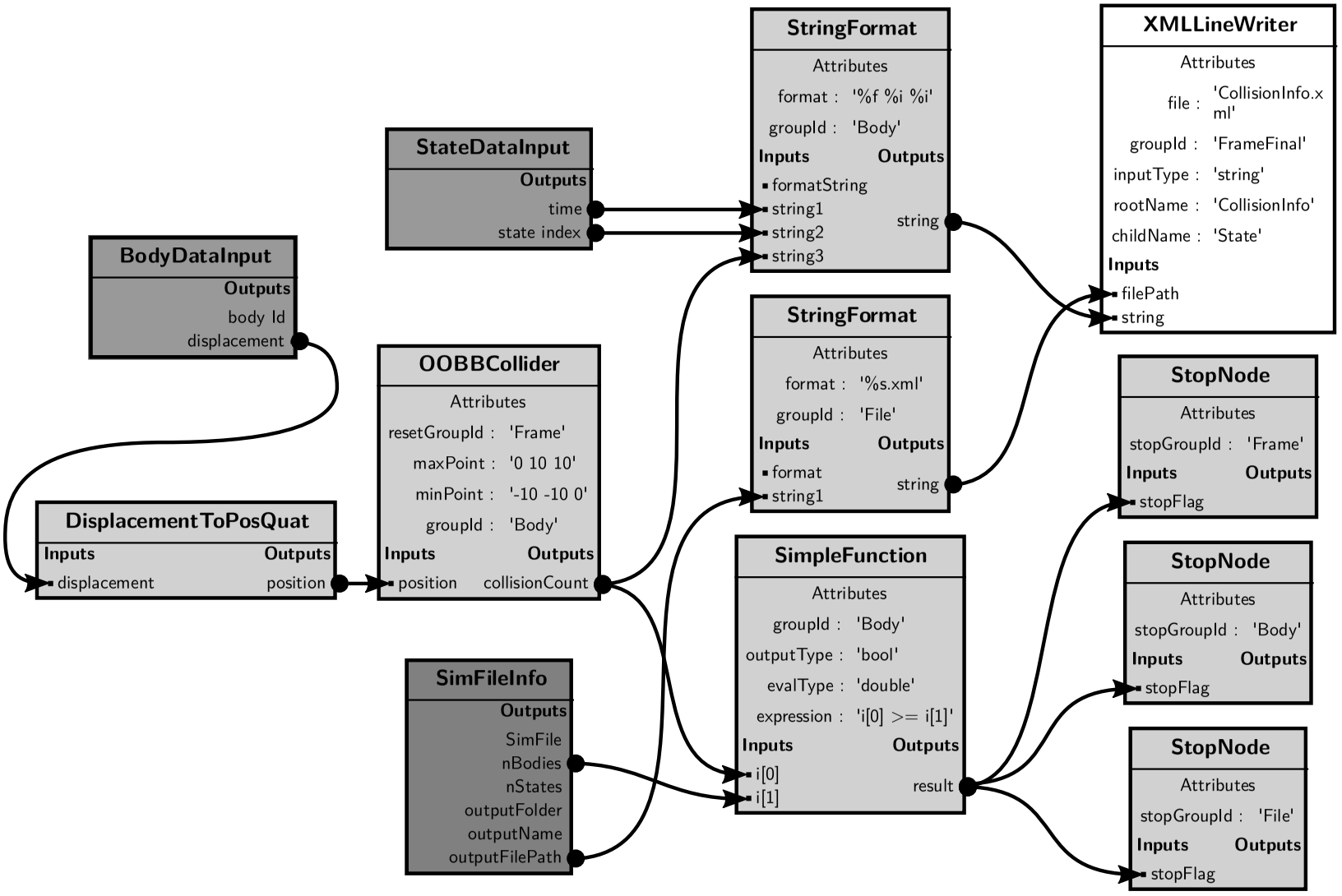


Figure 14.2: Execution graph in [1][file] for the analyzer tool of the `GRSFConverter` which counts the number of bodies in an oriented bounding box and writes the results to an XML file. All edges denote *get* links between output and input. More complex graphs have been used for the rendering tool in the `GRSFConverter` application.

Part III

Application to Chute Flows

"Programming today is a race between software engineers striving to build bigger and better idiot-proof programs, and the Universe trying to produce bigger and better idiots. So far, the Universe is winning."

— Rich Cook, The Wizardry Compiled, 1989

This part discusses all application related work of this thesis. Chapter 15 introduces some useful notions from statistics and particle size analysis which is necessary to properly describe a granular material. Chapter 16 discusses the mechanical model of the chute flow experiments which were performed at the WSL Institute for Snow and Avalanche Research SLF in Davos. Chapter 16 gives insight into the evaluation of the experiments, the subsequent advanced velocity field reconstruction from the recorded video footage and the simulation studies computed on the high-performance cluster Euler at the ETH Zurich. Chapter 17 discusses some results of the comparison between experiment and simulation. The thesis closes with a conclusion and outlook in chapter 18.

Chapter 15

Particle Size Analysis

In reality, the particle sizes of a granular material, for example sand, gravel, glass beads etc., are heterogeneous and statistically distributed. Estimating and approximating the size distribution of a granular material, also referred to as *granulometry*, is important when comparing experimental data and results obtained from simulations. A common measure for an arbitary shaped particle is its *equivalent diameter* of an enclosing sphere. The outcome of a particle or grain size analysis is the particle size distribution expressed in terms of the equivalent diameter of the particle. This chapter gives a brief theoretical and experimental overview about obtaining these quantities. A reader experienced in the field of statistics may find the discussion in this chapter rather trivial. However, the proper and brief demonstration in this chapter is justified mainly because the intricacies of the presented results lie in the details which when misunderstood lead to severe consequences for validating simulation results by experiments. The subtle differences between, for example, a cumulative diameter distribution based on the mass of particles or based on the number of particles quickly leads to confusion and does not become clear from references such as [179, 105], also due to the use of insufficient notation.

15.1 Measurement Methods

Particle size measurements can be done with several procedures, for example by a *sieve analysis*, by *sedimentation* or *optical granulometry*, to name a few. The sieve analysis, also called gradation test, is a simple and effective method which is commonly applied to material in the sub-millimeter to meter range and often used in the agriculture industry. The particle size distribution, that is, the density function with respect to the equivalent diameter, of a granular material can be expressed in terms of either the number, the surface area, the volume or the mass of particles, which is again implicitly given by the used measurement method. The sieve analysis, for example, generally provides an equivalent diameter distribution in terms of the mass of particles. The procedure of the sieve analysis is as follows: a sample of the granular material is passed through a stack of sieves from largest to smallest opening size. After shaking the sieves for some amount of time, each sieve contains its retained mass. By weighting the retained mass in each sieve, a histogram of the particle size distribution is obtained.

15.2 Particle Size Distribution

Some concepts from probability theory and statistics are introduced first to later properly describe the mass or count distribution of a granular material depending on its equivalent diameter.

Let $X : \Omega \to \mathbb{R}$ be a real continuous random variable with a density differential measure $\mathrm{d}r$ such that $\int_{(-\infty,\infty]} \mathrm{d}r = r_{tot}$. The probability space is denoted as Ω.

Definition 15.1 *(Cumulative Distribution Function)*:
The cumulative distribution function F_X^r of X can be defined as

$$F_X^r(x) \coloneqq \frac{\int_{(-\infty,x]} \mathrm{d}r(\tau)}{\int_{(-\infty,\infty]} \mathrm{d}r(\tau)} = \frac{1}{r_{tot}} \int_{(-\infty,x]} \mathrm{d}r(\tau) \ , \tag{15.1}$$

where the superscript ^{r} denotes the used measure, that is, the measure $\mathrm{d}r$.

The Lebesque-Stieltjes measure $\mathrm{d}r$ in (15.1) can be split into the sum of a Lebesque measure, a pure atomic measure and a possible singular measure, which, for the sake of simplicity, is assumed to be zero. A nonvanishing atomic part in $\mathrm{d}r$ results in a cumulative distribution function which contains jumps at certain points. For a given value x, the value $F_X^r(x)$ describes the percentage with respect to $\mathrm{d}r$ of the total value r_{tot}. If the measure contains only a Lebesque integrable part, the above integral can be written as

$$\frac{1}{r_{tot}} \int_{(-\infty,x]} \mathrm{d}r(\tau) = \int_{(-\infty,x]} f_X^r(\tau) \mathrm{d}\tau \quad \Leftrightarrow \quad \frac{\mathrm{d}r(\tau)}{r_{tot}} = f_X^r(\tau) \mathrm{d}\tau \ , \tag{15.2}$$

where f_X^r is recognized as a normalized density function with respect to the Lebesque measure $\mathrm{d}\tau$.

Definition 15.2 *(Quantile Function)*:
The inverse of the cumulative distribution function F_X is the quantile function Q_X given as

$$Q_X^r(p) \coloneqq F_X^{r\,-1}(p) = \inf_{x \in \mathbb{R}} \{ F_X^r(x) \geqslant p \} \ . \tag{15.3}$$

Let D be a real continuous random variable for the equivalent diameter d of a particle. The cumulative diameter distribution function F_D^m and its normalized density f_D^m in terms of the mass measure $\mathrm{d}m(d)$ is given as

$$F_D^m(d) \coloneqq \frac{1}{m_{tot}} \int_0^d \mathrm{d}m(\tau) = \int_0^d f_D^m(\tau) \mathrm{d}\tau, \tag{15.4}$$

where m_{tot} denotes the total mass of the granular material. The cumulative density F_D^m given by its normalized mass density f_D^m can be approximated by an experimental sieve analysis explained in more detail in the following. The functions F_D^m and f_D^m are illustrated in figure 15.2. The following section describes the sieve analysis and shows how the cumulative size distribution F_D^m in terms of the mass of particles is mapped to to a cumulative size distribution in terms of the number of particles.

15.3 Sieve Analysis

Consider n stacked sieves with nominal screen openenings $d_1 > d_2 > \cdots > d_n = 0$ through which a granular material of total mass M_T passes. Each sieve $i \in [1, n-1]$ with opening size d_i retains particles with an equivalent diameter greater than d_i. The last sieve n has an opening $d_n = 0$ such that no material can pass and is considered to simulate the floor which retains the rest material at the end of the sieving process. The sieve analysis is depicted in figure 15.1.

The mass ratio s_i of the retained mass M_i in sieve i to the total mass M_T is defined as

$$s_i \coloneqq \frac{M_i}{M_T}. \tag{15.5}$$

The cumulative retained mass ratio k_i for a sieve i is given as

$$k_i \coloneqq \sum_{j \leqslant i} s_j = \frac{1}{M_T} \sum_{j \leqslant i} M_j. \tag{15.6}$$

The cumulative retained mass ratio k_i in a sieve i corresponds to

$$k_i = \frac{1}{m_{tot}} \int_{d_i}^{d_{max}} \mathrm{d}m(\tau) = 1 - F_D^m(d_i). \tag{15.7}$$

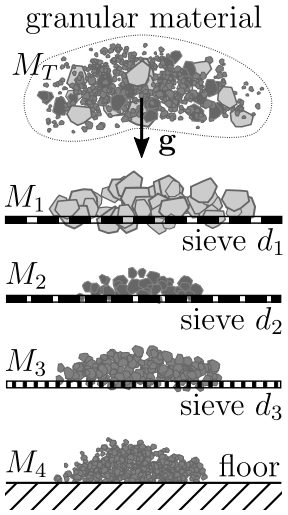


Figure 15.1: Sieve analysis

Discrete function values of the cumulative size distribution F_D^m can be computed for all sieves, that is, $F_D^m(d_i) = k_i - 1 \ \forall i \in [1\,;\, n]$. The more sieves are used for the analysis, the more discrete evaluations of the cumulative size distribution are known. These points can be used as sample points for a fitted curve to obtain a smooth approximation of F_D^m. Figure 15.2 shows a piecewise linear approximation $\bar{F}_D^m$ of the cumulative size distribution F_D^m which results in a piecewise constant density approximation $\bar{f}_D^m$.

The mass measure $\mathrm{d}m(d)$ depending on the equivalent diameter d can also be related to the count measure $\mathrm{d}n(d)$ which when integrated over the domain of d sums up to the total amount of particles n_{tot} contained in the granular material, that is,

$$n_{tot} = \int_{d_{min}}^{d_{max}} \mathrm{d}n(d). \tag{15.8}$$

Using the volume $d^3 \frac{\pi}{6}$ of a sphere with diameter d yields the equality of measure

$$m_{tot} = \int_{d_{min}}^{d_{max}} \mathrm{d}m(d) = \int_{d_{min}}^{d_{max}} \rho d^3 \frac{\pi}{6} \mathrm{d}n(d) \quad \Leftrightarrow \quad \mathrm{d}m(d) = \rho \frac{\pi}{6} d^3 \mathrm{d}n(d), \tag{15.9}$$

where the volumetric mass density ρ is assumed to be independent of d and constant for simplicity. The factor $\Psi_m = \rho \frac{\pi}{6}$ is also called form factor in the terminology of particle

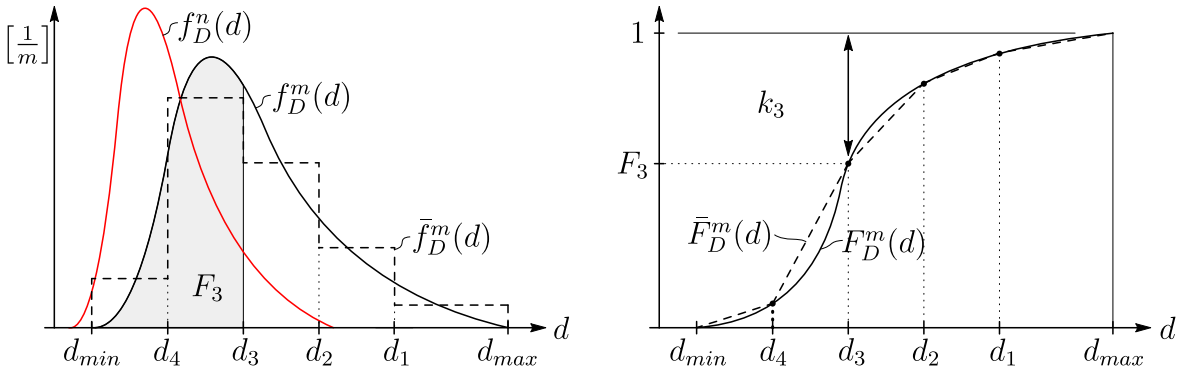


Figure 15.2: The particle diameter density f_D^m with respect to the mass of particles and the transformed density F_D^n with respect to the number of particles. The discrete points on the left, for example, (d_3, F_3), are obtained from a particle size analysis, such as the sieve analysis described in section 15.3.

size analysis (see section 2.5 in [179]). The cumulative count distribution function F_D^n can now be expressed in terms of the density f_D^m as

$$F_D^n(d) := \frac{1}{n_{tot}} \int_{d_{min}}^{d} \mathrm{d}n(\tau) = \frac{\displaystyle\int_{d_{min}}^{d} \Psi_m^{-1} \tau^{-3} \mathrm{d}m(\tau)}{\displaystyle\int_{d_{min}}^{d_{max}} \Psi_m^{-1} \tau^{-3} \mathrm{d}m(\tau)} \tag{15.10}$$

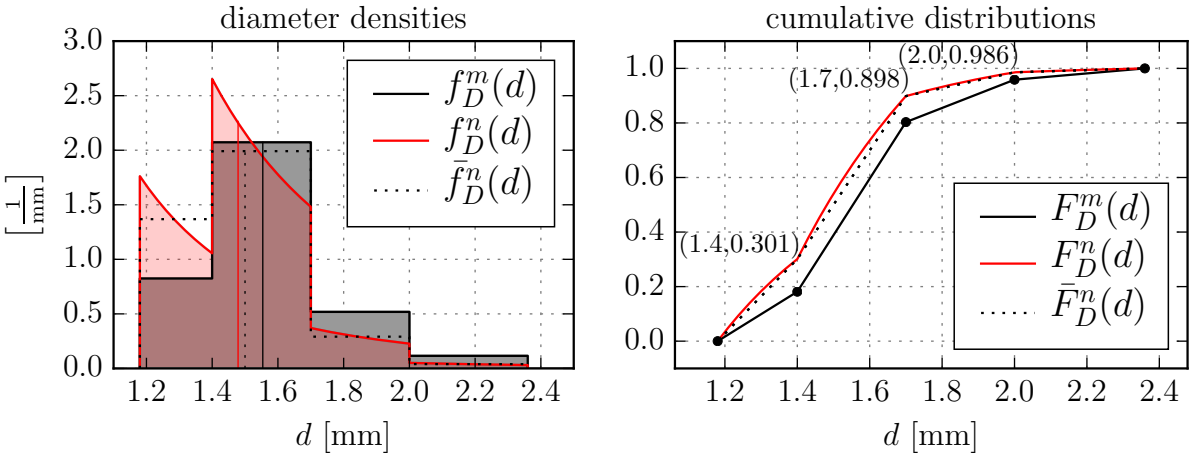
$$= \frac{\displaystyle\int_{d_{min}}^{d} \Psi_m^{-1} \tau^{-3} f_D^m(\tau) \mathrm{d}\tau}{\displaystyle\int_{d_{min}}^{d_{max}} \Psi_m^{-1} \tau^{-3} f_D^m(\tau) \mathrm{d}\tau}, \tag{15.11}$$

where the form factor Ψ_m can be canceled out if it is not dependent on d. The particle size density f_D^n based on the number of particles then follows as

$$f_D^n(d) := \frac{\Psi_m^{-1} d^{-3} f_D^m(d)}{\displaystyle\int_{d_{min}}^{d_{max}} \Psi_m^{-1} \tau^{-3} f_D^m(\tau) \mathrm{d}\tau} \,. \tag{15.12}$$

15.4 Particle Generation for Simulation

The measure on which the diameter density and cumulative diameter distribution function depends on is directly given by the measurement method used for the particle size analysis. The last section described the sieve analysis which yields an approximation of the equivalent diameter density f_D^m and cumulative distribution F_D^m which are based on the mass of particles. These functions can be transformed to a number based representation f_D^n and F_D^n by using (15.12) and (15.10), respectively. The important question arises



(a) (b)

	$f_D^m(d)$	$f_D^n(d)$	$\bar{f}_D^n(d)$
$\mu := \mathrm{E}[d]$	1.576	1.494	1.507
$d_{50} := \mathrm{Med}[d]$	1.554	1.481	1.500

(c)

Figure 15.3: Diameter density and cumulative distribution of the granural material *Starlitebead® 1400* based on the mass (black) and based on the number of particles (red). The functions f_D^n and F_D^n are obtained by (15.10) and (15.12). The approximative piecewise linear distribution $\bar{F}_D^n$ of the density $\bar{f}_D^n$ (dashed red) is used for the particle generation. The vertical lines in (a) show the median values d_{50}.

which of the two distributions F_D^m or F_D^n should be sampled to obtain a representative collection of particles for the numerical simulation. This does not become evidently clear from book references such as [179] and should be addressed in the following.

For the following discussion, the number of particles to simulate is known and denoted by n_{sim}. All particles are assumed to have a spherical shape and are represented on the computer by spheres. The distribution to sample to obtain a collection of n_{sim} particles for the simulation is the diameter distribution based on the number of particles F_D^n. The reason why this is correct is that sampling from the mass based diameter distribution F_D^m would give a collection which when run through the sieve analysis again would produce a perturbed result compared to F_D^m. This is not the case for a collection sampled according to F_D^n, since performing again a sieve analysis would result in a mass based distribution F_D^m, at least for $n_{sim} \to \infty$.

The granular material used in the experiments is called *Starlitebead® 1400* and is produced by the European division of Potters Industries LLC. The main properties of the material *Starlitebead® 1400* are given in table 15.1. The sieve analysis 3° from the certification of conformity [147] of the material *Starlitebead® 1400* for the 5 sieves is shown in table 15.2. The diameter density and cumulative diameter distribution based on the mass and number of particles computed by (15.10) and (15.12) are shown in figure 15.3. The mean equivalent diameter $\mathrm{E}[D] = \mu$ and median $\mathrm{Med}[D] = d_{50}$ given a density $f_D(d)$ and distribution $F_D(d)$

volumetric mass density	2400	$\mathrm{kg\,m^{-3}}$
bulk density	1590	$\mathrm{kg\,m^{-3}}$
angle of repose	26 ± 1	$^\circ$
nominal diameter	1.4	mm

Table 15.1: Properties extracted from [34, 147] of the granular material *Starlitebead® 1400*.

sieve number	1	2	3	4	5
diameter d_i [mm]	2.36	2	1.7	1.4	1.18
retained mass ratio range k_i [%]	0-2	0-10	0-40	60-100	95-100
average retained mass ratio k_i [%]	1	5	20	80	97.5
$F_D^m(d_i) = 1 - k_i$ [%]	99	95	80	20	2.5

Table 15.2: The sieve analysis in [147] for the granular material *Starlitebead® 1400*.

is computed by

$$\mathrm{E}[D] \coloneqq \int_{d_{min}}^{d_{max}} \tau f_D(\tau)\mathrm{d}\tau \ , \qquad \mathrm{Med}[D] \coloneqq Q_D(0.5) \ . \tag{15.13}$$

It is important to note that all aforementioned densities and cumulative distributions describe the equivalent diameter of the particles. To enrich the picture about distributions and densities in the following, we consider the mass density f_M^n and mass cumulative distribution F_M^n based on the number of particles. The mass density which describes how the mass of the particle is distributed can be derived from the diameter density f_D^n. If the density ρ is constant and the particles are assumed to be perfectly spherical then the mass m of a particle with diameter d is given as $m = g(d) = \frac{\pi}{6} d^3 \rho$. The function g is strictly increasing, that is, $\forall d_1, d_2 \in I_d = [d_{min}, d_{max}] : d_2 > d_1 \Rightarrow g(d_2) > g(d_1)$ which admits a strictly increasing inverse $g^{-1}(m)$ on the support of $m \in I_m = \{m = g(d) : d \in I_d\}$. The cumulative distribution function F_M^n for the mass m can be computed by

$$F_M^n(m) \coloneqq \begin{cases} 1 & \text{if } m > x \ \forall x \in I_m \\ F_D^n(g^{-1}(m)) & \text{if } m \in I_m = [g(d_{min}), g(d_{max})] \\ 0 & \text{if } m < x \ \forall x \in I_m \ . \end{cases} \tag{15.14}$$

Because D is an absolutely continous random variable, the mass density is

$$f_M^n(m) \coloneqq \begin{cases} f_D^n(g^{-1}(m)) \frac{\mathrm{d}g^{-1}(m)}{\mathrm{d}m} & \text{if } m \in I_m \\ 0 & \text{if } m \notin I_m \ . \end{cases} \tag{15.15}$$

Another important value for the calibration of a numerical simulation and a real experiment is the expected mass $\mathrm{E}[M_{tot}] = \mathrm{E}[M_1 + \cdots M_k]$ of k particles, where $M_i(D_i)$ is the

$\mathrm{E}[M]$ with $D \backsim F_D^n$	$4.412\,51 \times 10^{-6}$	kg
$\mathrm{E}[M]$ with $D \backsim \bar{F}_D^n$	$4.525\,94 \times 10^{-6}$	kg

Table 15.3: Expected mass for one particle for the cumulative distributions F_D^n and $\bar{F}_D^n$ of the granular material *Starlitebead® 1400*.

random variable of the mass of particle i. It can be computed by

$$\mathrm{E}[M_{tot}(D_1,\ldots,D_k)] = \mathrm{E}\left[\sum_{i=1}^{k} g(D_i)\right] \quad \text{with: } D_i \backsim F_{D_i}^n \tag{15.16}$$

$$= \int_{d_{min}}^{d_{max}} \ldots \int_{d_{min}}^{d_{max}} \left(\sum_{i=1}^{k} g(d_i)\right) \; f(d_1,\ldots,d_k)\; \mathrm{d}\tau_1 \ldots \mathrm{d}\tau_k, \tag{15.17}$$

where f is the joint density for all random variables D_i. Because all D_i are statistically independent from each other, the density can be split into the multiplication of the densities $f_{D_i}^n$ such that

$$\mathrm{E}[M_{tot}] = \int_{d_{min}}^{d_{max}} \ldots \int_{d_{min}}^{d_{max}} \left(\sum_{i=1}^{k} g(\tau_i)\right) \; \prod_{j=1}^{k} f_{D_j}^n(d_j)\; \mathrm{d}\tau_1 \ldots \mathrm{d}\tau_k \tag{15.18}$$

$$= \sum_{i=1}^{k} \int_{d_{min}}^{d_{max}} \ldots \int_{d_{min}}^{d_{max}} g(\tau_i) \; \prod_{j=1}^{k} f_{D_j}^n(\tau_j)\; \mathrm{d}\tau_1 \ldots \mathrm{d}\tau_k \tag{15.19}$$

$$= \sum_{i=1}^{k} \left(\int_{d_{min}}^{d_{max}} g(\tau_i) f_{D_i}^n(\tau_i) \mathrm{d}\tau_i\right) \underbrace{\int_{d_{min}}^{d_{max}} \ldots \int_{d_{min}}^{d_{max}} \prod_{j=1, j\neq i}^{k} f_{D_j}^n(\tau_j) \underbrace{\mathrm{d}\tau_1 \ldots \mathrm{d}\tau_k}_{\text{without } \mathrm{d}\tau_i}}_{=1} \tag{15.20}$$

$$= \sum_{i=1}^{k} \int_{d_{min}}^{d_{max}} g(\tau_i) f_{D_i}^n(\tau_i) \mathrm{d}\tau_i = \sum_{i=1}^{k} \mathrm{E}[M_i]. \tag{15.21}$$

Equation (15.21) shows the linearity of the expected value operator $\mathrm{E}[\cdot]$ in the case of statistically independent random variables. If we assume that all k particles have the same cumulative diameter distribution F_D^n, that is, $D_i \backsim F_{D_i}^n = F_D^n$, equation (15.21) yields

$$\mathrm{E}[M_{tot}] = k\; \mathrm{E}[M] = k \int_{d_{min}}^{d_{max}} g(\tau) f_D^n(\tau) \mathrm{d}\tau. \tag{15.22}$$

The expected values for the mass of a particle of the granular material *Starlitebead® 1400* are shown in table 15.3.

Chapter 16

Mechanical Model and Experiment

The mechanical model for the conducted chute flow experiments in this work is visualized in figure 16.1. Due to the simplicity of the experimental setup shown in figure 16.2, the mechanical model can be cast as an accurate replica with respect to the used geometry in the experiment. The mechanical model in figure 16.1 therefore also represents the experimental setup. Differences to the experimental setup, the model assumptions and parameter choices will be explained throughout this section.

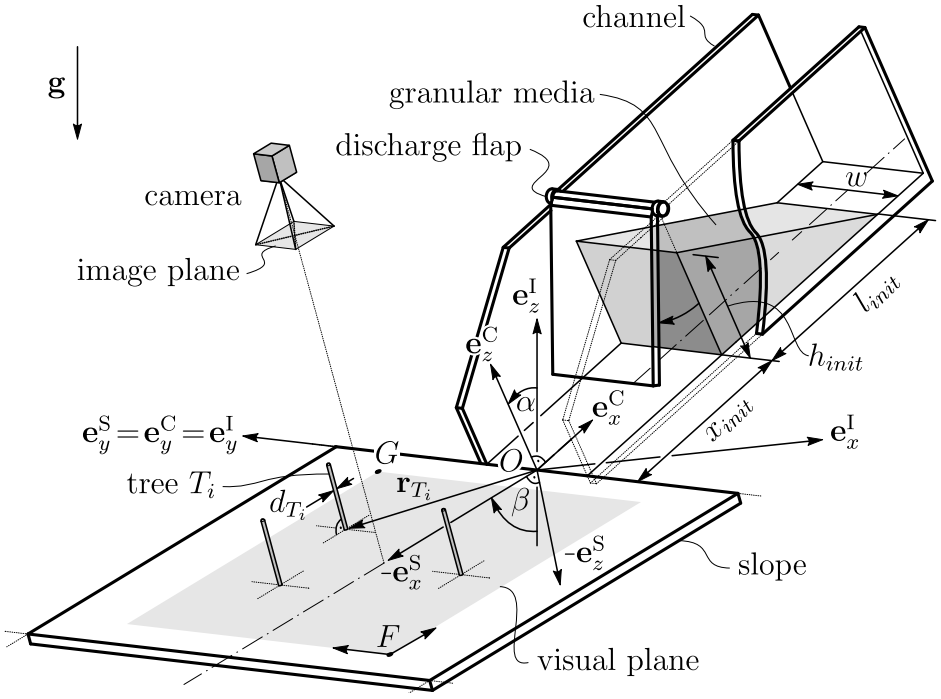


Figure 16.1: A visualization of the mechanical model for the chute flow experiments.

The mechanical model presented in figure 16.1 for the chute flow consists of an inclined channel where the granular material is retained until the discharge flap is opened and the material flows down the channel and spreads on the slope attached at the end of the channel. The inertial coordinate system I with basis $(\mathbf{e}_x^{\mathrm{I}}, \mathbf{e}_y^{\mathrm{I}}, \mathbf{e}_z^{\mathrm{I}})$ is located at the origin O at the end of the channel. The direction of the gravitational acceleration is given

as $\mathbf{g} := -g\mathbf{e}_z^{\mathrm{I}}$, where g denotes the gravitational acceleration. The rectangular channel is rotated positively by an angle α (also called angle of dip) around the axis $\mathbf{e}_y^{\mathrm{I}}$ which defines the channel-fixed coordinate system $\mathrm{C} := (\mathbf{e}_x^{\mathrm{C}}, \mathbf{e}_y^{\mathrm{C}}, \mathbf{e}_z^{\mathrm{C}})$ located at the origin O. The axis $\mathbf{e}_x^{\mathrm{C}}$ marks the center line of the channel. The channel consists of two flat parallel boundary walls and a flat bottom surface. All three walls are aligned with the coordinate system C. The lower slope is rotated positively around the axis $\mathbf{e}_y^{\mathrm{I}}$ by an angle $\frac{\pi}{2} - \beta$ which defines the surface-fixed coordinate system $\mathrm{S} := (\mathbf{e}_x^{\mathrm{S}}, \mathbf{e}_y^{\mathrm{S}}, \mathbf{e}_z^{\mathrm{S}})$ located at the origin O. The surface is equipped with several tree models T_i each located at $\mathbf{r}_{T_i}$ from the origin O. Each tree consists of a cylinder with diameter d_{T_i}. The cylinder axis is pointing in the normal direction of the slope, namely in the direction $\mathbf{e}_z^{\mathrm{S}}$. The discharge flap is shifted along the axis $\mathbf{e}_x^{\mathrm{C}}$ by the amount x_{init} which marks the initial configuration of the front boundary of the granular material at time $t = 0$. At the initial configuration, the boundary of the poured granular material defines a height h_{init} and a length l_{init} in the direction $\mathbf{e}_z^{\mathrm{C}}$ and $\mathbf{e}_x^{\mathrm{C}}$, respectively. For conducting the experiment, a camera is mounted above and normal to the slope such that the image and visual plane are aligned with the $\mathbf{e}_x^{\mathrm{S}}$ and $\mathbf{e}_y^{\mathrm{S}}$ axes. The visual plane is defined by the displacement vectors $\mathbf{r}_{OF}$ and $\mathbf{r}_{OG}$ of the two points F and G, respectively.

Since the glass beads of the granular material *Starlitebead® 1400* used in the experiment have spherical shape, the granular material is modeled likewise as a polydisperse rigid body assembly consisting of n_b spheres. The spheres are generated by drawing n_b samples from the cumulative diameter distribution $\bar{F}_D^n$ in figure 15.3b which results in a total simulated mass denoted by M_{tot}.

The interactions among the rigid bodies itself and between all other static objects in the scene are modeled as unilateral contacts with spatial Coulomb friction in combination with a Newton-type impact law as given in (7.75). The contact parameters: the friction coefficient $\mu_i \in \mathbb{R}_0^+$ and the normal and tangential restitution coefficient $\epsilon_{N,i}$ and $\epsilon_{T,i}$ can be chosen individually for each contact i. However, the restitution coefficients of all contacts are chosen identical to prevent a further potential inconsistency in the total kinetic energy caused by the Newton-type impact law.

The only additional external force exerted on the simulated rigid body assembly is the gravitation given as ${}_{\mathrm{I}}\mathbf{F}_{B_i}^{\mathrm{e}} = m_{B_i}\,{}_{\mathrm{I}}\mathbf{g}$ in (5.43). The mass for a body B_i is given as $m_{B_i} = \rho V_{B_i}$, where V_{B_i} denotes its volume and ρ is the volumetric mass density. It is worth noting that the mass m_{B_i} cancels out in the equation of motion and thus does not change the outcome of the granular simulation. This holds true for the case when the external forces consist only of gravitation and set-valued force laws of normal cone type with a Newton-type impact extension (cf. (7.75)) where all convex sets (force reservoirs) are convex cones. Even if the density of the granular material does not play a role, it is chosen such that the average mass of the rigid bodies is around 1 kg. This is a simple numerical precaution: the values of the inverse of a mass matrix with small entries can become large and computations with large numbers is less accurate due to the logarithmic distribution of floating point numbers on the computer.

The different model parameters used for the conducted simulations are discussed in the further course of this chapter.

Geometry	Experimental Setup	Mechanical Model
granular material *group id:* 0	granular glass beads *Starlitebead®1400* [147]	dynamic rigid bodies with sphere geometries sampled from diameter distribution $\bar{F}_D^n$ as in figure 15.3
channel surface *group id:* 1	flat wood surface covered with sandpaper P180 (FEPA)	static rigid body with a polygonal mesh geometry
sidewalls of channel *group id:* 3	12 mm thick acrylic glass	static rigid bodies, mesh geometries
lower slope *group id:* 1	flat wood surface covered with the backside of a sandpaper (P24)	static rigid body with a half-space geometry
trees *group id:* 2	wooden cylinders with diameter d_{T_i}, axially grooved	static rigid bodies with capsule geometries of diameter d_{T_i}
discharge flap	acrylic glass, not directly mounted on the channel	not modeled

Table 16.1: Comparision between the geometries used in the experimental setup and their representation in the mechanical model in figure 16.1.

(a) **(b)**

Figure 16.2: The used experimental setup as described in [34] at the SLF test facility located in the Flüela valley. Figure (a) and (b) show a side and front view of the channel and slope.

16.1 Performing the Experiment

The experiments were performed on 11$^{\text{th}}$ March 2015 at the WSL Institute for Snow and Avalanche Research SLF. The test facility is located in the Flüela valley close to Davos in Switzerland. The experimental setup is the one used by Carisch in the excellently documented work [34]. To record the granular flow, the high-speed camera S-PRI F1 [10] by AOS Technology AG was used. The camera and the camera lens were the same as used in [34]. The camera was set to record monochrome images at a speed of 1000 frames per second with a resolution of 800×600 pixels per frame. A total mass M_{tot}, the same as used for the simulation, of the granular material *Starlitebead® 1400* was filled into the channel and distributed such that the initial value l_{init} was within a tolerance of $\pm 3\,\text{mm}$ of the value obtained from the simulation of the initial condition of the granular material. We also assured that the shape of the initial material closely resembled the form of a triangular prism. The resulting constrained value h_{init} was then measured and found to be within a tolerance of $\pm 3\,\text{mm}$ to the values obtained from the simulation of the initial condition. In this way, we validated the accuracy of the sampled diameter distribution $\bar{F}_D^n$ used for the simulation to reassure that the number of particles used in the experiment is in the same range as in the simulation. The discharge flap was held by hand and on countdown to zero rapidly turned and moved away from the granular material towards the slope in negative $\mathbf{e}_x^{\text{C}}$ direction. The experimental setup was illuminated by 2 halogen headlights with approximately 200 W. The high-speed camera was triggered manually by hand and recorded approximately 2 seconds of footage. Focusing the camera lens on the surface was not satisfactory and we were not able to remove the blur in the image completely. Furthermore, the only cheap halogen spots available at the facility were not at all supposed to be used for high-speed imaging since their 100 Hz flickering illumination directly causes image intensity fluctuations which render image correlation algorithms as described in section 16.2 less accurate. Similar headlights have been used in [34]. The frequency of the power of the halogen spot, which is mainly an electric resistor, is 100 Hz, wich is twice the frequency of the alternating current provided to the lamp, which is 50 Hz in Switzerland. Another unwanted side effect for further image processing was the specular reflections from the illuminated glass spheres. Unfortunately, a polarization filter which could have reduced this effect was not readily available at the time of performing the experiment. The experiment was setup and performed during one single day due to time limitations. Therefore, there was small room for improvements of the experimental setup and we tried our best to obtain a satisfactory output from the available hardware at the facility. The discussion at the end of section 16.2 summaries all occurred problems during the analysis of the footage and lists improvements for future experiments of this kind.

Three different types of experiments have been performed in total. We recorded 3 chute flows (experiment 1-3) where the slope was covered with a sand paper P24 (FEPA norm) to stop the granular material on the slope. We also recorded two unhindered flows (experiment 4 and 5) where the backside of the sandpaper was used: one flow without trees and one with 3 trees arranged in a triangular pattern on the slope. Two frames of experiment 4 and 5 are shown in figure 16.3.

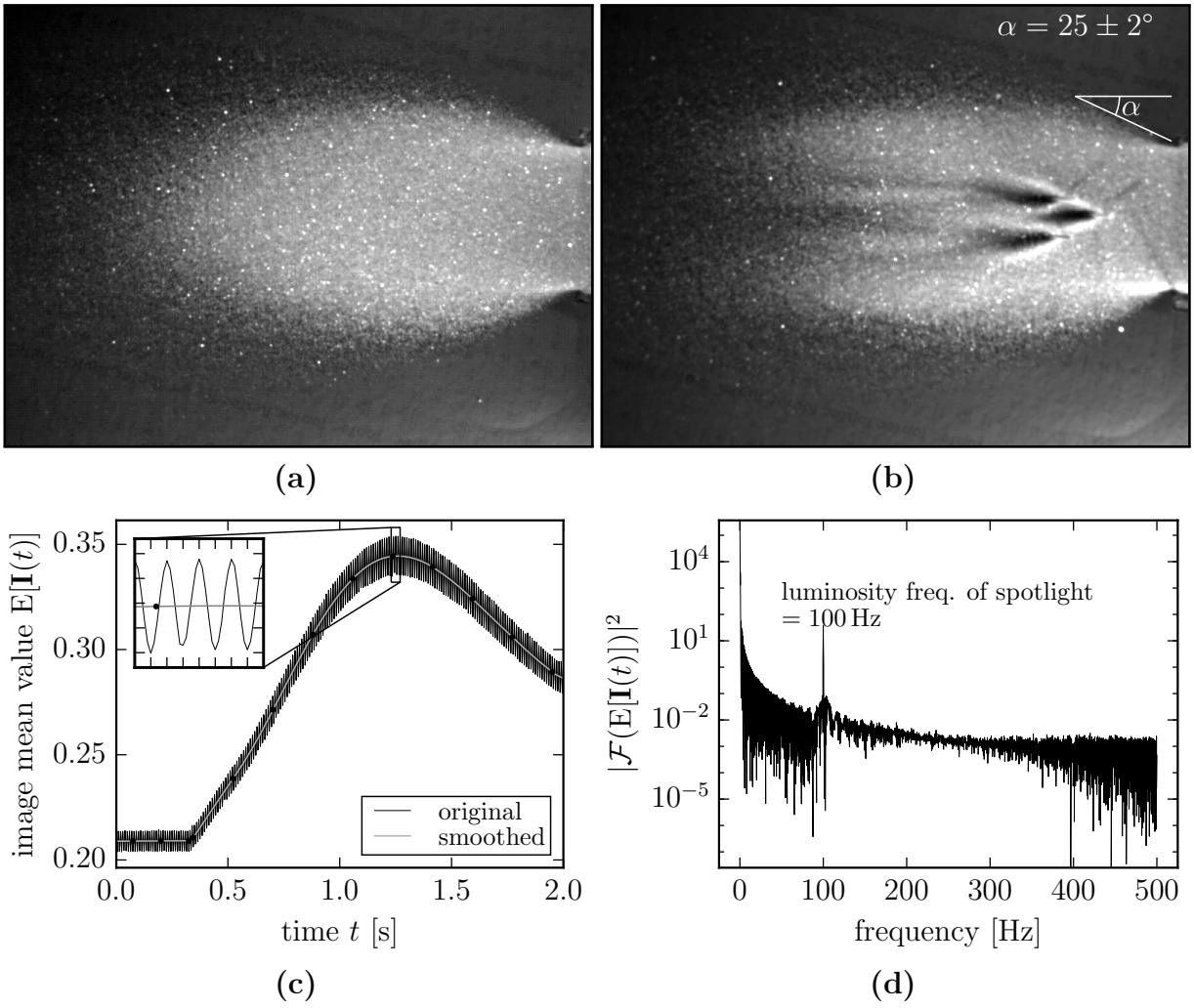


Figure 16.3: Two monochrome video frames at time $t = 0.42\,\mathrm{s}$ for experiment 4 without trees in (a) and experiment 5 in (b) with three trees. Figure (c) shows the image mean intensity value $\mathrm{E}[\mathbf{I}(t)]$ over time of experiment 4 and a smoothed fitting $s(\mathrm{E}[\mathbf{I}(t)])$. Figure (d) shows the discrete frequency spectrum of the original image mean signal $\mathrm{E}[\mathbf{I}(t)]$ in (c).

16.2 Velocity Field Reconstruction

In this section, we briefly describe the velocity field reconstruction from the video footage of the experiments. To extract a gridded velocity field from the video footage, an image correlation tool called *Advection Corrected Correlation Image Velocimetry* (`ACCIV`) developed in [12, 13] has been applied. The extreme blurriness of the images rendered particle tracking algorithms unsuccessful. The reader is referred to [12] for an extensive overview about the strength and weaknesses of several image correlation techniques as well as to [191, 162] for a profound overview about the more mathematical details of these techniques.

The image correlation technique `ACCIV` can capture a full 8-degree-of-freedom projective

transformation of infinitesimal patches of a two-dimensional flow. That is, it can reproduce two-dimensional translation, rotation, shear and distortion in the resulting velocity field. `ACCIV` is an iterative mulit-pass technique of lets say n_p passes, where each pass improves the velocity field by incorporating the results from the previous pass. Each pass $p \in [1\,;\, n_p]$ in `ACCIV` can contain an image sequence $S_p \coloneqq \{\mathbf{I}(t_i), \mathbf{I}(t_{i+1}), \ldots, \mathbf{I}(t_k)\}$ with arbitrary times $t_i < t_{i+1} < \cdots < t_k$ for which image correlation is performed. The two-dimensional tensor $\mathbf{I}(t_i) \coloneqq \mathbf{I}(t_i, m, n)$ denotes the image at time t_i where m and n are the indices for the pixel row and pixel column of the image, respectively.

Based on the user input, `ACCIV` performs image correlation between two images from the set S_p, lets say $\mathbf{I}(t_i)$ and $\mathbf{I}(t_{i+1})$ which are separated by a time period Δt. Image correlation in the first pass $p = 1$ is performed by maximizing the cross-correlation between the intensity of a fixed-size correlation box C_l in the image $\mathbf{I}(t_i)$ with correlation boxes C_s of the same size in the image $\mathbf{I}(t_{i+1})$. The correlation boxes C_s are shifted systematically over a search range around the center point of C_l in the image $\mathbf{I}(t_{i+1})$. In this way, the maximum of the correlation function is searched up to sub-pixel accuracy for a given correlation box C_l. This leads to a velocity vector located at the mean of centers between to adjacent correlation boxes, C_l and C_{l+1}, in the image $\mathbf{I}(t_i)$. Computing the correlation for all correlations boxes in $\mathbf{I}(t_i)$ yields a rough approximation of a gridded velocity field. At the end of each pass p, an outlier rejection method and a hierarchical b-spline smooth fitting procedure is used to combine all results from the correlations of all image pairs in S_p into a gridded velocity field.

After the first pass, each next `ACCIV` pass $p \in [2\,;\, n_p]$ uses the information of the velocity field approximation of the last pass $p - 1$ to advect an image pair from S_p, lets say $\mathbf{I}(t_l)$ and $\mathbf{I}(t_s)$, to a common time point $t_c : t_l \leqslant t_c \leqslant t_s$. The velocity values are computed by maximizing cross-correlation between the images $\mathbf{I}(t_l)$ and $\mathbf{I}(t_s)$ as in the first pass. In our context the image sequences of two successive passes remain the same, that is, $S = S_p = S_{p+1}$. In each successive pass, the correlation box size is, in general, decreased to increase the resolution of the velocity field. For our rather blurred images, 2-3 passes have been used. The settings, which we found to produced good results, are given in table 16.2. In order to minimize noise due to the specular reflections and blur in the images, the correlation box size could not be decreased below ≈ 20 pixels and sufficient smoothing had to be applied in the smooth fitting step to produce a satisfactory result.

Finding parameters such as correlation box size, correlation box shifts, correlation search range and number of passes, to produce the best velocity field approximation is time-consuming and requires a lot of experience. Xylar Asay-Davis, one of the developers of `ACCIV`, was very helpful in this endeavor. The variance of the reconstructed velocity field was quite stable for smaller correlation box sizes, that is, $\leqslant 20\,\mathrm{px}$. That means, that further correlation passes with smaller correlation box sizes only lead to an amplification of the variance of the magnitude of the velocity field and that the mean values at each point of the magnitude did not vary much. The mean of the correlation velocity uncertainty as defined in [12] for experiment 5 is extremely high during the first 0.5 seconds when the glass beads start to appear in the image. This observation is also confirmed by the comparison in section 17.2. After the first 0.5 seconds, the mean of the correlation velocity uncertainty stabilizes around $0.25\,\mathrm{m\,s^{-1}}$ which is satisfactory for the given quality of the

video footage. The correlation velocity uncertainty is only a measure of how good the advection and image correlation performed and gives an indication of the quality of the reconstructed velocity field.

In the following, we list improvements in descending order of importance for future high-speed video recordings of granular flow experiments:

- The dynamic and static specular reflections from the glass beads, which occur in the image sequence, were a real issue for the correlation algorithm to produce accurate results. The static specular reflections at certain locations where so bright that the correlation algorithm returned pixel displacements around zero in these region of the image. The negative effect of the dynamic specular reflections on the resulting vector field was less severe than for the static ones and only affected the variance of the resulting velocity field. It is highly recommended to use a polarization filter and, if possible, to mat the glass beads in some way to reduce their specular reflections.
- Illuminate the granular material with flicker-free lamps. A metal-halide gas discharge medium arc-length lamp in combination with electric ballasts are not prone to flickering.
- Set up the camera and lens at a distance from the slope such that the plane of interest can be focused properly.
- The images of the video footage should be stored in a lossless image format such as `JPEG2000`, `TIFF` or `BMP`. The files were accidentally saved in high-quality in the lossy compression format `JPEG` which is a bad starting point for applying image correlation algorithms.

An in-depth discussion about the main sources of errors in granular particle image velocimetry is given in [49].

To compute time-dependent velocity fields from all frames of the video footage, a special job has been built within the developed `HPCJobConfigurator` [140] to leverage the parallel execution on a high-performance cluster. A set of video frames, for which a velocity field should be extracted, is assigend to each process. Each process then executes a tool pipeline consisting of two tools. The first tool is the image processing task which subtracts the background from the image, shifts all pixel intensities by the value $d = \mathrm{E}[\mathbf{I}(t_i)] - s(\mathrm{E}[\mathbf{I}(t_i)])$ as shown in figure 16.3c to remove the intensity flickering from the halogen spotlights, extracts some binary mask and boundary contour of the granular material and finally outputs the `HDF5` file which is the input file for the second tool, that is, the correlator. The correlator simply performs all `ACCIV` passes for each frame. An `ACCIV` job with 3 correlation passes for 1656 frames and 4 images per frame took approximately 50 minutes on 552 processors.

pass i	image set S_i	correleation box size [px]	correlation search range $[x_{min}, x_{max}]$, $[y_{min}, y_{max}]$ [px]	correlation box stride [px]
1	$\{\mathbf{I}(t_i), \ldots, \mathbf{I}(t_{i+3})\}$	$[32, 32]$	$[-5, -1], [-3, -3]$	$[16, 16]$
2	$\{\mathbf{I}(t_i), \ldots, \mathbf{I}(t_{i+3})\}$	$[24, 24]$	$[-5, -5], [-5, -5]$	$[8, 8]$
3	$\{\mathbf{I}(t_i), \ldots, \mathbf{I}(t_{i+3})\}$	$[20, 20]$	$[-5, -5], [-5, -5]$	$[4, 4]$

Table 16.2: Image correlation parameters from [3]file for the three correlation passes over 4 consecutive images for the application `ACCIV` developed in [12].

16.3 Simulation Studies

The parameters of the mechanical model in figure 16.1 have been varied over a sequence of rigid body simulations of the granular flow. These simulations are compared with the experiments in a further step. The parameters of the mechanical model consist of the ones shown in figure 16.1 and the contact parameters for each possible combination of the group ids in table 16.1. The initial condition, namely the situation where the granular material is at rest after filling it into the channel above the discharge flap, is simulated with Moreaus time-stepping scheme in algorithm 8.1 using the drift correction algorithm explained in section 8.2. This ensures that the penetration of the glass beads after the settlement is as low as possible to provide a good initial condition for the further granular flow simulations. The time step for the simulation of the initial condition could be chosen large, that is, $\Delta t \in [10^{-3}, 10^{-2}]$, as we were only interested in the low-penetrated simulation state at the end of the settlement. For the main simulation, namely the chute flow, Moreau's time-stepping scheme has been used as well but without a drift correction due to its undesired side-effect of a potential increase in the total energy of the granular material. An extract of the main parameters and settings specified in the scene file used to simulate the chute flows is listed in table 16.3.

Two simulation studies A and B, each consisting of 15 chute flow simulations, were performed where only the friction parameter μ between the glass beads was varied, that is, $\mu \in S$ as given in table 16.3. Simulation study A corresponds to experiment 4 where the flow of the glass beads on the slope is unhindered (see figure 16.3b). Simulation study B is identical to A but includes three trees positioned in a triangular pattern on the slope and corresponds to experiment 5. The performance of the simulation with the `GRSFSimMPI` application is visualized in figure 16.4 and shows the time evolution of the kd-tree topology as well as the computation time per time step which is split into the time spent for the communication, for the collision detection and for the solution of the contact problem. The amount of data output of a single simulation summed up to approximately 100 GB for the binary simulation file.

solver settings for simulation studies A and B

time-stepper type :	Moreau without drift correction (algorithm 8.1)
time step Δt :	2×10^{-4} s
time steps/recorded state :	10
inclusion solver type :	SOR Prox iteration per contact in (8.37, 8.38)
inclusion solver parameters :	$\alpha = 1.5$, **R** matrix strategy : max (cf. sec. 8.3.1), $maxIter = 1000$
number of processes :	384
total computation time :	12 h
memory per process :	1024 MB
high-performance cluster :	Euler at ETH Zurich
load balancing technique :	kd-tree with approximate MVBB (cf. sec. 10.3)
topology rebuilding heuristic:	every 250^{th} time step (cf. sec. 11.1)

model parameters for simulation studies A and B

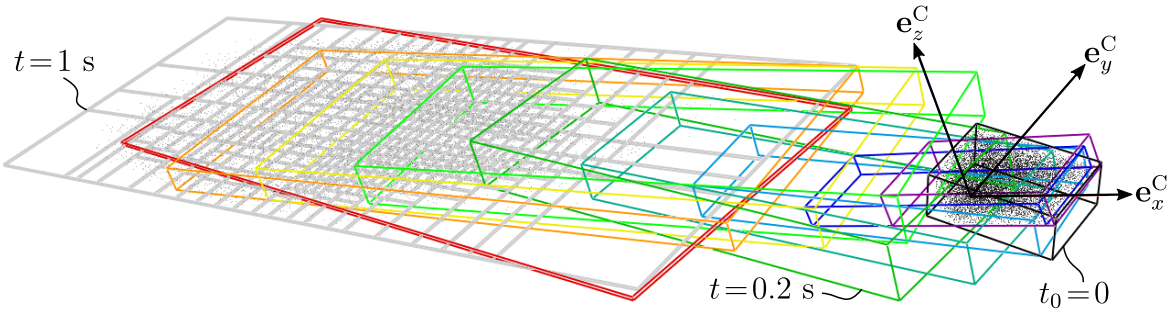
total number of glass beads n_b:	1×10^6
total mass of glass beads M_{tot}:	4.525×10^{-6} kg (cf. sec. 15.4)
gravity g :	$9.81\,\mathrm{m\,s^{-2}}$
start distances x_{init}, l_{init}, h_{init} :	29.6 cm, $\approx$ 238 mm , $\approx$ 116 mm
channel width w :	10 cm
angle α :	30°
angle β :	70°
pixels/meter on visual plane :	967.5931
tree thickness d_{T_i} :	5 mm
visual plane point ${}_S\mathbf{r}_{OF}$:	$[-783,\ -311,\ 0]^\top$ px
visual plane point ${}_S\mathbf{r}_{OG}$:	$[17,\ 289,\ 0]^\top$ px
tree positions ${}_S\mathbf{r}_{T_0}$, ${}_S\mathbf{r}_{T_1}$, ${}_S\mathbf{r}_{T_2}$ for study B :	$[-0.1254, 0.004, 0]$ m, $[-0.1794, 0.0277, 0]$ m, $[0.1829,\ -0.02704,\ 0]$ m

contact parameters for simulation study A and B

group id pair (cf. table 16.1)	contact model	model parameters
$\{0, 0\}$	UCF	μ varied over S, $\epsilon_N = 0$, $\epsilon_T = 0$
$\{0, 1\}$	UCF	$\mu = 0.5,\ \epsilon_N = 0,\ \epsilon_T = 0$
$\{0, 2\}$	UCF	$\mu = 0.3,\ \epsilon_N = 0,\ \epsilon_T = 0$
$\{0, 3\}$	UCF	$\mu = 0.1,\ \epsilon_N = 0,\ \epsilon_T = 0$

UCF : unilateral contact with Coulomb friction
$S \coloneqq \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 2, 2.5, 3\}$

Table 16.3: Extract of simulation parameters of the scene file [2]file for the simulation studies of the mechanical model in figure 16.1.



(a) Time evolution of kd-tree topologies. See figure 16.1.

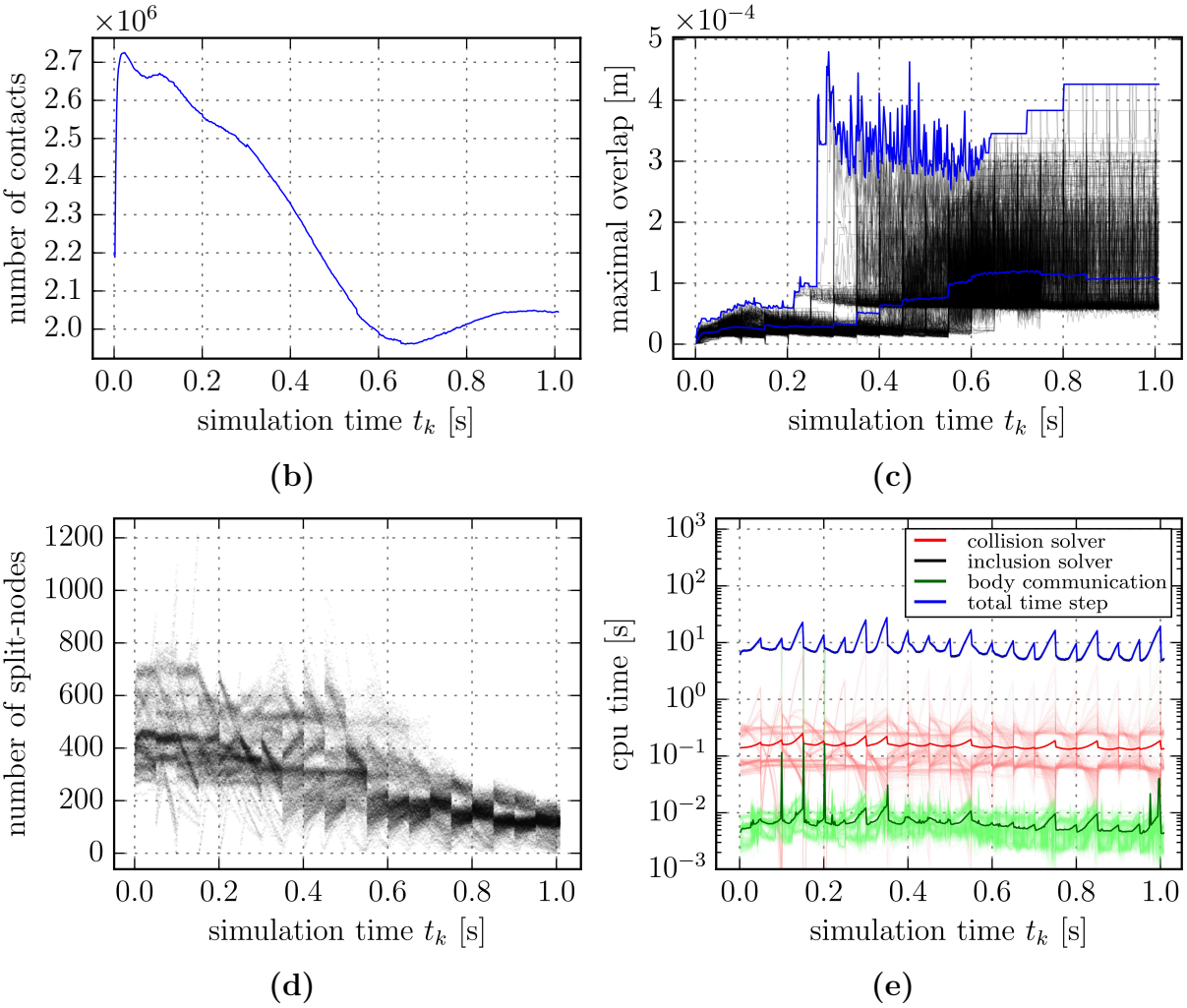


Figure 16.4: Performance measures of 384 processes for the simulation B-0 of the simulation study B with parameters given in table 16.3. The average computation time of a time step could be kept approximately constant due to the topology rebuilding steps during the simulation shown in (a). The maximal overlap is below 4.5×10^{-4} m. The computation time was set to 12 h which resulted in approximately 5000 time steps.

Chapter 17

Comparison of Simulation Studies with Experiments

In this section, the chute flow experiments are compared against the simulation studies. The discussion will mainly focus on comparisons between the velocity fields and related quantities and will be rather qualitative than quantitative. The reason lies in the accuracy of the velocity reconstruction of the video footage obtained from the experiments which is described in more detail in section 16.2.

17.1 Cumulative Body Count

To sync the time between the experiments and the simulation studies A and B discussed in section 16.3, there was the need to extract the start time from the simulations when the first glass beads enter the slope below the channel. We extracted the number of bodies $n_{b,S}(t)$ overlapping the half-space $H_S^+ \coloneqq \{\mathbf{x} \in \mathbb{E}^3 \mid (\mathbf{x} \,|\, \text{-}\mathbf{e}_x^{\mathrm{S}}) \geqslant 0\}$ defined by the normal vector $-\mathbf{e}_x^{\mathrm{S}}$ at the origin O in figure 16.1. This data extraction task has been performed with the analyzer tool of the `GRSFConverter` and a similar execution graph as shown in figure 14.2.

The normalized cumulative body count distribution $F^n(t)$ is given as

$$F^{n_{b,S}}(t) = \frac{1}{n_b} \int_{t_0=0}^{t} \mathrm{d}n_{b,S}(\tau) = \frac{n_{b,S}(t)}{n_b}, \tag{17.1}$$

where n_b denotes the total number of glass beads. The derivative of the above yields the density over time, namely the normalized flow count into the half-space H_S^+, this is

$$f^{n_{b,S}}(t) = \frac{\mathrm{d}}{\mathrm{d}t} \frac{n_{b,S}(t)}{n_b} \approx \frac{n_{b,S}(t_{k+1}) - n_{b,S}(t_k)}{n_b \Delta t}, \tag{17.2}$$

where t_k denotes the discrete simulation time and Δt the time step, see table 16.3. The quantity in (17.1) and (17.2) are shown in figure 17.1 for all 15 simulations of the simulation study B. The flow duration Δt_f depending on the friction coefficient is depicted

in 17.2a. The flow duration Δt_f is defined as

$$\Delta t_f \coloneqq t_{0.95} - t_{0.05} = Q^{n_b,S}(0.95) - Q^{n_b,S}(0.05), \tag{17.3}$$

which is the time duration until 90 % of the bodies have entered the half-space H_S^+. The quantile function $Q^{n_b,S}$ is given in (15.3).

Figure 17.1 shows that the time duration of the simulation B-0 with no friction, that is, $\mu = 0$, compared to the simulation B-14 with friction coefficient $\mu = 3.0$, is doubled. Figure 17.2a shows that the flow duration Δt_f seems to stagnate for higher friction coefficients. It is important to note that the iteration for solving the contact problem is terminated after *maxIter* iterations for all time-steps in each simulation (see table 16.3). High friction coefficients make the contact problem more difficult to solve and the stagnation in figure 17.2a might be a numerical artifact which could only be unraveled if the contact problem is solved to a certain precision. However, this is currently infeasible due to the undetermined run time of the simulation. An additional simulation $\overline{\text{B}}$-14 similar to simulation B-14 but with 20 times more SOR Prox iterations (20'000 global iterations) has been conducted to investigate any difference in the count flow. The flow count of simulation $\overline{\text{B}}$-14 in figure 17.1b shows that the granular flow is not much affected by the increased number of iterations. However, a proper argument can only be given when the residual of the SOR Prox iteration over the time-steps is properly investigated, which should be considered in future work.

The gray bars in figure 17.1b mark the approximate start and end time of experiment 5 which have been extracted manually from the video footage. These marks show that a rigid body simulation with friction coefficient $\mu \in [0.8, 1.5]$ provides enough dissipation such that the flow duration comes close to the one of experiment 5.

Figure 17.2b is interesting as it shows that the time evolution of the number of contacts is decreased over the whole simulation period for higher friction coefficients. This finding is not very intuitive at first and strongly suggest the study of the velocity fluctuations in the flow (also called granular temperature) in future work.

The next section tries to validate the findings of the above results by a qualitative comparison of the velocity field and flow shapes between the simulations and the experiment.

17.2 Qualitative Velocity Comparison

By using the gridder tool of the `GRSFConverter`, gridded velocity fields for simulation study B have been extracted which are identical in shape and position to the reconstructed gridded velocity field from the image correlation of experiment 5. The two-dimensional velocity field is denoted by $\mathbf{V} \in \mathbb{R}^{g_x \times g_y \times 2}$ (cf. table 16.3) and is the result of projecting the three-dimensional gridded velocity field constructed from bodies at the top of the chute flow onto the visual plane in figure 16.1. The magnitude is computed as

$$\mathbf{V}_{\mathrm{M},(i,j)} = \sqrt{\mathbf{V}_{(i,j,1)}{}^2 + \mathbf{V}_{(i,j,2)}{}^2}\ . \tag{17.4}$$

The magnitude of the two-dimensional projected velocity field from simulation B-9 and the corresponding reconstructed velocity field from the image correlation is shown in figure 17.3b and 17.3a. The overlap ratio of the simulation and the experiment is defined as

$$\text{overlap ratio} \coloneqq \frac{\sum_{i,j}(\mathbf{B}_\text{S} \wedge \mathbf{B}_\text{E})_{(i,j)}}{\sum_{i,j}(\mathbf{B}_\text{S} \vee \mathbf{B}_\text{E})_{(i,j)}}, \tag{17.5}$$

where $\mathbf{B}_\text{S}, \mathbf{B}_\text{E} \in \{0,1\}^{g_x \times g_y}$ denote the extracted binary masks of the simulation and experiment and the operator $\wedge$ and $\vee$ denote the element-wise logical *and* and *or* operations, respectively. The binary mask $\mathbf{B}_\text{E}$ used for the velocity magnitude in figure 17.3a was obtained by using thresholding techniques which gave satisfactory but not brilliant results. The overlap ratio shown in figure 17.4a shows the tendency of a better match (higher overlap ratio) between simulation and experiment for higher friction coefficients.

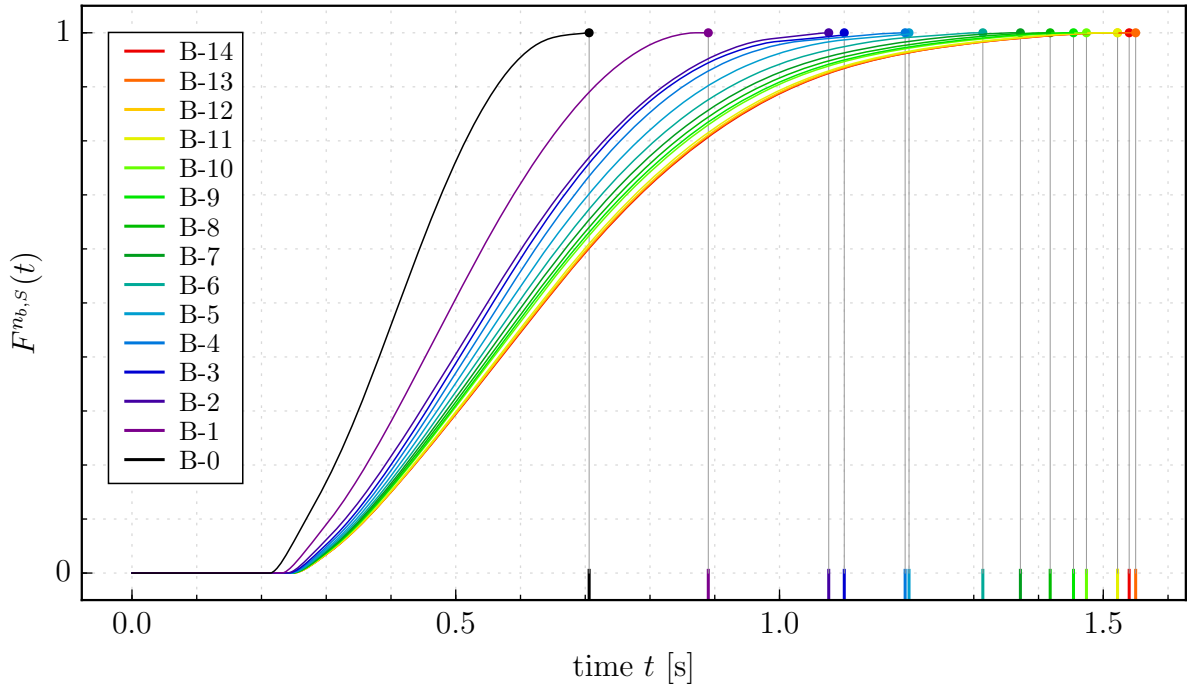
The root mean square error of two scalar tensors $\mathbf{X} \in \mathbb{R}^{m \times n}$ and $\mathbf{Y} \in \mathbb{R}^{m \times n}$ is computed as

$$\text{RMSE}(\mathbf{X}, \mathbf{Y}) \coloneqq \sqrt{\underset{i,j}{\text{E}}\big[(\mathbf{X}_{(i,j)} - \mathbf{Y}_{(i,j)})^2\big]}, \tag{17.6}$$

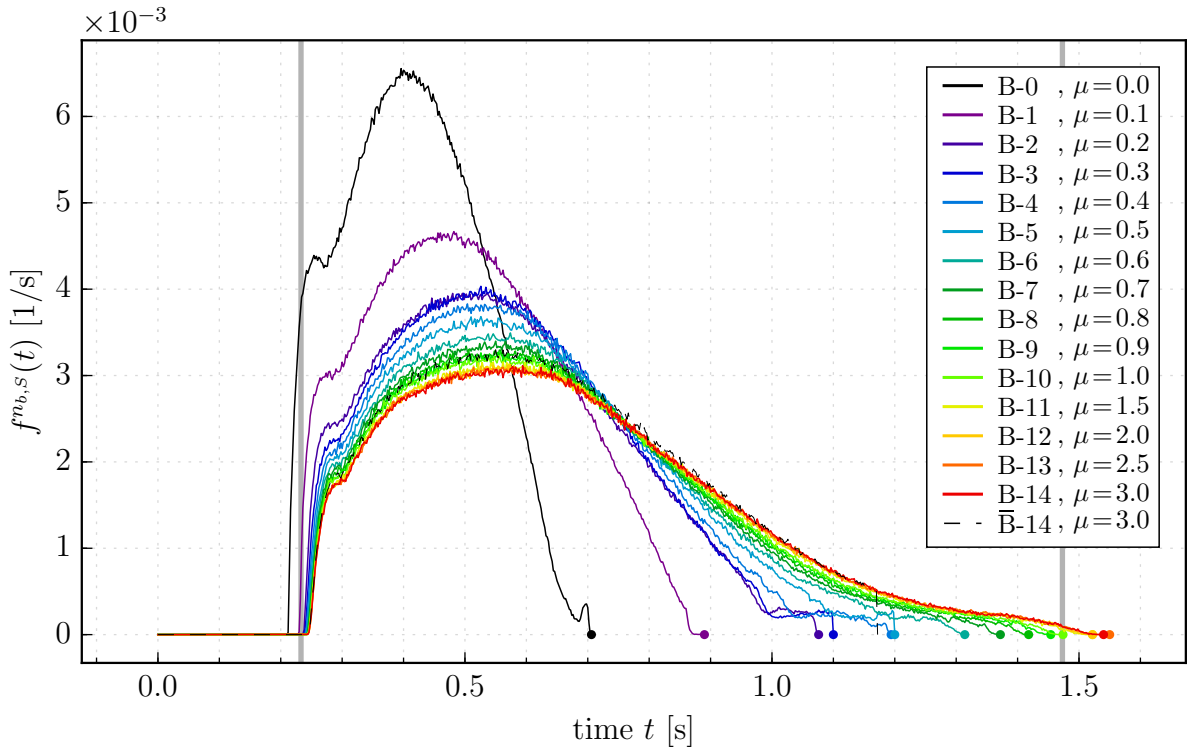
where the expected value $\text{E}_{i,j}$ is computed over the values of the tensor of the squared difference. Figure 17.4b shows that the root mean square error of the velocity magnitude between experiment 5 and the simulations decreases for higher friction coefficients. It has to be noted that the results presented in figure 17.4a, 17.4b depend on the extracted binary mask from the experiment.

Furthermore, the high RMSE error in figure 17.4b at the beginning of the flow is explained by the bad velocity reconstruction at the beginning of the video footage. Although the realization of the experiment and the velocity field reconstruction were far from optimal, a visual comparison of figure 17.3a with 17.3b shows that the solutions are not far apart. A video analysis of all comparisons between simulation B-0 – B-14 and the experiment showed that simulation B-9 with friction coefficient $\mu = 0.9$ provides the best match also with regard to the flow shapes developed after all particles passed the triangular tree pattern. The three-dimensional renderings (see figure 14.1) also showed that the front regime of the chute flow becomes more agitated with increasing friction coefficient despite the use of a completely inelastic impact. A similar observation is made for chute flow experiments in a channel in [31, section 3.4].

It is promising that conducting more simulations with increased accuracy for the contact resolution and with different coefficient of frictions for the slope and channel as well as better experimental results would lead to an even better agreement between experiment and simulation.

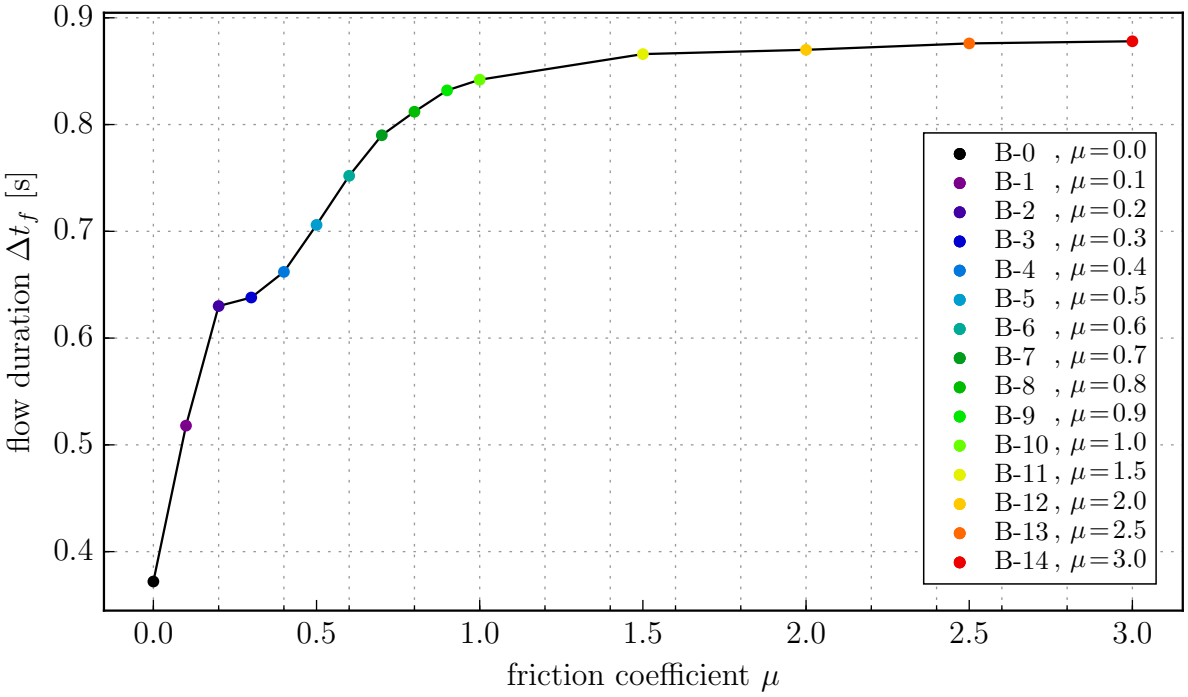


(a) Cumulative count distribution $F^{n_{b,s}}(t)$ of bodies overlapping the half-space H_S^+ defined by the normal vector $-\mathbf{e}_y$ in figure 16.1.

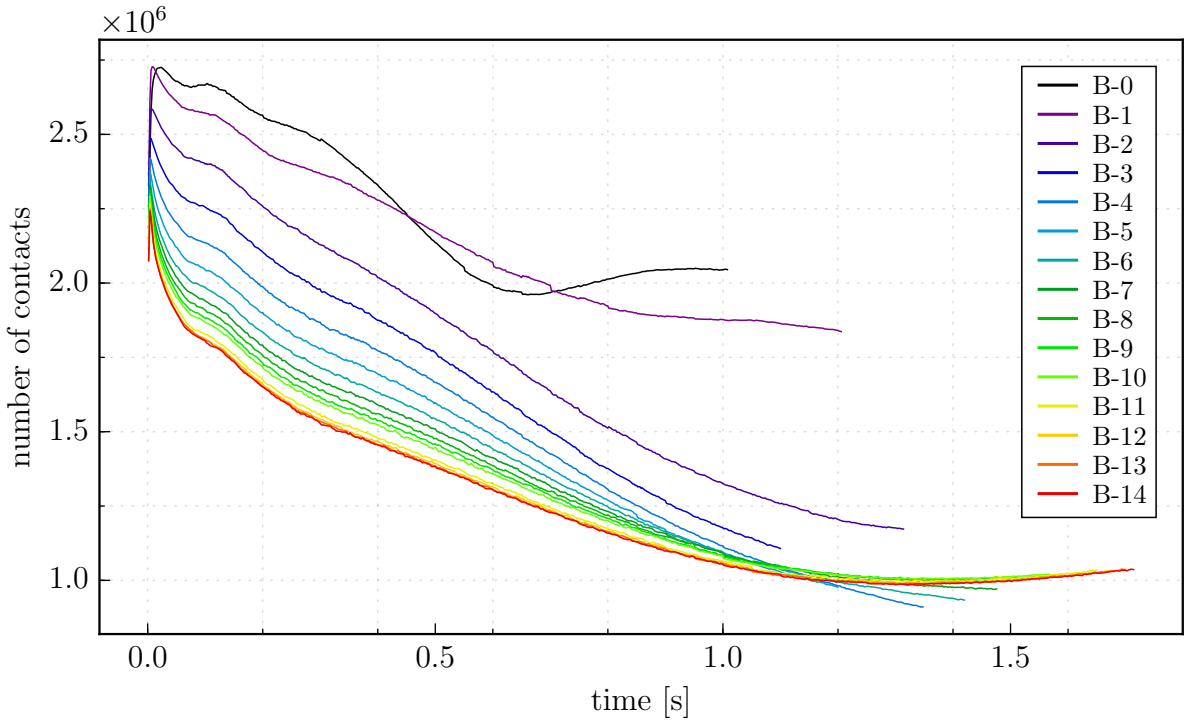


(b) The derivative of $F^{n_{b,s}}(t)$, that is, the flow count $f^{n_{b,s}}(t)$ into the half-space H_S^+.

Figure 17.1: Body count analysis of the simulation study B in table 16.3.

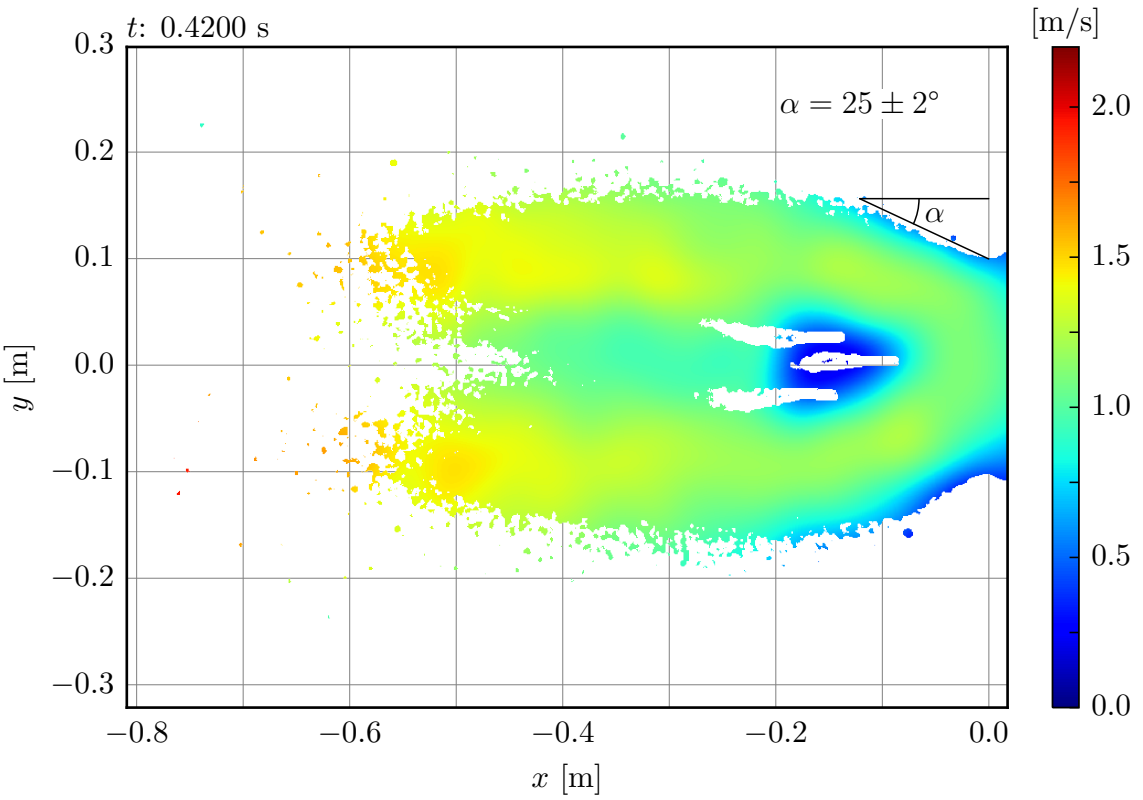


(a) The flow duration depending on the friction coefficient between the glass beads.

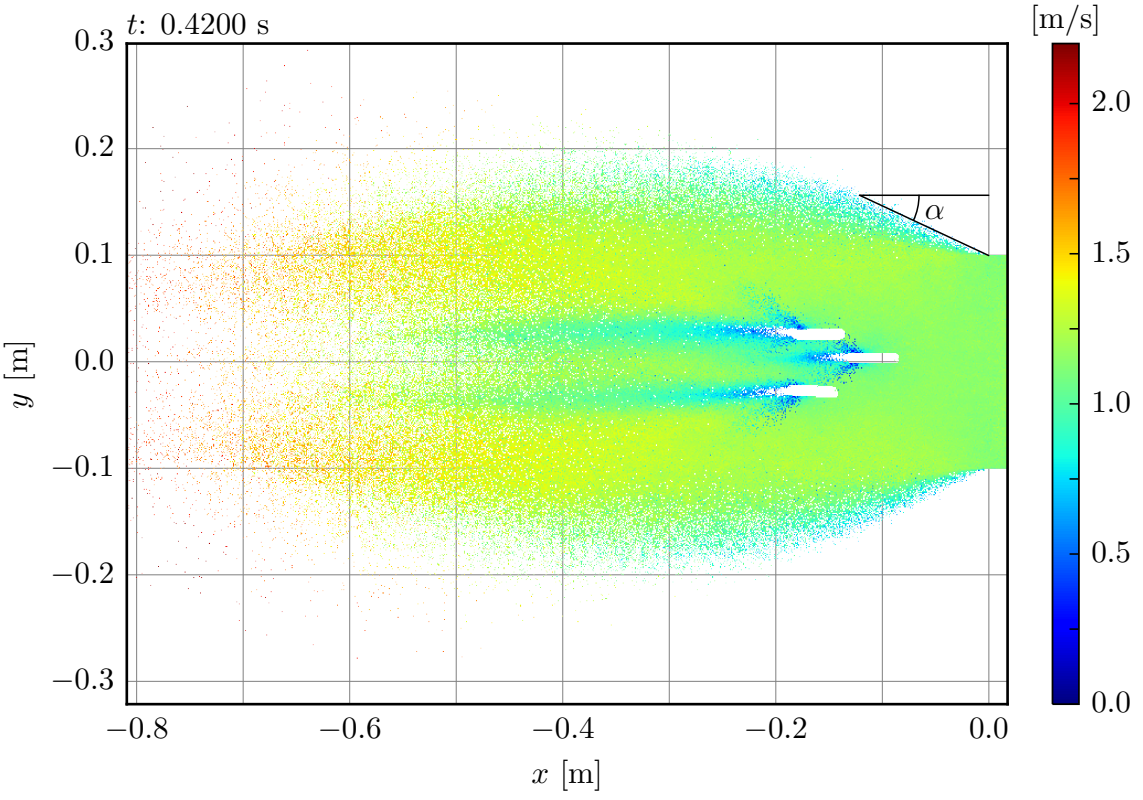


(b) Time-evolution of number of contacts.

Figure 17.2: Body count analysis of the simulation study B in table 16.3.

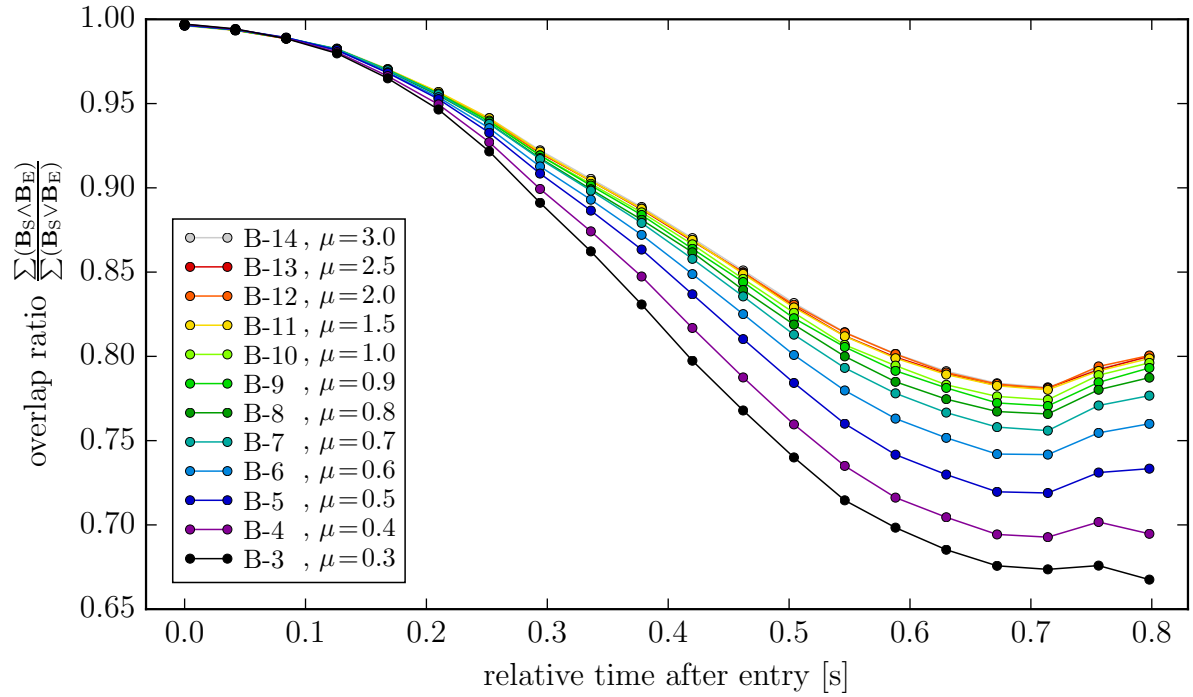


(a) Reconstructed velocity magnitude $\mathbf{V}_{\mathrm{E,M}}$ from experiment 5 (see settings 16.2). The angle α is determined in figure 16.3b.

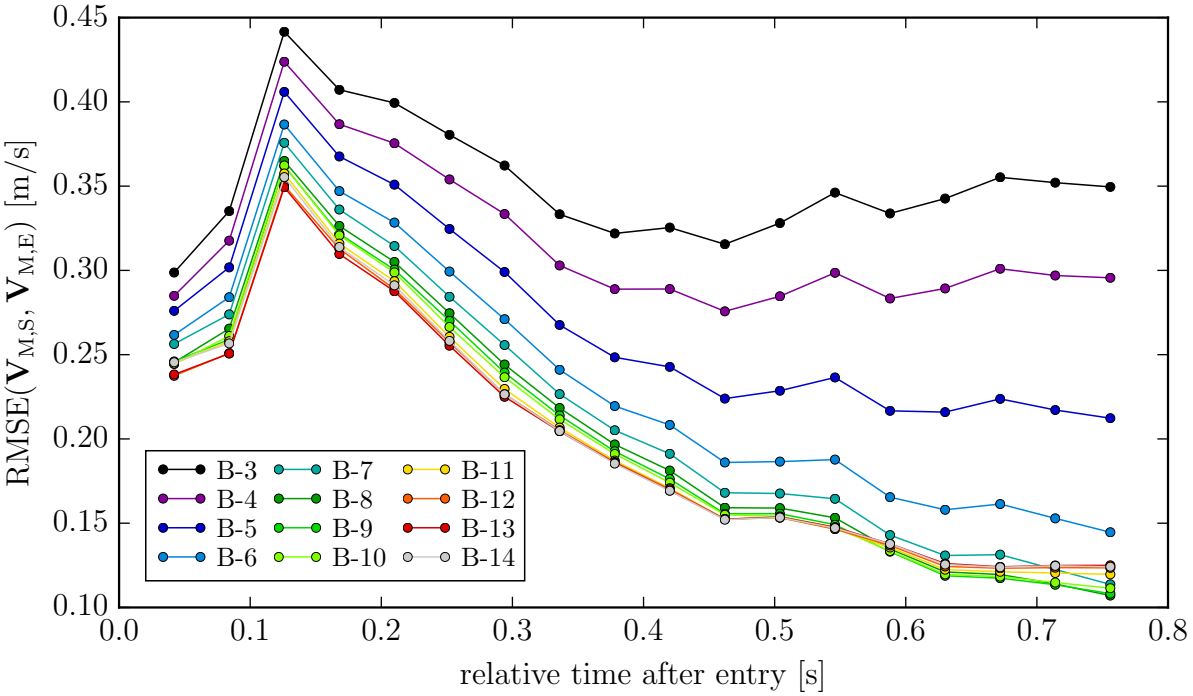


(b) Gridded velocity magnitude $\mathbf{V}_{\mathrm{S,M}}$ of simulation B-9.

Figure 17.3: Comparison of velocity magnitude between study B-9 (subscript S, see table 16.3) and experiment 5 (subscript E, see figure 16.3b).



(a) Overlap ratio (17.5) between the binary mask of the simulation $\mathbf{B}_S$ and the experiment $\mathbf{B}_E$.



(b) Root mean square error (17.6) of the velocity magnitude of the simulations and the experiment.

Figure 17.4: Comparison of velocity magnitude between study B (subscript S, see table 16.3) and experiment 5 (subscript E, see figure 16.3b).

Chapter 18

Conclusion and Outlook

This work aims at modeling granular materials by large-scale rigid multi-body systems and is a contribution in the field of granular dynamics. This thesis is structured in three parts. Whereas the first part introduces all necessary concepts from mechanics, convex analysis and numerics, the second part applies these concepts for the parallel simulation of a granular material by means of a developed modern software framework encompassing simulation, visualization and data extraction tools. The last part discusses the simulation and partial validation of a chute flow experiment which is used at the WSL Institute for Snow and Avalanche Research (SLF) in Davos, Switzerland.

A granular material is described in this work within the framework of non-smooth rigid body dynamics which allows to formulate internal interactions between bodies by set-valued force laws. By applying the two most fundamental and powerful axioms in mechanics, the principle of virtual work and the variational law of interaction, the mechanical formulation of a rigid body is derived in a concise and systematic way starting from a continuous body embedded in a three-dimensional Euclidean space. To obtain a rigid body formulation using quaternions for the parametrization of the rotation, the equations of motion of the scalable body are derived in a first step and restricted in a later step to the constraint of isometry quintessential for the rigid body. In contrast to using the Lagrangian formalism as shown in [108, 170], the presented derivation of the scalable body by the principle of virtual work is elegant, more concise and simpler and serves as an educational example for a more rounded understanding of the nontrivial mechanical principles.

The interactions between the rigid bodies in a granular material are modeled in this thesis with set-valued force laws by using concepts from convex analysis. The unilateral contact with Coulomb friction, formulated by a normal cone to a convex set, has been proven to be a particularly useful contact law to model the dissipative behavior of a granular material and to provide a concise formulation of the impenetrability condition of its contained particles. Four mathematically identical formulations of this model have been discussed with special regard to convex optimization. Casting the solution of a mechanical system with contact laws of normal cone type into a constrained optimization problem, which is convex for only unilateral contacts, is useful for understanding the discussed iterative solution methods for the contact problem during the time-stepping

procedure. Moreau's time-stepping efficient, explicit scheme is used for the numerical integration of the equation of measures which contains both the smooth and non-smooth motion of a large-scale rigid multi-body system. All set-valued contact laws are extended with a Newton-type impact law which consequently renders the velocities discontinuous. An impact law provides a second dissipation mechanism in the case of partially elastic impacts. A inelastic Newton-type impact has been adopted for the simulation studies in the last part.

Simulation of granular media consisting of many rigid bodies is a large open research field in mechanics, computer science and computer graphics. Using parallelization techniques such as the Message Passing Interface MPI on high-performance distributed systems or writing specialized parallel code for graphics processing units are two ways of leveraging enough computational power to solve systems consisting of millions of bodies in a meaningful time. The open-source software framework `GRSF` developed in this thesis provides a quality-conscious and efficient implementation for simulating granular materials on high-performance distributed systems. Two domain decomposition methods, the grid decomposition and kd-tree decomposition, have been presented, implemented and successfully tested. The implemented simple load balancing strategy has been supported by the open-source software library `ApproxMVBB`, also developed in the course of this thesis. The body communication during the time-stepping procedure, inspired by [75], plays a substantial role for the parallel simulation and its intricate book-keeping process has been discussed in detail.

Furthermore, another contribution is given in the form of a proper and concise discussion on the mass-splitting method studied in [196, 7, 192] which builds the fundamental concept of the parallelization procedure of the rigid body simulation. Our presented analytical solution of the bilateral constraints which are constructed from the virtual splitting of domain overlapping bodies, called *split-nodes*, is properly derived and shows that the solution of a global contact problem can be split into local contact iterations in combination with a simple averaging of the velocities of all domain overlapping bodies.

We developed several data extraction tools such as a generic execution graph network which allows for all sorts of data extraction tasks such as producing input data for rendering a simulated scene. The job preparation and submission workflow on the high-performance cluster was structured and simplified a great deal by the open-source job configurator `HPCJobConfigurator` developed in the course of this work. Its application is independent of any particular cluster. In this work, the Euler cluster at ETH Zurich in Switzerland has been used.

In the last part, we made a first step in validating a chute flow experiment performed at the SLF in Davos, Switzerland, by comparing velocity fields extracted from experimental data with gridded data from the simulation. The friction coefficient has been varied in a simulation study for two different setups, one with and one without tree models. The results showed that a friction coeffiecient $\mu \in [0.8, 1.5]$ provides enough dissipation such that the root mean square error of the velocity magnitude is approximately $0.15\,\mathrm{m\,s^{-1}}$. The author is aware that the validation could only be performed qualitatively instead of quantitatively, mainly because of the low quality of the recorded video footage. Furthermore, it should be pointed out that conducting the experiment was far from optimal and

was performed in a single day due to time constraints and the announced discontinuation of the experimental setup by the SLF. The domain decomposition methods, in particular the kd-tree decomposition, and the presented simple load balancing strategy were especially beneficial for the conducted simulation study as it increased the performance of the time-stepping procedure during the rapid distribution of the rigid spheres on the slope.

In the following, we give a brief outlook on future work related to the presented three parts: the mechanical theory, the parallel software implementation and the validation of granular chute flow experiments.

Outlook

From a theoretical and numerical perspective, one urgent need is to improve the convergence behavior of the iterative solution technique of the contact problem by applying better numerical methods from convex optimization, in particular improved efficient first-order methods. One step in this direction has already been conducted in recent research [102]. Furthermore, robust and efficient constraint stabilization and drift correction techniques is another open research field. Drift correction is important for granular simulations since an increasing penetration leads to all sorts of unpleasant problems, not to mention the growth in the number of contacts. It is also promising to investigate time discretization schemes derived from variational principles which may lay the foundation of obtaining time-stepping schemes with a better trade-off between averaged energy conservation or average constraint violation and computational efficiency.

From a software perspective, the `GRS` framework is far from complete in respect to a full-fledged rigid body engine. However, further improvements include mainly all mentioned aspects which should as well be implemented and validated in further work. Another important open question is how to improve the communication overhead due to the applied mass-splitting procedure during the contact solution phase? Further work is also devoted to scalability testing and other performance analyses of parallel simulations computed by the `GRS` framework. This has not been properly treated at the time of writing.

From a validation perspective in the field of experimental mechanics, there are lots of open questions which have only been partially answered in this thesis. The realization of a well-prepared experiment and a proper data evaluation is extremely time consuming and almost encompasses a thesis by itself. The velocity comparisons presented in chapter 17 showed that the velocity fluctuations are of utmost interest which might explain the decreased count in the number of contacts for higher friction coefficients. Furthermore, it is intriguing to investigate and compare a granular chute flow simulation with a hindered chute flow experiment as performed in experiment 1–3 where the slope has been covered with a sand paper. This, however, requires the implementation of some special collision detection routines in the `GRS` framework which supports polygonal mesh patches with which the slope can be tiled.

Another field which has been completely abandoned in the course of this thesis is the field of fluid dynamics which has contributed the most to the research of granular materials in the past, especially in the field of avalanche research. In the authors opinion, a large-scale

rigid multi-body system with frictional contacts and possible other force laws is a better model to study impact regimes and velocity fluctuations in dry granular materials in the future because such a model has less model parameters and provides more detailed insight since the modeling scale is on the particle interaction level. The experimental results and references in [31] should be considered in future work to compare velocity fluctuations obtained from similar granular simulations as presented in this thesis.

The validation of the chute flow experiments presented in this thesis lacks also the study of dual quantities such as contact forces or stresses. In a first step, the focus was mainly on primal quantities such as velocities and displacements because the concept of force in mechanics, despite its simple mathematical model as a multi-linear functional, is a very difficult concept to grasp. However, data of force quantities would give more insight into understanding intriguing aspects in granular dynamics. In regard to the chute flow experiments such aspects may include the study of force networks, impact regimes, pressure distributions on the tree models, the described decrease in the number of contacts for high friction coefficients and the potentially associated velocity fluctuations. The stress distribution in granular materials is a very interesting aspect. However, to successfully study stress distributions in a granular material, a proper understanding of the stress tensor itself and its application to granular materials is needed in a first step. Recent visionary work has been presented recently by Moreau [118]. Another difficulty lies in the comparison of force quantities with force measurements from experiments which involves another layer of modeling abstraction since there exists no sensor which outputs force per se. The mechanism behind a force sensor is always the evaluation of a mechanical model with measured primal quantities such as time, displacements or velocities. In a first step, future work in this direction includes the study of contact percussions between rigid bodies.

A proper comparison between discrete element formulations, which are basically large-scale rigid body systems where the interactions are purely modeled as impressed forces, is another topic for future research. Studies in this direction have been conducted in [172]. A time-stepping scheme of a discrete element method which models a dry granular material is in general much simpler and, in contrast to the presented approach in this thesis, does not involve the solution of a global coupled contact problem in each time step. This fact, however, does not make them less applicable for dry or also wet granular flows. The presented non-smooth dynamics approach is a neat framework which allows elegant, mathematical formulations to accurately model impenetrability and stick-slip phenomena without being prone to stability issues which lead to high stiffness coefficients in the contact laws in a discrete element formulation. This advantage, however, comes at the prize of a higher computational burden which also entails numerical problems such as the lack of fast convergence, not to mention the issue of highly linear dependent contact problems. It is also interesting to study the influence of different parallelization methods for the contact problem on the motion of a granular material in a statistical sense.

Since granular particles are never completely rigid in nature, another open question is how a certain additional degree of freedom, allowing for a certain deformation of the body, changes the outcome of a chute flow simulation. Doing so, results in a large-scale nonlinear continuum dynamics problem and additional constitutive force laws need to be formulated

for the internal virtual work contributions of the deformable degrees of freedom. Using millions of deformable bodies is currently infeasible due to an extremely complex collision detection which would need to deal with deforming volumetric polygonal meshes which can even become non-convex over time. The increased computational costs for integrating the spatially discretized deformable bodies is another challenge.

A set-valued force law which models an adhesive frictional contact law is another interesting topic for the modeling of granular materials. A concise formulation of the problem is difficult since the set-valued force law of a unilateral contact with adhesion is not monotone (cf. first chapter in [201]) which renders its treatise in the framework of convex analysis problematic. Research in this direction has already been conducted in [164, 44, 59].

The endeavor of creating a granular material model consisting of a body formulation which allows for a smooth transition between rigid and deformable motion is intriguing. Such a model may provide an even better approximation for granular matter in the future. A certain allowed deformation is not only more realistic, the additional degrees of freedom may also help to alleviate the ambiguity problems of contact forces so far present in a rigid formulation. This difficult modeling challenge entails many new visionary research areas in the field of rigid body and continuum dynamics such as multi-impact modeling, material modeling and the study of weak variational formulations for the integration of the equations of motion of a mechanical system with unilateral constraints.

Appendix A

Proofs

A.1 Prox Properties

Proof: *(Lemma 6.1)*

Statement (a) is true if $\forall \mathbf{y} \in \mathcal{C}$ there exists at least one $\mathbf{x}$ subject to $\text{prox}_{\mathcal{C}}^{R}(\mathbf{x}) = \mathbf{y}$. Since $\mathcal{C} \subseteq V$, any $\mathbf{y} \in \mathcal{C}$ can be attained by taking $\mathbf{x} = \mathbf{y}$ since $\text{prox}_{\mathcal{C}}^{R}$ does not project in this case.

Statement (b) can be shown with (6.24). The equality $\mathbf{z} = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x})$ is equivalent to the normal cone inclusion

$$\begin{aligned}
\mathbf{R}(\mathbf{x}-\mathbf{z}) \in \mathcal{N}_{\mathcal{C}}(\mathbf{z}) &\Leftrightarrow \langle \mathbf{R}(\mathbf{x}-\mathbf{z}) \,|\, \mathbf{x}^*-\mathbf{z}\rangle \leqslant 0, && \mathbf{z} \in \mathcal{C}, \quad \forall \mathbf{x}^* \in \mathcal{C}, \\
&\Leftrightarrow (\mathbf{x}-\mathbf{z} \,|\, \mathbf{x}^*-\mathbf{z})_R \leqslant 0, && \mathbf{z} \in \mathcal{C}, \quad \forall \mathbf{x}^* \in \mathcal{C} \\
&\Leftrightarrow (\mathbf{z}-\mathbf{x} \,|\, \mathbf{z}-\mathbf{x}^*)_R \leqslant 0, && \mathbf{z} \in \mathcal{C}, \quad \forall \mathbf{x}^* \in \mathcal{C} \\
&\Leftrightarrow (\text{prox}_{\mathcal{C}}^{R}(\mathbf{x})-\mathbf{x} \,|\, \text{prox}_{\mathcal{C}}^{R}(\mathbf{x})-\mathbf{x}^*)_R \leqslant 0, && \forall \mathbf{x}^* \in \mathcal{C}, \quad \forall \mathbf{x} \in V.
\end{aligned}$$

Let $\mathbf{z} = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}) \Leftrightarrow \mathbf{R}(\mathbf{x}-\mathbf{z}) \in \mathcal{N}_{\mathcal{C}}(\mathbf{z})$. The normal cone $\mathcal{N}_{\mathcal{C}}(\mathbf{z})$ is equal to the subdifferential of the indicator function $\partial I_{\mathcal{C}}(\mathbf{z})$ which is maximal monotone. By exploiting this property and taking two points $\mathbf{z}_1 = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_1)$ and $\mathbf{z}_2 = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_2)$, it follows that

$$\begin{aligned}
& \langle \mathbf{R}(\mathbf{x}_1-\mathbf{z}_1) - \mathbf{R}(\mathbf{x}_2-\mathbf{z}_2) \,|\, \mathbf{z}_1-\mathbf{z}_2\rangle \geqslant 0 \\
\Leftrightarrow \quad & \langle \mathbf{R}(\mathbf{x}_1-\mathbf{x}_2) \,|\, \mathbf{z}_1-\mathbf{z}_2\rangle \geqslant \langle \mathbf{R}(\mathbf{z}_1-\mathbf{z}_2) \,|\, \mathbf{z}_1-\mathbf{z}_2\rangle \\
\Leftrightarrow \quad & (\mathbf{x}_1-\mathbf{x}_2 \,|\, \mathbf{z}_1-\mathbf{z}_2)_R \geqslant (\mathbf{z}_1-\mathbf{z}_2 \,|\, \mathbf{z}_1-\mathbf{z}_2)_R = \|\mathbf{z}_1-\mathbf{z}_2\|_R \geqslant 0 \\
\Leftrightarrow \quad & (\text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_1) - \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_2) \,|\, \mathbf{x}_1-\mathbf{x}_2)_R \geqslant 0.
\end{aligned}$$

This inner product is only zero for $\text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_1) = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_2)$ or $\mathbf{x}_1 = \mathbf{x}_2 \Rightarrow \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_1) = \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_2)$ and only positive if $\text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_1) \neq \text{prox}_{\mathcal{C}}^{R}(\mathbf{x}_2)$. This proofs statement (c).

Statement (d) is obtained by expanding the following norm

$$\begin{aligned}
\|(\mathbf{x}_1 - \mathbf{x}_2) - (\mathbf{z}_1 - \mathbf{z}_2)\|_R^2 \geqslant 0 \quad &\Leftrightarrow \\
\|\mathbf{x}_1 - \mathbf{x}_2\|_R^2 \geqslant -\|\mathbf{z}_1 - \mathbf{z}_2\|_R + 2\underbrace{(\mathbf{x}_1 - \mathbf{x}_2 \,|\, \mathbf{z}_1 - \mathbf{z}_2)_R}_{\geqslant \|\mathbf{z}_1 - \mathbf{z}_2\|_R \quad \text{with (c)}} \quad &\Leftrightarrow \\
\|\mathbf{x}_1 - \mathbf{x}_2\|_R^2 \geqslant \|\mathbf{z}_1 - \mathbf{z}_2\|_R = \left\|\mathrm{prox}_{\mathcal{C}}^R(\mathbf{x}_1) - \mathrm{prox}_{\mathcal{C}}^R(\mathbf{x}_2)\right\|_R &
\end{aligned}$$

The inequality of the right term in the second line occurs in proof (c). ∎

A.2 Frictional Contact Formulations

Proof: *(Equivalence of Cone Complementarity* (7.38) *and De Saxé's Formulation* (7.33)*)*
UNCF ⇒ CCF: Show that the cone complementarity problem can be extracted from the unified normal cone formulation. As a good practice, we give a self-contained proof which is analogue to lemma 4.2 in [61]. Recall the definition 6.10 of the normal cone at a point $\boldsymbol{\lambda} \in \mathcal{K}_\mu$ as

$$-\tilde{\boldsymbol{\gamma}} \in \mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda}) \coloneqq \left\{\mathbf{y} \in \mathbb{E}^{3**} \mid \langle \mathbf{y} \,|\, \boldsymbol{\lambda}^* - \boldsymbol{\lambda}\rangle \leqslant 0,\ \forall \boldsymbol{\lambda}^* \in \mathcal{K}_\mu\right\} \subset \mathbb{E}^{3**} \tag{A.1}$$

$$\cong \left\{\mathbf{y} \in \mathbb{E}^{3} \mid \langle \boldsymbol{\lambda}^* - \boldsymbol{\lambda} \,|\, \mathbf{y}\rangle \leqslant 0,\ \forall \boldsymbol{\lambda}^* \in \mathcal{K}_\mu\right\} \subset \mathbb{E}^{3}, \tag{A.2}$$

and it follows that for a particular $\boldsymbol{\lambda} \in \mathcal{K}_\mu$

$$\langle \boldsymbol{\lambda}^* - \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}}\rangle \geqslant 0 \ \forall \boldsymbol{\lambda}^* \in \mathcal{K}_\mu. \tag{A.3}$$

If we set $\boldsymbol{\lambda}^* = \mathbf{0} \in \mathcal{K}_\mu$, we get $\langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}}\rangle \leqslant 0$. For every $\boldsymbol{\lambda} \in \mathcal{K}_\mu$ we can choose $\boldsymbol{\lambda}^* = 2\boldsymbol{\lambda} \in \mathcal{K}_\mu$ which results in $\langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}}\rangle \geqslant 0$. These two inequalities are the same as stating $\langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}}\rangle = 0$.

It is left to show that $-\tilde{\boldsymbol{\gamma}} \in \mathcal{K}_\mu^\circ$. This is true if $\mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda})$ at $\boldsymbol{\lambda} \in \mathcal{K}_\mu$ is a subset of $\mathcal{K}_\mu^\circ$, that is, $\mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda}) \subseteq \mathcal{K}_\mu^\circ$. We reformulate the normal cone inclusion (A.2) as

$$\mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda}) \coloneqq \left\{\mathbf{y} \in \mathbb{E}^3 \mid \langle \mathbf{z}^* \,|\, \mathbf{y}\rangle \leqslant 0 \quad \forall \mathbf{z}^* = \mathcal{K}_\mu - \boldsymbol{\lambda}\right\}. \tag{A.4}$$

Because of the convexiy of $\mathcal{K}_\mu$, every $\mathbf{x} \in \mathcal{K}_\mu$ is also in the set $\mathcal{K}_\mu - \boldsymbol{\lambda}$ and it follows that $\mathcal{K}_\mu \subseteq \mathcal{K}_\mu - \boldsymbol{\lambda}$. Equation (A.4) is the definition of the polar cone to $\mathcal{K}_\mu$ but with a more restrictive $\forall$-quantifier. It follows that $\mathcal{K}_\mu^\circ$ is a superset of $\mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda})$, because it is less restrictive on $\mathbf{y}$ which finalizes this proof. Furthermore, $\mathcal{K}_\mu^\circ = \mathcal{N}_{\mathcal{K}_\mu}(\mathbf{0})$.

UNCF ⇐ CCF: To show the contrary, start with the cone complementarity and take two vectors

$$-\tilde{\boldsymbol{\gamma}} \in \mathcal{K}_\mu^\circ \cong \left\{\mathbf{y} \in \mathbb{E}^3 \mid \langle \boldsymbol{\lambda}^* \,|\, \mathbf{y}\rangle \leqslant 0, \quad \forall \boldsymbol{\lambda}^* \in \mathcal{K}_\mu\right\} \tag{A.5}$$

and $\boldsymbol{\lambda} \in \mathcal{K}_\mu$ and write $\langle \boldsymbol{\lambda} \,|\, \tilde{\boldsymbol{\gamma}} \rangle = 0$ as two inequalities $\langle \boldsymbol{\lambda} \,|\, -\tilde{\boldsymbol{\gamma}} \rangle \leqslant 0 \wedge \langle \boldsymbol{\lambda} \,|\, -\tilde{\boldsymbol{\gamma}} \rangle \geqslant 0$ and combine them with the polar cone which results in

$$\langle \boldsymbol{\lambda}^* \,|\, -\tilde{\boldsymbol{\gamma}} \rangle \leqslant 0 \leqslant \langle \boldsymbol{\lambda} \,|\, -\tilde{\boldsymbol{\gamma}} \rangle \quad \Leftrightarrow \quad \langle \boldsymbol{\lambda}^* - \boldsymbol{\lambda} \,|\, -\tilde{\boldsymbol{\gamma}} \rangle \leqslant 0, \quad \forall \boldsymbol{\lambda}^* \in \mathcal{K}_\mu. \tag{A.6}$$

From (A.6) follows the definition $-\tilde{\boldsymbol{\gamma}} \in \mathcal{N}_{\mathcal{K}_\mu}(\boldsymbol{\lambda})$. ■

Appendix B

Principal Component Analysis

The principal component analysis is a statistical method to obtain orthogonal principal directions $\{\mathbf{d}_1, \dots, \mathbf{d}_n\} \in \mathbb{R}^n$ from a set of observations of possibly correlated variables in $\mathbb{R}^n$. This procedure has many names depending on the field of application. It is also named proper orthogonal decomposition in mechanical engineering or singular value decomposition of $\mathbf{A}^\top\mathbf{A}$ of a matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$. The principal directions form an orthogonal basis $\{\mathbf{d}_i\}$ such that the first direction $\mathbf{d}_1$ is associated with the greatest variance of the observations projected onto $\mathbf{d}_1$ and the last direction $\mathbf{d}_n$ is associated with the smallest variance. In this chapter, the principal component analysis is explained in detail with a focus on mechanics, and links to quantities used in the field of statistics are given at the end.

The observation data of interest in this work is a point cloud of n points S_i with displacement vectors $\mathbf{r}_{OS_i}$ in the Eucledian vector space $\mathbb{E}^3$. The origin is denoted by O. The weighted mean point (center of mass) G is given as $\mathbf{r}_{OG} \coloneqq \frac{1}{m_T}\sum_{i=1}^n m_i\mathbf{r}_{OS_i}$ where the weights (masses) m_i are positive and the total mass is given as $m_T \coloneqq \sum_{i=1}^n m_i$. The point cloud is described, in the following relative, to its mean point, that is, $\mathbf{p}_i = \mathbf{r}_{GS_i} = \mathbf{r}_{OS_i} - \mathbf{r}_{OG}$. The whole derivation in the following is independent of the choice of a coordinate system. A principal direction $\mathbf{d}^*$ can be found by minimizing the projection residuals over all possible directions $\mathbf{d}$. Given a direction $\mathbf{d}$, the projection residual is the total sum of the normed differences between the projection $\hat{\mathbf{p}}_i$ of $\mathbf{p}_i$ onto $\mathbf{d}$ and $\mathbf{p}_i$ over all points with $i \in [1\,;\,n]$. Using the induced M_i-norm $\|\cdot\|_{M_i}$ of the inner product $(\mathbf{x}\,|\,\mathbf{y})_{M_i} = \mathbf{x}^\top\mathbf{M}_i\mathbf{y}$, the minimization problem can be stated as

$$\min_{\mathbf{d}} f(\mathbf{d}) = \min_{\mathbf{d}} \sum_{i=1}^{n} \frac{1}{2}\left\|\mathbf{p}_i - \hat{\mathbf{p}}_i(\mathbf{d})\right\|_{M_i}^2 \ , \tag{B.1}$$

$$\textit{subj. to: } g(\mathbf{d}) \coloneqq \|\mathbf{d}\|_2^2 - 1 = 0 \ . \tag{B.2}$$

The positive definite metric $\mathbf{M}_i$ is chosen proportional to the identity matrix as $m_i\mathbf{I}_3$, such that m_i is the weighting factor for the projection residual of each point. The constraint $g(\mathbf{d}) = 0$ ensures that $\mathbf{d}$ is of unit length, which is crucial because otherwise $\mathbf{d} = \mathbf{0} \Rightarrow \hat{\mathbf{p}}_i(\mathbf{d}) = \mathbf{0}\ \forall i \in [1\,;\,n]$ would minimize the above problem and is treated as an infeasible solution. An example in $\mathbb{R}^2$ is shown in figure B.1.

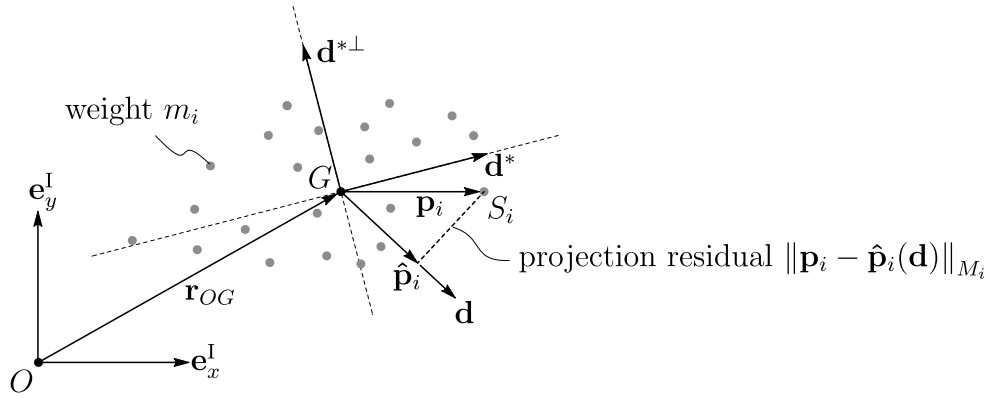


Figure B.1: A gray point cloud in $\mathbb{R}^2$. The optimal principal direction which minimizes (B.1) is denoted by $\mathbf{d}^*$. The direction with the greatest and smallest variance is denoted by $\mathbf{d}^*$ and $\mathbf{d}^{*\top}$, respectively.

The projection $\hat{\mathbf{p}}_i$ of a point i can be expressed as

$$\hat{\mathbf{p}}_i = \alpha_i \mathbf{d} \ , \quad \forall i \in [1\,;\, n] \ , \tag{B.3}$$

where the scalar α_i is a yet unknown projection length for every point i. As seen in the following, the principal direction $\mathbf{d}$ will depend on the choice of the reference point G. Inserting the above into the minimization problem B.1 and adding the indicator function $I_{\{0\}}(g(\mathbf{d}))$ to the convex set $\{0\}$ results in a unconstrained minimization problem in the variables $\mathbf{d}$ and $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_n]^\top$ as

$$\begin{aligned}
\underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmin}}\, h(\mathbf{d}, \boldsymbol{\alpha}) &= \underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmin}}\, f(\mathbf{d}, \boldsymbol{\alpha}) + I_{\{0\}}(g(\mathbf{d})) && \text{(B.4)} \\
&= \underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmin}}\, \frac{1}{2} \sum_{i=1}^{n} (\mathbf{p}_i - \alpha_i \mathbf{d})^\top m_i (\mathbf{p}_i - \alpha_i \mathbf{d}) + I_{\{0\}}(g(\mathbf{d})) && \text{(B.5)} \\
&= \underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmin}}\, \sum_{i=1}^{n} -\alpha_i m_i \mathbf{d}^\top \mathbf{p}_i + \frac{1}{2} \alpha_i^2 m_i \mathbf{d}^\top \mathbf{d} + I_{\{0\}}(g(\mathbf{d})) \ . && \text{(B.6)}
\end{aligned}$$

A necessary condition for a local optimum is that the zero vector $\mathbf{0}$ is contained in the subdifferential of the objective function, that is, $\mathbf{0} \in \partial h(\mathbf{d}, \boldsymbol{\alpha})$, see theorem 6.1. Note that $h(\mathbf{d}, \boldsymbol{\alpha})$ does not need to be convex for the application of the subdifferential. Under the restriction that some function $f(\mathbf{d})$ is monotone, the composition $I_{\{0\}}(f(\mathbf{d}))$ would still be convex. Unfortunately, the above constraint $g : \mathbb{R}^n \to \mathbb{R}$ is an increasing function in all directions and is not monotone. One could possible say, that g is at least locally monotone, and thus the composition $I_{\{0\}}(f(\mathbf{d}))$ might be locally convex. Evaluating the subdifferential with respect to $\mathbf{d}$ and $\boldsymbol{\alpha}$ yields the following equations for the local optimum $\mathbf{d}^*, \mathbf{a}^*$:

$$\begin{aligned}
0 \overset{!}{\in} \partial_{\alpha_i} h(\mathbf{d}^*, \boldsymbol{\alpha}^*) &= \frac{\partial f}{\partial \alpha_i}(\mathbf{d}^*, \boldsymbol{\alpha}^*) && \text{(B.7)} \\
\Rightarrow 0 &= -\mathbf{d}^{*\top} \mathbf{p}_i{}^\top + \alpha_i^* \mathbf{d}^{*\top} \mathbf{d}^{*\top} \quad \forall i \in [1\,;\, n] \ , && \text{(B.8)}
\end{aligned}$$

$$\mathbf{0} \overset{!}{\in} \partial_{\mathbf{d}} h(\mathbf{d}^*, \boldsymbol{\alpha}^*) = \frac{\partial f}{\partial \mathbf{d}}(\mathbf{d}^*, \boldsymbol{\alpha}^*) + \partial_{\mathbf{d}} I_{\{0\}}(g(\mathbf{d})) \tag{B.9}$$

$$= \sum_{i=1}^{n} -\alpha_i^* m_i \mathbf{p}_i + \alpha_i^{*2} m_i \mathbf{d}^* + \partial_g I_{\{0\}}(g) \left.\frac{\partial g}{\partial \mathbf{d}}^\top\right|_{\mathbf{d}^*} \tag{B.10}$$

$$\Rightarrow \mathbf{0} = \sum_{i=1}^{n} -\alpha_i^* m_i \mathbf{p}_i + (\alpha_i^{*2} m_i + \lambda)\mathbf{d}^*, \tag{B.11}$$

where $\lambda \in \mathbb{R}^* = 2\ \partial_g I_{\{0\}}(g)$ is the set-valued placeholder or lagrange multiplier which will be chosen such that the constraint $g(\mathbf{d}^*) = 0$ is enforced. It is essential to note that (B.8) and (B.11) and $g(\mathbf{d}^*) = 0$ is the complete set of conditions to enforce for finding the optimum value $\mathbf{d}^*, \boldsymbol{\alpha}^*$. Solving the first equation (B.8) results in

$$\alpha_i^* = \frac{\mathbf{d}^{*\top} \mathbf{p}_i}{\mathbf{d}^{*\top} \mathbf{d}^*}, \quad \forall i \in [1\,;\, n] \tag{B.12}$$

and substituting into (B.5) results in the simplified unconstrained optimization problem

$$\min_{\mathbf{d}, \boldsymbol{\alpha}} \frac{1}{2} \sum_{i=1}^{n} m_i \mathbf{p}_i{}^\top (\mathbf{I} - \mathbf{d}\mathbf{d}^\top)^2 \mathbf{p}_i + I_{\{0\}}(g(\mathbf{d})), \tag{B.13}$$

where $(\mathbf{I} - \mathbf{d}\mathbf{d}^\top)^2 = (\mathbf{I} - \mathbf{d}\mathbf{d}^\top)$ is the imdepotent projection operator onto the hyperplane with unit normal direction $\mathbf{d}$. By substituting (B.12) into (B.11) one gets a condition for the optimal direction $\mathbf{d}^*$, that is,

$$\sum_{i=1}^{n} m_i \mathbf{p}_i \mathbf{p}_i{}^\top \mathbf{d}^* = \underbrace{\mathbf{d}^{*\top} \mathbf{d}^*}_{=1} (\sum_{i=1}^{n} \alpha_i^{*2} m_i + \lambda)\mathbf{d}^*. \tag{B.14}$$

One may notice that the above equation is an Eigenvalue problem and that $\mathbf{d}^*$ is an Eigenvector of the matrix

$$\mathbf{E}_G = \sum_{i=1}^{n} m_i \mathbf{p}_i \mathbf{p}_i{}^\top = \sum_{i=1}^{n} m_i (\mathbf{r}_{OS_i} - \mathbf{r}_{OG})(\mathbf{r}_{OS_i} - \mathbf{r}_{OG})^\top = \mathbf{P}^\top \mathbf{P}\ , \quad \text{with:} \quad \mathbf{P}^\top \coloneqq [\sqrt{m_1}\mathbf{p}_1, \ldots, \sqrt{m_n}\mathbf{p}_n] \tag{B.15}$$

and that the term $s = \sum_{i=1}^{n} \alpha_i^{*2} m_i + \lambda$ is the corresponding Eigenvalue. Note, that λ is still free and can be determined when s and $\mathbf{d}^*$ are known. In mechanical terms, matrix $\mathbf{E}_G$ is the Binet inertia tensor at point G of all mass points with masses m_i. In a statistical setting, the matrix $\frac{1}{m_T}\mathbf{E}_G$ corresponds to the weighted covariance matrix of the observations $\mathbf{r}_{OS_i}$ with weighted sample mean $\mathbf{r}_{OG}$ and weights m_i. A smaller weight might account for outlier points which should not contribute as much as regular points. It is still left to show that the eigenvector to the largest eigenvalue minimizes $h(\mathbf{d}^*, \boldsymbol{\alpha}^*)$. This can be seen if we reformulate (B.6) by subtituting (B.12) and using $g(\mathbf{d}) = 0$, which

yields

$$\underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{i=1}^{n} -(\mathbf{d}^\top \mathbf{p}_i) m_i \mathbf{d}^\top \mathbf{p}_i + \frac{1}{2}(\mathbf{d}^\top \mathbf{p}_i)^2 m_i \underbrace{\mathbf{d}^\top \mathbf{d}}_{=1} + I_{\{0\}}(g(\mathbf{d})) \tag{B.16}$$

$$= \underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmax}} \sum_{i=1}^{n} \frac{1}{2} m_i \mathbf{d}^\top \mathbf{p}_i {\mathbf{p}_i}^\top \mathbf{d} - I_{\{0\}}(g(\mathbf{d})) \tag{B.17}$$

$$= \underset{\mathbf{d},\boldsymbol{\alpha}}{\operatorname{argmax}} \sum_{i=1}^{n} \frac{1}{2} \mathbf{d}^\top \mathbf{E}_G \mathbf{d} - I_{\{0\}}(g(\mathbf{d})) \ . \tag{B.18}$$

If $\mathbf{d}$ is a unit Eigenvector of $\mathbf{E}_G$ with Eigenvalue s, then (B.18) simplifies to the maximization over the Eigenvalue s, since $\mathbf{d}^\top \mathbf{E}_G \mathbf{d} = s\mathbf{d}^\top \mathbf{d} = s$ by (B.14). The Binet tensor $\mathbf{E}_G$ is symmetric and real and thus diagonalizable and an orthogonal normalized basis can be constructed. It can be shown that the Eigenvector to the smallest Eigenvalue corresponds to the principal direction with a maximal projection residual. Instead of using an Eigenvalue decomposition, one can use the singular value decomposition of $\mathbf{P} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ and extract the Eigenvectors from $\mathbf{V}^\top$.

The principal component analysis has been used in this thesis for the construction of a fitted bounding box in section 11.1.

Appendix C

Integrator with Displacement Jumps

The explicit integrator discussed in this section is an adaptation obtained from communications with Ondrej Papes and presented here as a rather experimental motivation to obtain a drift correction acting directly on displacement level, instead of the numerical trickery presented in section 8.2.

The equations of motion and the kinematic equation in (8.1) are written as two measure differential equations in the form:

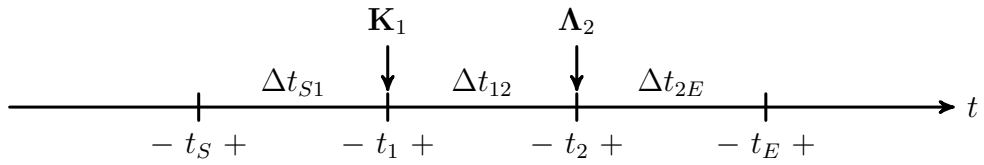
$$\mathrm{d}\mathbf{u} = \mathbf{M}(\mathbf{q},t)^{-1}\left(\mathbf{h}(\mathbf{q},\mathbf{u},t)\mathrm{d}t - \mathbf{W}(\mathbf{q},t)\mathrm{d}\mathbf{P}\right) \tag{C.1}$$

$$\mathrm{d}\mathbf{q} = \mathbf{F}(\mathbf{q},t)\mathbf{u}\mathrm{d}t \ + \mathbf{b}(\mathbf{q},t)\mathrm{d}t + \mathbf{Q}(\mathbf{q},t)\mathrm{d}\boldsymbol{\Sigma}. \tag{C.2}$$

These measure differential equations can describe smooth continuous time intervals as well as intervals with discontinuities in the displacement $\mathbf{q}$ and velocity $\mathbf{u}$. Similar to the impulse measure $\mathrm{d}\mathbf{P}$, a measure $\mathrm{d}\boldsymbol{\Sigma}$ is introduced in the kinematic equation (C.2) which allows the displacement $\mathbf{q}$ to jump at certain time instants. All measures above contain a Lebesgue measure $\mathrm{d}t$ and an atomic measure $\mathrm{d}\eta$ and are given as

$$\begin{aligned}
\mathrm{d}\mathbf{q} &\coloneqq \dot{\mathbf{q}}\mathrm{d}t + (\mathbf{q}^+ - \mathbf{q}^-)\mathrm{d}\eta && \text{(displacement measure)}\\
\mathrm{d}\mathbf{u} &\coloneqq \dot{\mathbf{u}}\mathrm{d}t + (\mathbf{u}^+ - \mathbf{u}^-)\mathrm{d}\eta && \text{(velocity measure)}\\
\mathrm{d}\mathbf{P} &\coloneqq \boldsymbol{\lambda}\mathrm{d}t + \boldsymbol{\Lambda}\mathrm{d}\eta, \quad \boldsymbol{\Lambda} \coloneqq \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^- && \text{(impulse measure)}\\
\mathrm{d}\boldsymbol{\Sigma} &\coloneqq \mathbf{k}\mathrm{d}t + \mathbf{K}\mathrm{d}\eta, \quad \mathbf{K} \coloneqq \mathbf{K}^+ - \mathbf{K}^-. && \text{(impulse-impulse measure)}
\end{aligned} \tag{C.3}$$

The term $\mathbf{Q}(\mathbf{q},t)$ corresponds to some yet unknown generalized force direction for the impulse-impulse quantity $\mathrm{d}\boldsymbol{\Sigma}$. In the following, we are interested in finding an approximate solution of (C.1) and (C.2) in a time interval $[t_S, t_E]$ with $\Delta t \coloneqq t^E - t^S$. This interval is split into three distinct intervals such that $\Delta t = \Delta t_{S1} + \Delta t_{12} + \Delta t_{2E}$ with intermediate times t_1 and t_2 and $t_2 > t_1$. It is assumed that a displacement jump may occurs at time t_1 and is solely caused by the impulse-impulse $\mathbf{K}_1$. Similarly, a velocity jump occurs at time t_2 and is exclusively caused by the impulse $\boldsymbol{\Lambda}_2$. We assume that the force and impulse density $\boldsymbol{\lambda}(\cdot)$ and $\mathbf{k}(\cdot)$, respectively, are both zero in the time interval Δt. Furthermore, it is assumed that all quantities $\mathbf{M}, \mathbf{F}, \mathbf{b}, \mathbf{W}, \mathbf{Q}$ are constant within the time interval Δt. The time line is given as



The integration of (C.1) and (C.2) over the interval (t_S, t_E) can be split into the following integrals:

$$\int\limits_{(t_S,t_E)} \mathrm{d}\mathbf{q} = \int\limits_{(t_S,t_1)} \mathrm{d}\mathbf{q} + \underbrace{\int\limits_{\{t_1\}} \mathrm{d}\mathbf{q}}_{\text{displacement jump}} + \int\limits_{(t_1,t_2)} \mathrm{d}\mathbf{q} + \int\limits_{\{t_2\}} \mathrm{d}\mathbf{q} + \int\limits_{(t_2,t_E)} \mathrm{d}\mathbf{q}\ , \tag{C.4}$$

$$\int\limits_{(t_S,t_E)} \mathrm{d}\mathbf{u} = \int\limits_{(t_S,t_1)} \mathrm{d}\mathbf{u} + \int\limits_{\{t_1\}} \mathrm{d}\mathbf{u} + \int\limits_{(t_1,t_2)} \mathrm{d}\mathbf{u} + \underbrace{\int\limits_{\{t_2\}} \mathrm{d}\mathbf{u}}_{\text{velocity jump}} + \int\limits_{(t_2,t_E)} \mathrm{d}\mathbf{u}\ . \tag{C.5}$$

The time integration over a time interval without impulsive forces can be evaluated as

$$\int\limits_{(t_i,t)} \mathrm{d}\mathbf{u} = \mathbf{u}(t) - \mathbf{u}_i = \mathbf{M}^{-1}\mathbf{h}(t - t_i), \tag{C.6}$$

$$\int\limits_{(t_i,t)} \mathrm{d}\mathbf{q} = \mathbf{q}(t) - \mathbf{q}_i = \int\limits_{(t_i,t)} \mathbf{F}(\mathbf{M}^{-1}\mathbf{h}(t - t_i) + \mathbf{u}_i) + \mathbf{b}\ \mathrm{d}t \tag{C.7}$$

$$= \mathbf{F}\,\mathbf{u}_i(t - t_i) + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{(t - t_i)^2}{2} + \mathbf{b}\ (t - t_i). \tag{C.8}$$

The integrated intervals evaluate to the following 10 equations:

$$(t_S, t_1): \quad \begin{bmatrix} \mathbf{u}_1^- = \mathbf{u}_S^+ + \mathbf{M}^{-1}\mathbf{h}\Delta t_{S1} \\ \mathbf{q}_1^- = \mathbf{q}_S^+ + \mathbf{F}\,\mathbf{u}_S^+\Delta t_{S1} + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{{\Delta t_{S1}}^2}{2} + \mathbf{b}\Delta t_{S1} \end{bmatrix} \tag{C.9}$$

$$\text{displacement jump at } \{t_1\}: \quad \begin{bmatrix} \mathbf{u}_1^+ = \mathbf{u}_1^- \\ \mathbf{q}_1^+ = \mathbf{q}_1^- + \mathbf{Q}\mathbf{K}_1 \end{bmatrix} \tag{C.10}$$

$$(t_1, t_2): \quad \begin{bmatrix} \mathbf{u}_2^- = \mathbf{u}_1^+ + \mathbf{M}^{-1}\mathbf{h}\Delta t_{12} \\ \mathbf{q}_2^- = \mathbf{q}_1^+ + \mathbf{F}\,\mathbf{u}_1^+\Delta t_{12} + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{{\Delta t_{12}}^2}{2} + \mathbf{b}\ \Delta t_{12} \end{bmatrix} \tag{C.11}$$

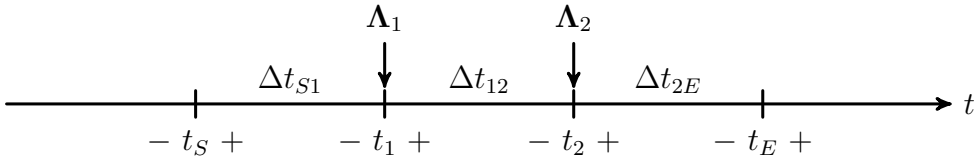
$$\text{velocity jump at } \{t_2\}: \quad \begin{bmatrix} \mathbf{u}_2^+ = \mathbf{u}_2^- + \mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}_2 \\ \mathbf{q}_2^+ = \mathbf{q}_2^- \end{bmatrix} \tag{C.12}$$

$$(t_2, t_E): \quad \begin{bmatrix} \mathbf{u}_E^- = \mathbf{u}_2^+ + \mathbf{M}^{-1}\mathbf{h}\Delta t_{2E} \\ \mathbf{q}_E^- = \mathbf{q}_2^+ + \mathbf{F}\,\mathbf{u}_2^+\Delta t_{2E} + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{{\Delta t_{2E}}^2}{2} + \mathbf{b}\ \Delta t_{2E}\ . \end{bmatrix} \tag{C.13}$$

Given $\mathbf{q}_S^+$ and $\mathbf{u}_S^+$, the equation can be solved for the remaining 10 variables which results in the update equations

$$\begin{aligned}\mathbf{q}_E^- &= \mathbf{q}_S^+ + (\mathbf{F}\ \mathbf{u}_S^+ + \mathbf{b})\Delta t_{SE} + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{\Delta t_{SE}{}^2}{2} + \mathbf{Q}\mathbf{K}_1 + \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}_2\Delta t_{2E} \\ \mathbf{u}_E^- &= \mathbf{u}_S^+ + \mathbf{M}^{-1}\mathbf{h}\Delta t_{SE} + \mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}_2.\end{aligned} \tag{C.14}$$

The integration of the time line, where an impulse $\mathbf{\Lambda}_1$ is acting at time t_1 instead of an impulse-impulse $\mathbf{K}_1$, given as



results in the following update equation

$$\begin{aligned}\mathbf{q}_E^- &= \mathbf{q}_S^+ + (\mathbf{F}\ \mathbf{u}_S^+ + \mathbf{b})\Delta t_{SE} + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{\Delta t_{SE}{}^2}{2} + \mathbf{F}\mathbf{M}^{-1}\mathbf{W}(\mathbf{\Lambda}_1\Delta t_{1E} + \mathbf{\Lambda}_2\Delta t_{2E}) \\ \mathbf{u}_E^- &= \mathbf{u}_S^+ + \mathbf{M}^{-1}\mathbf{h}\Delta t_{SE} + \mathbf{M}^{-1}\mathbf{W}(\mathbf{\Lambda}_1 + \mathbf{\Lambda}_2)\ .\end{aligned} \tag{C.15}$$

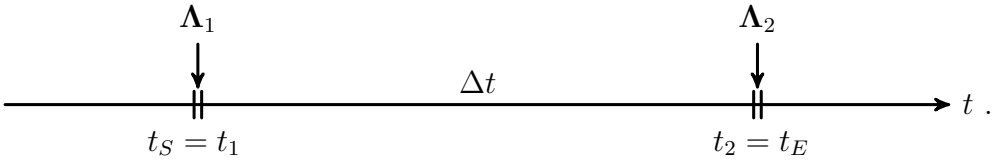
Comparing (C.14) with (C.15), the impulse-impulse term $\mathbf{Q}\mathbf{K}_1$ can be identified as

$$\mathbf{Q}\mathbf{K}_1 = \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}_1\Delta t_{1E}\ , \tag{C.16}$$

from which follows the generalized force direction $\mathbf{Q} = \mathbf{F}\mathbf{M}^{-1}\mathbf{W}$ of the impule-impulse $\mathbf{K}_1$. By the motivation to only have one single dual variable influencing the displacement update in (C.15), we introduce the following transformation $\mathbf{K} \coloneqq \mathbf{\Lambda}_1\Delta t_{1E} + \mathbf{\Lambda}_2\Delta t_{2E}$ and $\mathbf{\Lambda} \coloneqq \mathbf{\Lambda}_1 + \mathbf{\Lambda}_2$ which is written as

$$\begin{bmatrix} \mathbf{K} \\ \mathbf{\Lambda} \end{bmatrix} = \begin{bmatrix} \Delta t_{1E} & \Delta t_{2E} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda}_1 \\ \mathbf{\Lambda}_2 \end{bmatrix}. \tag{C.17}$$

The transformation is singular if both impulses $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ are acting at the same time, that is, $\Delta t_{1E} = \Delta t_{2E}$. This means, that a pair $(\mathbf{K}, \mathbf{\Lambda})$ can not be split anymore into two impulses $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$. Avoiding this singularity as much as possible, we set $\Delta t_{1E} = \Delta t$ and $\Delta t_{2E} = 0$ which results in the time line



The update schemes (C.14) and (C.15), omitting left and right limits, result to

$$\begin{aligned}\mathbf{q}^E &= \mathbf{q}^S + (\mathbf{F}\ \mathbf{u}^S + \mathbf{b})\Delta t + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{\Delta t^2}{2} + \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{K} \\ \mathbf{u}^E &= \mathbf{u}^S + \mathbf{M}^{-1}\mathbf{h}\Delta t + \mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}\ .\end{aligned} \tag{C.18}$$

Algorithm C.1 *(Papes Time-Stepping Scheme for Unilateral Contacts)*:
For a given start time t^S and known displacement $\mathbf{q}^S := \mathbf{q}(t^S)$ and velocity $\mathbf{u}^S := \mathbf{u}(t^S)$ the following 3 steps compute an approximation $\mathbf{q}^E \approx \mathbf{q}(t^E)$ and $\mathbf{u}^E \approx \mathbf{u}(t^E)$ at the end time t^E of the time interval $\Delta t := [t^S, t^E]$.

Step 1 : Compute the approximations

$$\begin{aligned} t^E &= t^S + \Delta t, & t^M &= t^S + \frac{\Delta t}{2} \\ \mathbf{F} &= \mathbf{F}(\mathbf{q}^S, t^S), & \mathbf{b} &= \mathbf{b}(\mathbf{q}^S, t^S) \\ \mathbf{q}^M &= \mathbf{q}^S + (\mathbf{F}\ \mathbf{u}^S + \mathbf{b})\frac{\Delta t}{2} & \mathbf{h} &= \mathbf{h}(\mathbf{q}^M, \mathbf{u}^S, t^M) \\ \mathbf{M} &= \mathbf{M}(\mathbf{q}^M, t^M), & \mathbf{W} &= \mathbf{W}(\mathbf{q}^M, t^M)\ . \end{aligned} \tag{C.19}$$

Step 2 : Solve the impenetrability condition for $\mathbf{q}^E$ and $\mathbf{K}$ for all contacts $i \in [1\,;\,k]$ given as

$$\begin{aligned} \mathbf{K} &= \mathrm{prox}^R_{\mathbb{R}_0^+ \times \cdots \times \mathbb{R}_0^+}\big(\mathbf{K} - \mathbf{R}^{-1}\mathbf{g}_N(\mathbf{q}^E, t^E)\big) \\ \mathbf{q}^E &= \mathbf{q}^{E^*} + \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{K}\ , \end{aligned} \tag{C.20}$$

where $\mathbf{g}_N(\mathbf{q}^E, t^E) := [g_{N,1}(\mathbf{q}^E, t^E), \ldots, g_{N,k}(\mathbf{q}^E, t^E)]^\top$,
$\mathbf{K} := [K_{N,1}, \ldots, K_{N,k}]^\top$ and $\mathbf{q}^{E^*} := \mathbf{q}^S + (\mathbf{F}\ \mathbf{u}^S + \mathbf{b})\Delta t + \mathbf{F}\mathbf{M}^{-1}\mathbf{h}\frac{\Delta t^2}{2}$.

Step 3 : Solve the Newton-type impact law for $\mathbf{u}^E$ and $\mathbf{\Lambda}$ only for the contacts $i \in \mathcal{I}^E := \{i \mid g_{N,i}(\mathbf{q}^E, t^E) \leqslant 0\}$ given as

$$\begin{aligned} \boldsymbol{\gamma}_N^S &= \mathbf{W}\mathbf{u}^S + \boldsymbol{\chi}(\mathbf{q}^M, t^M)\ , \quad \boldsymbol{\gamma}_N^E = \mathbf{W}(\mathbf{q}^E, t^E)\mathbf{u}^E + \boldsymbol{\chi}(\mathbf{q}^E, t^E) \\ \mathbf{\Lambda} &= \mathrm{prox}^R_{\mathbb{R}_0^+ \times \cdots \times \mathbb{R}_0^+}\big(\mathbf{\Lambda} - \mathbf{R}^{-1}(\boldsymbol{\gamma}_N^E + \boldsymbol{\epsilon}\boldsymbol{\gamma}_N^S)\big) \\ \mathbf{u}^E &= \mathbf{u}^S + \mathbf{M}^{-1}\mathbf{h}\Delta t + \mathbf{M}^{-1}\mathbf{W}\mathbf{\Lambda}\ , \end{aligned} \tag{C.21}$$

where $\mathbf{\Lambda} = [\Lambda_{N,1}, \ldots, \Lambda_{N,k}]^\top$ and all other terms $\boldsymbol{\gamma}_N, \mathbf{W}, \boldsymbol{\chi}, \mathbf{R}$ only account for the contacts in the set $\mathcal{I}^E$.

The update scheme (C.18) is convenient as the impulse $\mathbf{\Lambda}$ does not influence the displacement update. The impulse $\mathbf{\Lambda}$ and impulse-impulse $\mathbf{K}$ can be used to formulate two different independent requirements, namely the enforcement of the unilateral contact on displacement level at the end of the time step by $\mathbf{K}$ and the enforcement of the Newton-type impact law of the unilateral contact on velocity level formulate by $\mathbf{\Lambda}$.

If the mechanical system of interest has k possible unilateral contacts, the impenetrability requirement and the impact law are then formulated as

$$-g_{N,i}(\mathbf{q}^E, t^E) \in \mathcal{N}_{\mathbb{R}_0^+}(K_{N,i}) \quad \forall i \in [1\,;\,k] \tag{C.22}$$

$$-(\gamma_{N,i}^E + \varepsilon_{N,i}\ \gamma_{N,i}^S) \in \mathcal{N}_{\mathbb{R}_0^+}(\Lambda_{N,i}) \quad \forall i \in \mathcal{I}^E\ , \tag{C.23}$$

where $\mathcal{I}^E := \{i \mid g_{N,i}(\mathbf{q}(t^E), t^*) \leqslant 0\}$. The reader is referred to (8.1) and (7.75) and (7.27) for the used notation in (C.22). Solving the update equations (C.18) in combination with the two force laws (C.22) and (C.23) suggests the time-stepping scheme in algorithm C.1.

The resolution process for $\mathbf{q}^E$ in (C.20) clearly influences the total energy of the system and influences only the potential energy if the mechanical system is not externally excited and has a constant mass matrix. It is important to note that all possible unilateral contacts need to participate in step 2, and repeated evaluations of $\mathbf{g}_N(\mathbf{q}^E, t^E)$ during the resolution in step 2 can be rather expensive. Step 2 poses one significant problem for large-scale multi-body systems: the number of possible unilateral contacts and the combinations between all possible collision partners quickly becomes intractable. Furthermore, the gap functions are not available analytically. This means that only an active set of closed contacts can be computed for some $\mathbf{q}^E$ in (C.20). Including a changing active set of penetrated contacts during the iterative resolution of (C.20) is not suggested as this leads to a chaotic iteration where contacts are switched on and off continuously. Step 2 can be modified, for example, such that only contacts in an active set $\mathcal{I}^M$ at the mid point t^M are included in the resolution of (C.20). This, however, leads to the same on-off switching behavior over multiple time steps for certain configurations of the mechanical system. Nevertheless, an approximate solution to (C.20) can be viewed as a drift correction on displacement level, and by using a linearized version of $\mathbf{g}_N(\mathbf{q}^E, t^E)$ together with a fixed active set of contacts yields a drift stabilization which does not interfere with the impact law.

The gap function at the end time can be linearized around $\mathbf{q}^{E^*} \coloneqq \mathbf{q}_S + (\mathbf{F}\ \mathbf{u}_S + \mathbf{b})\Delta t + \frac{\Delta t^2}{2}\mathbf{F}\mathbf{M}^{-1}\mathbf{h}$ with the approximations in (C.19). By applying the displacement update in (C.18) and using $t^E - t^{E^*}$ this yields

$$\begin{aligned}
\mathbf{g}_N(\mathbf{q}^E, t^E) &\approx \mathbf{g}_N(\mathbf{q}^{E^*}, t^{E^*}) + \left.\frac{\partial \mathbf{g}_N}{\partial \mathbf{q}}\right|_{\mathbf{q}^{E^*}, t^{E^*}} (\mathbf{q}^E - \mathbf{q}^{E^*}) + \underbrace{\left.\frac{\partial \mathbf{g}_N}{\partial t}\right|_{\mathbf{q}^{E^*}, t^{E^*}} (t^E - t^{E^*})}_{\mathbf{0}} \\
&= \mathbf{g}_N(\mathbf{q}^{E^*}, t^{E^*}) + \left.\frac{\partial g_N}{\partial \mathbf{q}}\right|_{\mathbf{q}^{E^*}, t^{E^*}} \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{K} \\
&\approx \mathbf{g}_N^{E^*} + \mathbf{W}^\top \mathbf{M}^{-1}\mathbf{W}\mathbf{K} = \mathbf{g}_N^{E^*} + \mathbf{G}\mathbf{K}\ . \qquad \text{(C.24)}
\end{aligned}$$

The approximation (C.24) can be substituted into (C.20) to obtain a drift correction step as

$$\begin{aligned}
\mathbf{K} &= \mathrm{prox}^R_{\mathbb{R}_0^+ \times \cdots \times \mathbb{R}_0^+}(\mathbf{K} - \mathbf{R}^{-1}(\mathbf{G}\mathbf{K} + \mathbf{g}_N^{E^*}))\ , \\
\mathbf{q}^E &= \mathbf{q}^{E^*} + \mathbf{F}\mathbf{M}^{-1}\mathbf{W}\mathbf{K}.
\end{aligned} \qquad \text{(C.25)}$$

The above proximal equation can be solved by the following JOR Prox iteration (cf. (8.26) for SOR Prox iteration) till convergence in $\mathbf{K}$ as

$$\boxed{\begin{aligned}
\bar{\mathbf{q}}^{k+1} &= \mathbf{M}^{-1}\mathbf{W}\mathbf{K}^k \\
\mathbf{K}^{k+1} &= \mathrm{prox}^R_{\mathbb{R}_0^+ \times \cdots \times \mathbb{R}_0^+}\left(\mathbf{K}^k - \mathbf{R}^{-1}(\mathbf{W}^\top \bar{\mathbf{q}}^k + \mathbf{g}_N^{E^*})\right) \\
\mathbf{q}^E &\coloneqq \mathbf{q}^{E^*} + \alpha \mathbf{F}\ \bar{\mathbf{q}}^k
\end{aligned}} \qquad \text{(C.26)}$$

with a given initial impulse-impulse $\mathbf{K}^0 \in (\mathbb{R}_0^+)^n$ and initial displacement correction $\bar{\mathbf{q}}^0 \coloneqq \mathbf{M}^{-1}\mathbf{W}\mathbf{K}^0$. The parameter $\alpha \in [0, 1]$ is some drift correction parameter to tune the influence of the correction for an early termination of the iteration.

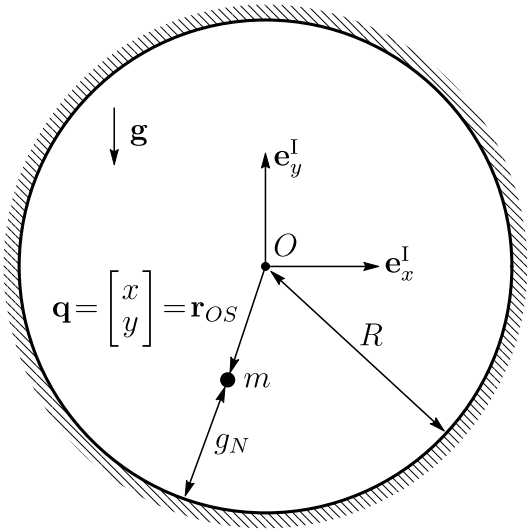
The iteration scheme (C.26) can also be used as a drift correction in Moreau's time-stepping scheme as a fifth step in algorithm (8.1) with

$$\mathbf{q}^{E^*} := \mathbf{q}^S + \Delta t\ \mathbf{F}(\mathbf{q}^M, t^M)\frac{\mathbf{u}^S + \mathbf{u}^E}{2} + \Delta t \mathbf{b}(\mathbf{q}^M, t^M)\ . \tag{C.27}$$

If the evaluation of $\mathbf{g}_N^E$ is not too expensive and is done by a collision detection step which only returns closed contacts, one might try to continuously enlarge the contact set during an iteration of (C.26) or (C.20). Also of interest might be to use a fixed index set of contacts detected in a certain neighborhood around each body.

C.1 Test Example: Mass Point inside Circle

In this section, we demonstrate Papes time-stepping scheme in algorithm C.1 for a simple mechanical system consisting of a mass point under gravity with position vector ${}_I\mathbf{g} = [0, -g]^\top$ inside a circle visualized in figure C.1. The gap function between the mass point and the circle is nonlinear and can be stated analytically. The equation of motions for the generalized coordinate $\mathbf{q} = [x, y]^\top$ and velocity $\mathbf{u} = [\dot{x}, \dot{y}]^\top$ of the system are given in figure C.1. Three evaluations of Papes time-stepping scheme versus Moreau's time-



$$\mathbf{M}\dot{\mathbf{u}} - \mathbf{h} - \mathbf{W}\boldsymbol{\lambda} = \mathbf{0}, \quad \dot{\mathbf{q}} = \mathbf{u}$$

$$\mathbf{M} := \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix}, \quad \mathbf{h} := \begin{bmatrix} 0 \\ -mg \end{bmatrix},$$

$$g_N = R - \|\mathbf{r}_{OS}\|_2 = R - \sqrt{\mathbf{q}^\top \mathbf{q}}$$

$$\boldsymbol{\gamma}_N = \mathbf{W}(\mathbf{q})\mathbf{u}, \quad \mathbf{W}(\mathbf{q}) := \frac{\partial \mathbf{g}_N}{\partial \mathbf{q}} = -\frac{\mathbf{q}}{\sqrt{\mathbf{q}^\top \mathbf{q}}}.$$

Parameters:

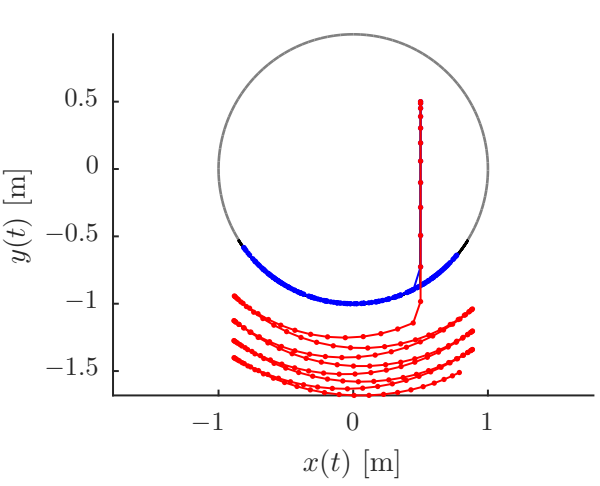
$m = 1\,\mathrm{kg}, \quad R = 1\,\mathrm{m}, \quad g = 9.81\,\mathrm{kg\,m\,s^{-2}}$

Initial Condition:

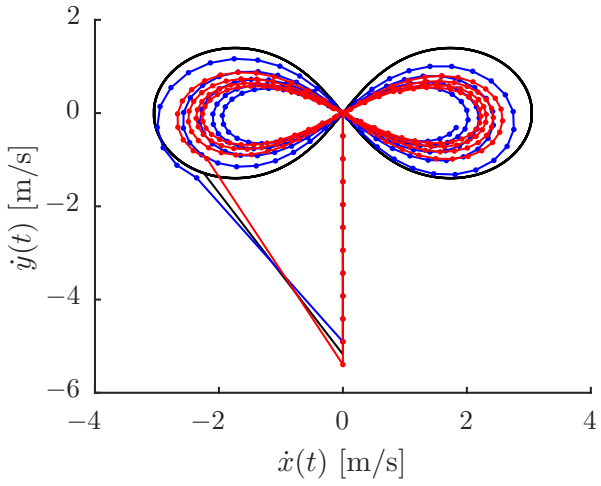
$\mathbf{q} = [0.5, 0.5]^\top, \quad \mathbf{u} = \mathbf{0}$

Figure C.1: Mass point m under gravity inside a circle and the corresponding mathematical description.

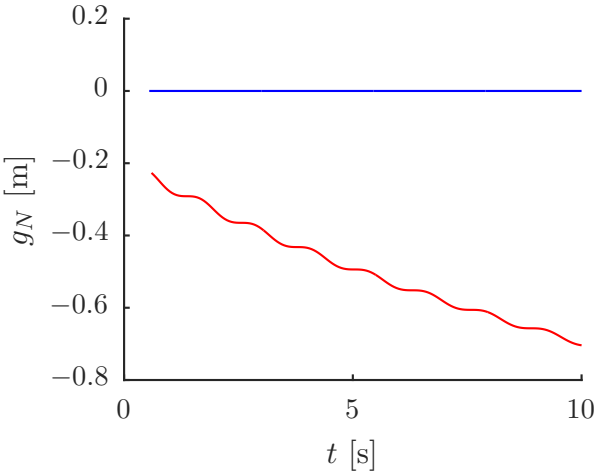
stepping scheme are shown in figure C.2. All simulations use the same $\Delta t = 0.05$ s but once with an inelastic impact with $\epsilon_N = 0$, with an elastic impact $\epsilon_N = \sqrt{0.5}$ and also with a fully elastic impact with $\epsilon_N = 1$. All simulations are compared to a roughly accurate reference simulation with Moreau's time-stepping scheme with $\Delta t = 10^{-4}$. Figure C.2 shows that impenetrability is fulfilled for the whole integration period for Papes scheme whereas the position in Moreau's scheme is prone to large drift for large time steps. Papes time-stepping scheme is not energy consistent and also not for a fully inelastic impact such as Moreau's scheme.



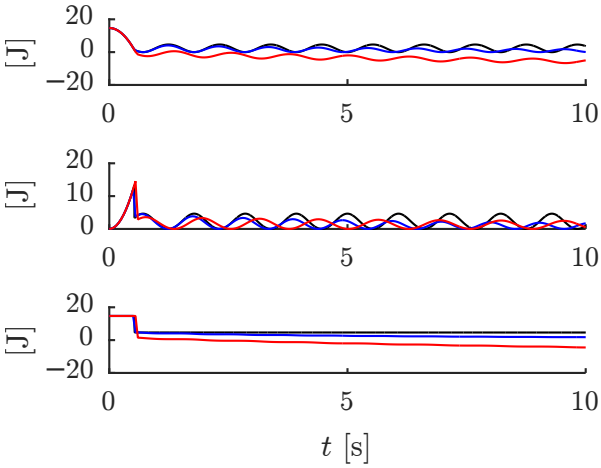
(a) generalized displacement $\mathbf{q}(t)$



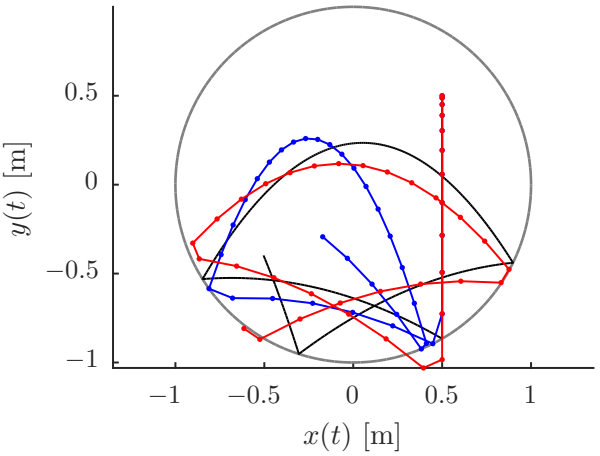
(b) generalized velocity $\mathbf{u}(t)$



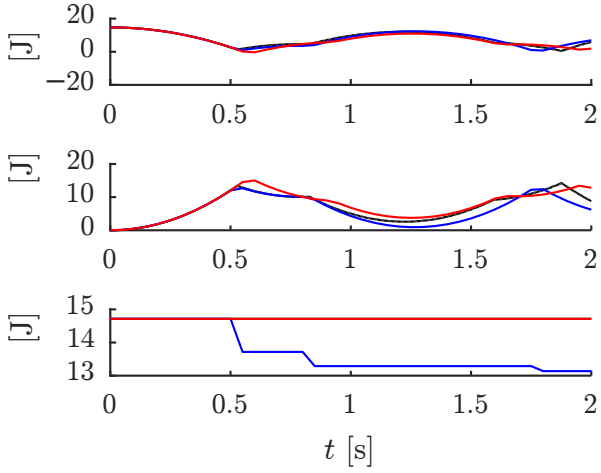
(c) gap function $g_N(t)$



(d) top-bottom: pot., kin. and total energy.



(e) generalized displacement $\mathbf{q}(t)$



(f) top-bottom: pot., kin. and total energy.

Figure C.2: Mass point under gravity inside a circle (gray). Moreau's time-stepping scheme (red) versus Papes time-stepping scheme (blue) compared to an exact reference (black), Settings (a-d): $\Delta t = 0.05\,\mathrm{s}$ and $\epsilon_N = 0$. Settings (e-f): $\Delta t = 0.05\,\mathrm{s}$ and $\epsilon_N = 1$.

Appendix D

Linear Algebra in Mechanics

This chapter should help the reader to familiarize himself with some concepts from linear algebra in the context of mechanics. The lack of notational strictness, also due to commonly regarding this fundamental basics as trivial[1], often causes confusion between concepts such as coordinate tuples, vectors, transformation matrices, rotations and so forth. Proper use of these concepts and its notation is not only important for correct computations but also extremely useful for collaborative software development in mechanics. This chapter makes an attempt to remove these confusions as much as possible. When not stated differently, Einstein summation convention is applied throughout this chapter. The reader should note that the proofs shown in this chapter may not be complete.

D.1 Linear Space

The mathematical concept which is of most interest in mechanics is the notion of a *vector space* or *linear space*.

Definition D.1 *(Linear Space, Vector Space)*:
A *linear space* $(V, \mathbb{F}, \oplus, \odot)$ is a set V (of vectors) and a field $\mathbb{F}$ (of scalars) equipped with two binary operations, $\oplus : V \times V \to V$, called *vector addition*, and $\odot : \mathbb{F} \times V \to V$, called *scalar multiplication*, such that:

- the vector addition satisfies the following properties:

  Associativity: $\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in V, \quad \mathbf{x} \oplus (\mathbf{y} \oplus \mathbf{z}) = (\mathbf{x} \oplus \mathbf{y}) \oplus \mathbf{z}$.
  Commutativity: $\forall \mathbf{x}, \mathbf{y}, \in V, \quad \mathbf{x} \oplus \mathbf{y} = \mathbf{y} \oplus \mathbf{x}$.
  Identity Element: $\mathbf{0} \in V, \ \forall \mathbf{x} \in V, \quad \mathbf{x} \oplus \mathbf{0} = \mathbf{x}$.
  Inverse Element: $\forall \mathbf{x} \in V \ \exists (-\mathbf{x}) \in V, \quad \mathbf{x} \oplus (-\mathbf{x}) = \mathbf{0}$.

- the scalar multiplication satisfies the following properties:

  Associativity: $\forall a, b \in \mathbb{F}, \ \mathbf{z} \in V, \quad a \odot (b \odot \mathbf{z}) = (a \cdot b) \odot \mathbf{z}$.
  Multiplication by Identity of $\mathbb{F}$: $\forall \mathbf{x} \in V, \quad 1 \odot \mathbf{x} = \mathbf{x}$.

[1] In disagreement to the authors opinion.

Distributivity: $\forall a, b \in \mathbb{F},\ \forall \mathbf{x}, \mathbf{y} \in V,\ (a \oplus b) \odot \mathbf{x} = (a \odot \mathbf{x}) \oplus (b \odot \mathbf{x})$ and $a \odot (\mathbf{x} \oplus \mathbf{y}) = (a \odot \mathbf{x}) \oplus (b \odot \mathbf{y})$.

In the following, the two operators $\oplus$ and $\odot$ are omitted for the notation of a linear space, that is, $(V, \mathbb{F})$, and are written with the normal operators used for multiplication and addition of scalars, that is, "+" and "$\cdot$", respectively.

Definition D.2 *(Linear Independence)*:
A set of vectors $S = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \subseteq V$ is called *linearly independent* if and only if

$$\sum_{i=1}^{n} a_i \mathbf{v}_i = \mathbf{0} \quad \Leftrightarrow \quad a_i = 0 \quad \forall i \in [1\,;\, n] \ . \tag{D.1}$$

Definition D.3 *(Basis of a Vector Space)*:
For a linear space $(V, \mathbb{F})$, a set of vectors $\bar{\bar{\mathbf{e}}}^{\mathrm{A}} \coloneqq \{\mathbf{a}_i\} \coloneqq \{\mathbf{a}_1, \ldots, \mathbf{a}_n\} \subseteq V$ is called a *basis* of $(V, \mathbb{F})$ if and only if they are linearly independent and they span the linear space V. The letter A denotes the name of the basis.

Definition D.4 *(Dimension of a Vector Space)*:
If a basis of a linear space $(V, \mathbb{F})$ with a finite number of elements exists, the number of elements of this basis is called the *dimension* of $(V, \mathbb{F})$ and denoted by $\dim_{\mathbb{F}}(V)$ and the linear space $(V, \mathbb{F})$ is called *finite*-dimensional. If not, $(V, \mathbb{F})$ is called *infinite*-dimensional.

If the linear space $(V, \mathbb{F})$ has dimension $\dim_{\mathbb{F}}(V) = n$, then there exist n linearly independent vectors which form a basis. Any set of $n + 1$ or more vectors are linearly dependent.

The reader should note that in this chapter only *finite*-dimensional vector spaces are considered.

D.2 Linear Map

A linear map maps elements from one linear space into another and is defined as the following:

Definition D.5 *(Linear Map)*:
Let $(U, \mathbb{F})$ and $(V, \mathbb{F})$ be two linear spaces. Then, the function or map $\mathcal{A} : U \to V$ is called linear if and only if $\forall \mathbf{u}_1, \mathbf{u}_2 \in U,\ a_1, a_2 \in \mathbb{F}$ and

$$\mathcal{A}(a_1 \mathbf{u}_1 + a_2 \mathbf{u}_2) = a_1 \mathcal{A}(\mathbf{u}_1) + a_2 \mathcal{A}(\mathbf{u}_2) \ .$$

If, for example, $(U, \mathbb{F}) = (\mathbb{R}^n, \mathbb{R})$ and $(V, \mathbb{F}) = (\mathbb{R}^m, \mathbb{R})$ with a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ then the matrix multiplication $\mathbf{u} \mapsto \mathbf{v} = \mathbf{A}\mathbf{u}$ is a linear map.

D.3 Coordinate Map with respect to a Basis

The coordinate map is a linear map which maps elements from a linear space $(V, \mathbb{F})$ to a linear space $(\mathbb{F}^n, \mathbb{F})$. A vector $\mathbf{x} \in V$ can be represented by a linear combination of the

basis vectors $\bar{\bar{\mathbf{e}}}^{\mathrm{A}}$ as

$$\mathbf{x} = {}_{\mathrm{A}}x^1\mathbf{a}_1 + \cdots + {}_{\mathrm{A}}x^n\mathbf{a}_n \ . \tag{D.2}$$

The coordinate map $\mathcal{K}_{\mathrm{A}}(\mathbf{x})$ of $\mathbf{x}$ is defined as

$$\begin{array}{rccl} \mathcal{K}_{\mathrm{A}}: & V & \to & \mathbb{F}^n \\ & \cup & & \cup \\ & \mathbf{x} & \mapsto & \mathcal{K}_{\mathrm{A}}(\mathbf{x}) \coloneqq {}_{\mathrm{A}}\mathbf{x} = \begin{bmatrix} {}_{\mathrm{A}}x^1 & \cdots & {}_{\mathrm{A}}x^n \end{bmatrix}^\top . \end{array} \tag{D.3}$$

The reader should note the difference between a *vector* $\mathbf{x}$ and its *coordinate representation* or *coordinate tuple* ${}_{\mathrm{A}}\mathbf{x}$. The coordinate mapping $\mathcal{K}_{\mathrm{A}}(\cdot)$ is a linear map and bijective and thus has an inverse $\mathcal{K}_{\mathrm{A}}^{-1}(\cdot)$.

D.4 Matrix Representation of a Linear Map

We briefly explain the matrix representation $\mathbf{T}$ of a linear map $\mathcal{T}$. For the remainder of this chapter, we consider mostly two vectors spaces $(V, \mathbb{F})$ and $(W, \mathbb{F})$ with dimensions $\dim_{\mathbb{F}}(V) = n$ and $\dim_{\mathbb{F}}(W) = m$ and bases $\bar{\bar{\mathbf{e}}}^{\mathrm{A}} \coloneqq \{\mathbf{a}_i\}$ and $\bar{\bar{\mathbf{e}}}^{\mathrm{B}} \coloneqq \{\mathbf{b}_j\}$, respectively.

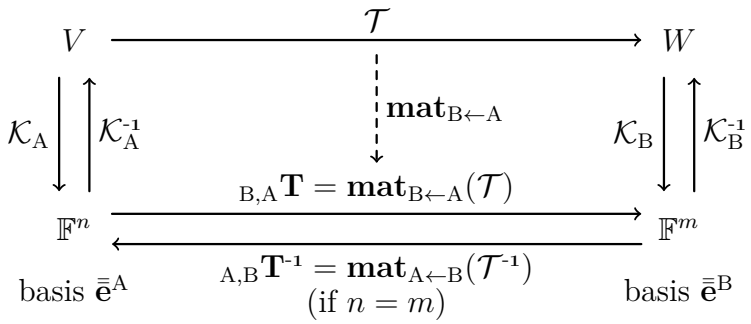
A linear map $\mathcal{T}$ from the linear space V to the linear space W is given as

$$\begin{array}{rccc} \mathcal{T}: & V & \to & W \\ & \cup & & \cup \\ & \{\mathbf{a}_i\} & & \{\mathbf{b}_i\} \\ & \mathbf{x} & \mapsto & \mathcal{T}(\mathbf{x}) = \mathbf{y} \ . \end{array} \tag{D.4}$$

Let the input argument of $\mathcal{T}$ be $\mathbf{x} \in V$ and is be mapped to coordinates ${}_{\mathrm{A}}\mathbf{x} \in \mathbb{F}^n$ in basis A and the image $\mathbf{y} = \mathcal{T}(\mathbf{x}) \in W$ is mapped to coordinates ${}_{\mathrm{B}}\mathbf{y} \in \mathbb{F}^m$ in basis B. There exists a map ${}_{\mathrm{B,A}}\mathbf{T}$ corresponding to $\mathcal{T}$ which maps the coordinate tuples represented in basis A to coordinate tuples in basis B, that is,

$$\begin{array}{rccc} {}_{\mathrm{B,A}}\mathbf{T}: & \mathbb{F}^n & \to & \mathbb{F}^m \\ & \cup & & \cup \\ & {}_{\mathrm{A}}\mathbf{x} & \mapsto & {}_{\mathrm{B,A}}\mathbf{T}({}_{\mathrm{A}}\mathbf{x}) = {}_{\mathrm{B}}\mathbf{x} \ . \end{array} \tag{D.5}$$

Using the bijective coordinate maps $\mathcal{K}_{\mathrm{A}}$ and $\mathcal{K}_{\mathrm{B}}$ introduced in appendix D.3, the commutative diagram of interest looks as the following:



(D.6)

By the above diagram, the linear map ${}_{\mathrm{B,A}}\mathbf{T}$ results in the following composition of functions:

$$\boxed{{}_{\mathrm{B,A}}\mathbf{T} = \mathcal{K}_{\mathrm{B}} \circ \mathcal{T} \circ \mathcal{K}_{\mathrm{A}}^{-1} \ .} \tag{D.7}$$

Due to the fact that the composition of linear maps remains a linear map, the map ${}_{\mathrm{B,A}}\mathbf{T}$ is linear too. The map from the linear function $\mathcal{T}$ to ${}_{\mathrm{B,A}}\mathbf{T}$ is denoted as $\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}$, that is, ${}_{\mathrm{B,A}}\mathbf{T} \coloneqq \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})$, and is shown as a dashed line in (D.6).

As will be seen in the following, the linear map $\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})$ can be represented as a matrix in $\mathbb{F}^{m\times n}$, hence the word "mat" for its notation. Take two vectors $\mathbf{x} = {}_{\mathrm{A}}x^i\mathbf{a}_i \in V$ and $\mathbf{y} = {}_{\mathrm{B}}y^i\mathbf{b}_i \in W$. Then it follows that

$$\begin{aligned}
{}_{\mathrm{B}}\mathbf{y} = {}_{\mathrm{B,A}}\mathbf{T} \circ \mathcal{K}_{\mathrm{A}}(\mathbf{x}) = \mathcal{K}_{\mathrm{B}} \circ \mathcal{T} \circ \mathcal{K}_{\mathrm{A}}^{-1} \circ \mathcal{K}_{\mathrm{A}}(\mathbf{x}) = \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{x}) &= \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_i)\ {}_{\mathrm{A}}x^i && \text{(D.8)}\\
&= \begin{bmatrix} \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_1) & \dots & \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_n) \end{bmatrix} {}_{\mathrm{A}}\mathbf{x} && \text{(D.9)}\\
&= \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})\ {}_{\mathrm{A}}\mathbf{x} && \text{(D.10)}
\end{aligned}$$

and therefore ${}_{\mathrm{B,A}}\mathbf{T} = \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T}) \coloneqq [\ \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_1), \cdots, \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_n)\ ] \in \mathbb{F}^{m\times n}$.

The images of $\mathcal{T}(\mathbf{a}_i) \in W$ can be represented in basis B as

$$\left.\begin{aligned}
\mathcal{T}(\mathbf{a}_1) &= T^1{}_1\mathbf{b}_1 + T^2{}_1\mathbf{b}_2 + \cdots + T^m{}_1\mathbf{b}_m\\
&\vdots\\
\mathcal{T}(\mathbf{a}_j) &= T^1{}_j\mathbf{b}_1 + T^2{}_j\mathbf{b}_2 + \cdots + T^m{}_j\mathbf{b}_m\\
&\vdots\\
\mathcal{T}(\mathbf{a}_n) &= T^1{}_n\mathbf{b}_1 + T^2{}_n\mathbf{b}_2 + \cdots + T^m{}_n\mathbf{b}_m
\end{aligned}\right\} \quad \text{or:} \quad \mathcal{T}(\mathbf{a}_i) = \mathbf{b}_j\ T^j{}_i\ . \tag{D.11}$$

The coordinate tuple $[\ T^1{}_i, \cdots, T^m{}_i\ ]^\top$ corresponds to the vector $\mathcal{T}(\mathbf{a}_i)$ represented in basis B, that is, $\mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_1)$. This yields the following matrix ${}_{\mathrm{B,A}}\mathbf{T}$:

$$\boxed{\begin{aligned}
{}_{\mathrm{B,A}}\mathbf{T} = \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T}) &= [\ \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_1), \cdots, \mathcal{K}_{\mathrm{B}} \circ \mathcal{T}(\mathbf{a}_n)\ ] \in \mathbb{F}^{m\times n}\\
&= \begin{bmatrix} T^1{}_1 & T^1{}_2 & \dots & T^1{}_n\\ T^2{}_1 & T^2{}_2 & \dots & T^2{}_n\\ \vdots & \vdots & \ddots & \vdots\\ T^m{}_1 & T^m{}_2 & \dots & T^m{}_n \end{bmatrix} \in \mathbb{R}^{m\times n} \qquad \text{or:} \quad {}_{\mathrm{B,A}}\mathbf{T} = \left[T^i{}_j\right]\ ,
\end{aligned}} \tag{D.12}$$

and results in the matrix-vector multiplication:

$$\boxed{{}_{\mathrm{B}}\mathbf{y} = {}_{\mathrm{B,A}}\mathbf{T}\ {}_{\mathrm{A}}\mathbf{x}\ .} \tag{D.13}$$

The left subscripts of ${}_{\mathrm{B,A}}\mathbf{T}$ denote from which basis in the domain to which basis in the codomain this function maps, that is, from basis A to basis B. In the field of multilinear algebra, the linear map $\mathcal{T}$ is represented by the tensor $\mathbf{T} \coloneqq T^i{}_j\ \mathbf{b}_i \otimes \boldsymbol{\alpha}^j$, which is no more a matrix but a multilinear map where $\mathbf{b}_i \otimes \boldsymbol{\alpha}^j$ is called the *tensor product* between basis

vector $\mathbf{b}_i$ and the dual basis vector $\boldsymbol{\alpha}^j$ to $\mathbf{a}_j$. The dual basis is briefly introduced later in this chapter.

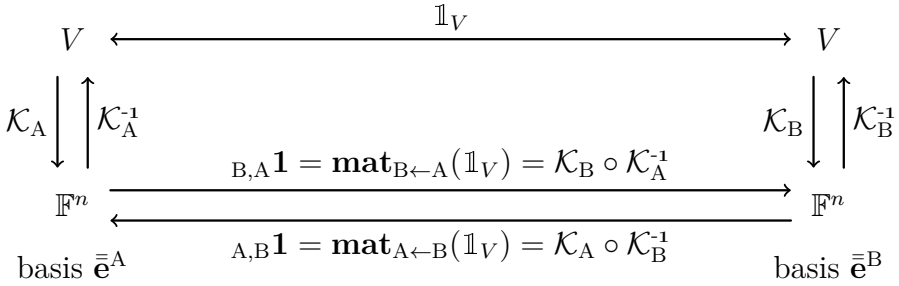
The inverse $\mathcal{T}^{-1} : W \to V$ of the linear map $\mathcal{T}$ exists if $\dim_{\mathbb{F}}(V) = n = m = \dim_{\mathbb{F}}(W)$ and its coordinate representation from basis B to basis A is given as ${}_{\mathrm{A,B}}\mathbf{T}^{-1} = \mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{B}}(\mathcal{T}^{-1})$ with the notation ${}_{\mathrm{A,B}}\mathbf{T}^{-1} = [{T^{-1}}^i{}_j]$ and the property ${}_{\mathrm{A,B}}\mathbf{T}^{-1} = ({}_{\mathrm{B,A}}\mathbf{T})^{-1}$. The reader should take attention between the corresponding notations ${}_{\mathrm{A,B}}(\mathbf{T}^{-1}) \leftrightarrow {}_{\mathrm{A,B}}\mathbf{T}^{-1}$ which denotes the matrix representation of the inverse map $\mathcal{T}^{-1}$ and the similar corresponding notations $({}_{\mathrm{B,A}}\mathbf{T})^{-1} \leftrightarrow {}_{\mathrm{B,A}}\mathbf{T}^{-1}$ which means the inversion of the matrix ${}_{\mathrm{B,A}}\mathbf{T}$. In correspondence to (D.13) it follows that

$$\boxed{{}_{\mathrm{A}}\mathbf{x} = {}_{\mathrm{A,B}}\mathbf{T}^{-1}\ {}_{\mathrm{B}}\mathbf{y}\ .} \tag{D.14}$$

In multilinear algebra, the map $\mathcal{T}^{-1}$ is represented as a tensor $\mathbf{T}^{-1} \coloneqq {T^{-1}}^i{}_j\ \mathbf{a}_i \otimes \boldsymbol{\beta}^j$ where $\boldsymbol{\beta}^j$ is the dual basis vector to $\mathbf{b}_j$.

D.5 Matrix Representation of the Identity Map or Coordinate Transformation

The change of a coordinate tuple ${}_{\mathrm{A}}\mathbf{x}$ in basis A of a vector $\mathbf{x} \in V$ to a new representation ${}_{\mathrm{B}}\mathbf{x}$ in basis B can be obtained by the identity map $\mathbb{1}_V$ illustrated in the following commutative diagram:



(D.15)

Then the matrix representation ${}_{\mathrm{B,A}}\mathbf{1}$ by (D.12) yields

$${}_{\mathrm{B,A}}\mathbf{1} = \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathbb{1}_V) = [\ \mathcal{K}_\mathrm{B}(\mathbf{a}_1), \ldots, \mathcal{K}_\mathrm{B}(\mathbf{a}_n)\ ] \in \mathbb{F}^{n\times n}\ . \tag{D.16}$$

This matrix is denoted by Glocker [60] in a way which is useful for manipulating coordinate tuples, that is,

$$\boxed{\mathbf{A}_\mathrm{BA} \coloneqq {}_{\mathrm{B,A}}\mathbf{1} = [{A_\mathrm{BA}}^i{}_j] = \mathcal{K}_\mathrm{B} \circ \mathcal{K}_\mathrm{A}^{-1} \in \mathbb{F}^{n\times n}\ .} \tag{D.17}$$

The matrix $\mathbf{A}_\mathrm{BA}$ is called *coordinate transformation matrix*, which changes a coordinate tuple ${}_{\mathrm{A}}\mathbf{x}$ to the tuple ${}_{\mathrm{B}}\mathbf{x}$. The vector $\mathbf{x}$ remains physically the same by the definition of the identity map and only the representation $\mathbf{x}$ changes from basis A to basis B. The inverse coordinate map is given as $\mathbf{A}_\mathrm{AB} = {\mathbf{A}_\mathrm{BA}}^{-1}$.

To summarize, by (D.11), the bases and coordinate tuples transform in the following way:

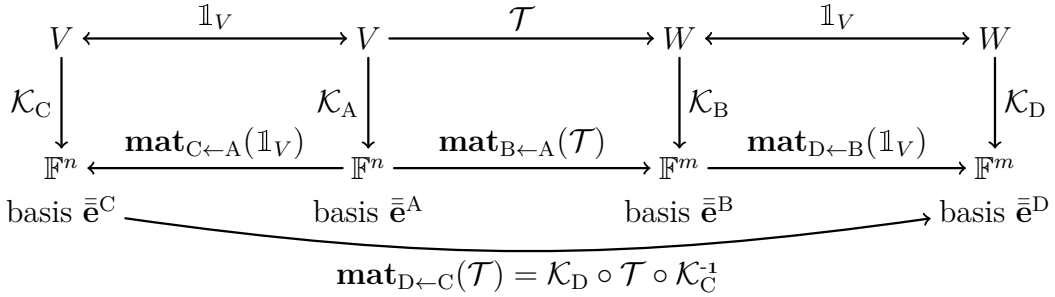
$$\boxed{\begin{aligned} {}_{\mathrm{B}}\mathbf{x} &= \mathbf{A}_{\mathrm{BA}}\ {}_{\mathrm{A}}\mathbf{x}\ , & {}_{\mathrm{A}}\mathbf{x} &= \mathbf{A}_{\mathrm{AB}}\ {}_{\mathrm{B}}\mathbf{x}\ , & &\text{(coordinate tuples)} \\ \mathbf{a}_i &= \mathbf{b}_j\ {A_{\mathrm{BA}}}^j{}_i\ , & \mathbf{b}_i &= \mathbf{a}_j\ {A_{\mathrm{AB}}}^j{}_i & &\text{(basis vectors)} \end{aligned}} \tag{D.18}$$

$$\text{with:}\quad \mathbf{A}_{\mathrm{AB}} = {\mathbf{A}_{\mathrm{BA}}}^{-1} \tag{D.19}$$

In multilinear algebra this transformation is often called *contravariant basis transformation*. Because the basis transforms with the inverse ${\mathbf{A}_{\mathrm{BA}}}^{-1}$ as $\mathbf{b}_i = \mathbf{a}_j\ {A_{\mathrm{AB}}}^j{}_i$, the coordinate tuple is said to *transform contravariantly*.

D.6 Basis Change of a Linear Map

Assume that the matrix representation $\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})$ is given which maps coordinate tuples in basis A to coordinate tuples in basis B. The relation to a new representation $\mathbf{mat}_{\mathrm{D}\leftarrow\mathrm{C}}(\mathcal{T})$ can be found by the following commutative diagram:



(D.20)

The basis change for the matrix $\mathbf{mat}_{\mathrm{D}\leftarrow\mathrm{C}}(\mathcal{T})$ is then given as:

$$\mathbf{mat}_{\mathrm{D}\leftarrow\mathrm{C}}(\mathcal{T}) = \mathbf{mat}_{\mathrm{D}\leftarrow\mathrm{B}}(\mathbb{1}_V)\ \mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})\ \mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{C}}(\mathbb{1}_V^{-1}). \tag{D.21}$$

With the relationship that $\mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{C}}(\mathbb{1}_V^{-1}) = \mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{C}}(\mathbb{1}_V) = \mathbf{mat}_{\mathrm{C}\leftarrow\mathrm{A}}(\mathbb{1}_V)^{-1}$ and (D.12) and (D.17), equation (D.21) translates to the matrix multiplication

$$\boxed{{}_{\mathrm{D,C}}\mathbf{T} = \mathbf{A}_{\mathrm{DB}}\ {}_{\mathrm{B,A}}\mathbf{T}\ \mathbf{A}_{\mathrm{AC}}\ .} \tag{D.22}$$

The next sections address the question how a rotation matrix is related to a coordinate transformation.

D.6.1 Relation between Linear Maps and Coordinate Transformation

Assume that a linear map $\mathcal{T} : V \to V$ with $\dim_{\mathbb{F}}(V) = n$ represented as ${}_{\mathrm{B,A}}\mathbf{T}$ from basis A to B has the property that it also transforms the basis vectors $\mathbf{a}_i$ into the basis vectors $\mathbf{b}_i$, that is,

$$\mathcal{T}(\mathbf{a}_i) = \mathbf{b}_i \quad \forall i \in [1\,;\, n]\ , \tag{D.23}$$

then by (D.12) follows that ${}_{\mathrm{B,A}}\mathbf{T} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. If ${}_{\mathrm{B,A}}\mathbf{T}$ is represented in a single basis A by a basis change (D.22), it simply follows that

$$ {}_{\mathrm{A,A}}\mathbf{T} = \mathbf{A}_{\mathrm{AB}}\ {}_{\mathrm{B,A}}\mathbf{T}\ \mathbf{A}_{\mathrm{AA}} = \mathbf{A}_{\mathrm{AB}}\ {}_{\mathrm{B,A}}\mathbf{T} = \mathbf{A}_{\mathrm{AB}}\ . \tag{D.24} $$

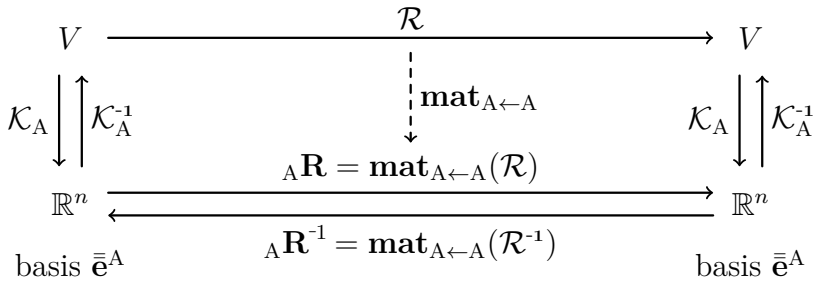
Hence, the specific linear map $\mathcal{T}$ represented in basis A, that is, ${}_{\mathrm{A,A}}\mathbf{T}$, is the coordinate transformation $\mathbf{A}_{\mathrm{AB}}$.

D.7 Rotation Matrix

Without loss of generality, we assume in the following that the field $\mathbb{F}$ of the vector space V is $\mathbb{R}$. Rotating a vector $\mathbf{x} \in V$ is achieved by a bijective map denoted as $\mathcal{R} : V \to V$ which is called rotation and yields a new vector $\mathbf{y} \in V$ as

$$ \mathbf{y} = \mathcal{R}(\mathbf{x})\ . \tag{D.25} $$

It makes most sense to represent a rotation $\mathcal{R}$ in only a single coordinate system A, that is, ${}_{\mathrm{A}}\mathbf{R} \coloneqq {}_{\mathrm{A,A}}\mathbf{R} \coloneqq \mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{A}}(\mathcal{R})$ such that ${}_{\mathrm{A}}\mathbf{y} = {}_{\mathrm{A}}\mathbf{R}\ {}_{\mathrm{A}}\mathbf{x}$ is the rotated version of the tuple ${}_{\mathrm{A}}\mathbf{x}$. The commutative diagram then looks as the following:



(D.26)

A general rotation $\mathcal{R}$ preserves length and angles which can be expressed by preserving a given inner product $(\cdot\,|\,\cdot) : V \times V \to \mathbb{R}$ as $(\mathcal{R}(\mathbf{x})\,|\,\mathcal{R}(\mathbf{y})) = (\mathbf{x}\,|\,\mathbf{y})$ (cf. (2.4)). The inner product induces the norm $\|\mathbf{x}\| = \sqrt{(\mathbf{x}\,|\,\mathbf{x})}$. A rotation is called *proper* when it preserves the orientation of vectors as well. Any inner product preserving map is necessarily linear[1] and thus a rotation $\mathcal{R}$ is a linear map.

Assume that the inner product in V is given in some basis A as $(\mathbf{x}\,|\,\mathbf{y}) \coloneqq {}_{\mathrm{A}}\mathbf{x}^{\top}\ \mathbf{M}\ {}_{\mathrm{A}}\mathbf{y}$, where $\mathbf{M}$ is a symmetric positive definite matrix (metric). A rotation then needs to fulfill ${}_{\mathrm{A}}\mathbf{R}^{\top}\ \mathbf{M}\ {}_{\mathrm{A}}\mathbf{R} = \mathbf{M}$ and additionally $\det({}_{\mathrm{A}}\mathbf{R}) = 1$ if it is *proper* (cf. (2.4)). A general rotation is therefore an element of the *generalized orthogonal group* $GO(n, \mathbf{M}) \coloneqq \{\mathbf{R} \in \mathbb{R}^{n\times n} \mid \mathbf{R}^{\top}\mathbf{M}\mathbf{R} = \mathbf{M}\}$ with property $\det(\mathbf{R}) = \pm 1$. The relation between the rotation matrix ${}_{\mathrm{A}}\mathbf{R}$ and the linear map $\mathcal{R}$ is given by (D.7) as

$$ \boxed{{}_{\mathrm{A}}\mathbf{R} = \mathcal{K}_{\mathrm{A}} \circ \mathcal{R} \circ \mathcal{K}_{\mathrm{A}}^{-1}\ .} \tag{D.27} $$

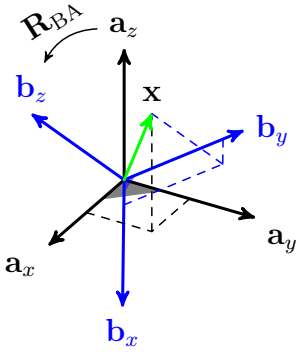
The matrix representation ${}_{\mathrm{A}}\mathbf{R}$ of the rotation $\mathcal{R}$ in a different basis B can be obtained by the relationship (D.22) as

$$ \boxed{{}_{\mathrm{B}}\mathbf{R} = \mathbf{A}_{\mathrm{BA}}\ {}_{\mathrm{A}}\mathbf{R}\ {\mathbf{A}_{\mathrm{BA}}}^{-1}\ .} \tag{D.28} $$

[1] This can be checked by evaluating $\|\mathcal{R}(\mathbf{v}+\mathbf{w}) - \mathcal{R}(\mathbf{v}) - \mathcal{R}(\mathbf{w})\|^2$ and $\|\mathcal{R}(\lambda\mathbf{v}) - \lambda\mathcal{R}(\mathbf{v})\|^2$ for any two vectors $\mathbf{v}, \mathbf{w} \in V$ which results in zero for both terms from which follows linearity.

If a new basis B with basis vectors $\{\mathbf{b}_i\}$ is defined by a given *proper* rotation $\mathcal{R} \coloneqq \mathcal{R}_{\mathrm{BA}}$ and its inverse $\mathcal{R}^{-1} = \mathcal{R}_{\mathrm{AB}}$ as

$$\mathbf{b}_i \coloneqq \mathcal{R}_{\mathrm{BA}}(\mathbf{a}_i) \quad \Rightarrow \mathbf{a}_i = \mathcal{R}_{\mathrm{AB}}(\mathbf{b}_i) \quad \forall i \in [1\,;\, n]\ . \tag{D.29}$$



Then, by (D.24) one obtains $\mathbf{A}_{\mathrm{AB}} = {}_{\mathrm{A}}\mathbf{R}_{\mathrm{BA}}$ and by (D.28) we get

$$\boxed{{}_{\mathrm{B}}\mathbf{R}_{\mathrm{BA}} = {}_{\mathrm{A}}\mathbf{R}_{\mathrm{BA}} = \mathbf{A}_{\mathrm{AB}}\ .} \tag{D.30}$$

Therefore, it does not matter in which basis the rotation matrix is represented. Either A or B correspond to the same transformation matrix $\mathbf{A}_{\mathrm{AB}}$. This makes sense because the Euler-Rotation theorem says that any rotation can be parametrized by a tuple $(\mathbf{n}, \varphi)$, where $\mathbf{n}$ is the rotation axis and φ the rotation angle. The rotation vector $\mathbf{n}$ is coordinate-dependent and does not change under its rotation, that is, $\mathcal{R}_{\mathrm{BA}}(\mathbf{n}) = \mathbf{n}$, from which follows ${}_{\mathrm{A}}\mathbf{n} = {}_{\mathrm{B}}\mathbf{n}$.

If the vector space is an n-dimensional Euclidean space $\mathbb{E}^n$ where $\mathbf{M} = \mathbf{I} \in \mathbb{R}^{n\times n}$, a *proper* rotation matrix belongs to the n-dimensional *special orthogonal group* $SO(n) \coloneqq \{\mathbf{R} \in \mathbb{R}^{n\times n} \mid \mathbf{R}^{\top}\mathbf{R} = \mathbf{I},\ \det(\mathbf{R}) = 1\}$.

D.8 Kinematics in a Vector Space

Assume two bases $\bar{\mathbf{e}}^{\mathrm{I}} \coloneqq \{\mathbf{e}_i^{\mathrm{I}}\}$ and $\bar{\mathbf{e}}^{\mathrm{B}} \coloneqq \{\mathbf{b}_i\}$ in vector space V are given together with a time-dependent vector $\mathbf{x}(t) = {}_{\mathrm{I}}x^i(t)\mathbf{e}_i^{\mathrm{I}}$ represented in basis I or in basis B as $\mathbf{x}(t) = {}_{\mathrm{B}}x^i(t)\mathbf{b}_i(t)$. The reference basis I is special in the sense that it is not dependent on time and is thus considered non-moving. The basis vectors $\mathbf{b}_i(t)$ are dependent on time and they can be expressed by a coordinate transformation as

$$\mathbf{A}_{\mathrm{BI}}(t) \coloneqq [A^i{}_j(t)]\ , \qquad \mathbf{A}_{\mathrm{IB}}(t) \coloneqq [A^{-1i}{}_j(t)] \tag{D.31}$$

$$\mathbf{e}_i^{\mathrm{I}} = \mathbf{b}_j\ A^j{}_i(t)\ , \qquad \mathbf{b}_i = \mathbf{e}_j^{\mathrm{I}}\ A^{-1j}{}_i(t)\ . \tag{D.32}$$

The time dependence $(\cdot)(t)$ is omitted in the following. Taking the time derivative $\frac{\mathrm{d}}{\mathrm{d}t}(\cdot) \coloneqq (\cdot)^{\bullet}$ of vector $\mathbf{x}(t)$ results in the *absolute velocity* vector $(\mathbf{x})^{\bullet} = \dot{\mathbf{x}}$, that is,

$$\mathbf{x}(t) \coloneqq {}_{\mathrm{I}}x^i(t)\mathbf{e}_i^{\mathrm{I}}\ , \qquad \mathbf{x}(t) \coloneqq {}_{\mathrm{B}}x^i(t)\mathbf{b}_i(t) \tag{D.33}$$

$$\Downarrow \qquad\qquad \Downarrow \tag{D.34}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{x}) = ({}_{\mathrm{I}}x^i)^{\bullet}\ \mathbf{e}_i^{\mathrm{I}}\ , \qquad \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{x}) = ({}_{\mathrm{B}}x^i)^{\bullet}\ \mathbf{b}_i + {}_{\mathrm{B}}x^i\ (\mathbf{b}_i)^{\bullet}\ . \tag{D.35}$$

The total time derivative $(\mathbf{b}_i)^{\bullet}$ can be obtained from (D.32) as

$$(\mathbf{b}_i)^{\bullet} = \mathbf{e}_j^{\mathrm{I}}(A^{-1j}{}_i)^{\bullet} = \mathbf{b}_k\ A^k{}_j(A^{-1j}{}_i)^{\bullet}. \tag{D.36}$$

From (D.36), equation (D.35) follows to

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{x}) \coloneqq \dot{\mathbf{x}} = ({}_{\mathrm{B}}x^i)^{\bullet}\ \mathbf{b}_i + A^k{}_j(A^{-1j}{}_i)^{\bullet}\ {}_{\mathrm{B}}x^i\ \mathbf{b}_k \tag{D.37}$$

$$= \left(({}_{\mathrm{B}}x^k)^{\bullet} + A^k{}_j(A^{-1j}{}_i)^{\bullet}\ {}_{\mathrm{B}}x^i\right)\ \mathbf{b}_k \tag{D.38}$$

The coordinate representation ${}_{\mathrm{B}}(\dot{\mathbf{x}}) = [\dots, {}_{\mathrm{B}}\dot{x}^i, \dots]^\top$ of the velocity vector $\dot{\mathbf{x}} = {}_{\mathrm{B}}\dot{x}^i \mathbf{b}_i$ in basis B yields

$$\boxed{{}_{\mathrm{B}}\dot{x}^k = ({}_{\mathrm{B}}x^k)^\bullet + A^k{}_j \, (A^{-1}{}^j{}_i)^\bullet \, {}_{\mathrm{B}}x^i \qquad \Leftrightarrow \qquad {}_{\mathrm{B}}(\dot{\mathbf{x}}) = ({}_{\mathrm{B}}\mathbf{x})^\bullet + \mathbf{A}_{\mathrm{BI}}(\mathbf{A}_{\mathrm{BI}}{}^{-1})^\bullet \, {}_{\mathrm{B}}\mathbf{x} \; .} \tag{D.39}$$

Equation (D.39) is called the *Euler differentiation rule* which accounts for a moving reference frame.

D.9 The Dual Space

The dual vector space V^* (abbreviated as dual space) for a finite vector space V with $\dim_{\mathbb{F}}(V) = n$ is of utmost interest in mechanics.

To define the dual space, one needs the definition of a *functional*. A function f defined as $f : V \to \mathbb{F}$ is called a *functional* and maps a vector $\mathbf{x} \in V$ to the underlying field $\mathbb{F}$, which is most often $\mathbb{R}$. A functional is called linear if it satisfies the properties in appendix D.2. A linear functional is sometimes also called *1-form*, *linear-form* or *covector* or *covariant* 1-tensor.

Definition D.6 *(Dual Space)*:
The dual vector space V^* to the vector space V is defined as the set of all linear *functionals* on the vector space V, that is,

$$\boxed{V^* \coloneqq \{f : V \to \mathbb{F}, \quad f \text{ linear}\}.} \tag{D.40}$$

A linear functional f applied on a vector $\mathbf{x} \coloneqq x^i \mathbf{a}_i \in V$ yields $f(\mathbf{x}) = f(\mathbf{a}_i)x^i$. The scalars $f(\mathbf{a}_i) \in \mathbb{R}$ define the functional uniquely. Since the functional is a linear map, the matrix representation $\mathbf{mat}_{1\leftarrow \mathrm{A}}(f)$ of the functional f (cf. appendix D.4) is given by (D.12) as

$$\mathbf{mat}_{1\leftarrow \mathrm{A}}(f) = [f(\mathbf{a}_1), f(\mathbf{a}_2), \dots, f(\mathbf{a}_n)] \; . \tag{D.41}$$

The tuple ${}_1\mathbf{f} \coloneqq \mathbf{mat}_{1\leftarrow \mathrm{A}}(f)$ maps ${}_{\mathrm{A}}\mathbf{x}$ to the representation in the trivial basis $1 \in \mathbb{R}$ as $f(\mathbf{x}) = \mathbf{f}^\top {}_{\mathrm{A}}\mathbf{x}$. Note that the addition of two functionals $\mathbf{f}, \mathbf{g} \in V^*$ is again a functional on V since

$$\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x}) = (\mathbf{f}(\mathbf{a}_i) + \mathbf{g}(\mathbf{a}_i))x^i = \mathbf{h}(\mathbf{a}_i)x^i = \mathbf{h}(\mathbf{x}) \in V^* \; . \tag{D.42}$$

Definition D.7 *(Duality Pairing)*:
The *duality pairing* of a dual vector $\mathbf{f} \in V^*$ with a primal vector $\mathbf{x} \in V$ is simply the functional $\mathbf{f}$ applied to $\mathbf{x}$. The duality pairing is written as $\mathbf{f}(\mathbf{x}) \coloneqq \langle \mathbf{f} \,|\, \mathbf{x} \rangle$.

Definition D.8 *(Dual Basis)*:
Let $\bar{\bar{\mathbf{e}}}^{\mathrm{A}} = \{\mathbf{a}_i\}$ be a basis for V, then there exists a *dual basis* $\bar{\bar{\boldsymbol{\epsilon}}}^{\alpha} = \{\boldsymbol{\alpha}^i\}$ such that

$$\boxed{\langle \boldsymbol{\alpha}^i \,|\, \mathbf{a}_j \rangle = \boldsymbol{\alpha}^i(\mathbf{a}_j) = \delta^i{}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \; ,} \tag{D.43}$$

where $\delta^i{}_j$ denotes the Kronecker delta. The set of vectors $\bar{\bar{\boldsymbol{\epsilon}}}^\alpha$ are linearly independent and is a basis for V^*. Therefore the dimension of the dual space is the same, that is, $\dim_{\mathbb{F}}(V) = \dim_{\mathbb{F}}(V^*) = n$.

The dual basis vectors $\boldsymbol{\alpha}^i$ are also called *coordinate forms*, because it is a *linear functional* which selects the i^{th} coordinate in its primal basis $\mathbf{a}_i$, that is, $\boldsymbol{\alpha}^i(\mathbf{x}) = {}_{\text{A}}x^i$ for some $\mathbf{x} = {}_{\text{A}}x^i\mathbf{a}_i \in V$.

Proof: *(Dual Basis and Representation)*
Let $\mathbf{x} \in V$ and $\mathbf{g} \in V^*$ be defined as $\mathbf{g} = g_i\boldsymbol{\alpha}^i$ with dual and primal basis $\bar{\bar{\mathbf{e}}}^{\text{A}} = \{\mathbf{a}_i\}$ and $\bar{\bar{\boldsymbol{\epsilon}}}^{\text{A}} = \{\boldsymbol{\alpha}_i\}$, respectively. Let $\mathbf{g}$ be the zero functional, then

$$\langle \mathbf{g} \,|\, \mathbf{x} \rangle = 0 = \langle g_i\boldsymbol{\alpha}^i \,|\, x^i\mathbf{a}_i \rangle = g_i x^i = 0 \quad \forall \mathbf{x} \in V \ , \tag{D.44}$$

which implies that $g_i = 0$ and shows the linear independence of the vectors $\boldsymbol{\alpha}^i$ and thus $\dim_{\mathbb{F}}(V) = \dim_{\mathbb{F}}(V^*)$.
If $\mathbf{f} \in V^*$ and define $f_i = \langle \mathbf{f} \,|\, \mathbf{a}_i \rangle$ then

$$\langle \mathbf{f} \,|\, \mathbf{x} \rangle = \langle \mathbf{f} \,|\, x^i\mathbf{a}_i \rangle = \langle \mathbf{f} \,|\, \mathbf{a}_i \rangle x^i = f_i x^i = f_i \langle \boldsymbol{\alpha}^i \,|\, \mathbf{x} \rangle = \langle f_i\boldsymbol{\alpha}^i \,|\, \mathbf{x} \rangle, \tag{D.45}$$

which holds for every $\mathbf{x}$ and shows that every linear functional $\mathbf{f} \in V^*$ can be expressed through its basis

$$\boxed{\mathbf{f} = \langle \mathbf{f} \,|\, \mathbf{a}_i \rangle \boldsymbol{\alpha}_i.} \tag{D.46}$$

The bidual space V^{**} is analogue to the dual space V^* and defined as the space of all linear functionals which map from the dual space to the underlying field $\mathbb{F}$ as

$$\boxed{V^{**} \coloneqq \{f : V^* \to \mathbb{F}, \quad f \text{ linear}\} \ .} \tag{D.47}$$

D.9.1 Natural Injective Map into V^{**} and Reflexivity

We search a map $\varphi : V \to V^{**}$ such that the functional $\varphi(\mathbf{x}) : V^* \to \mathbb{F}$ is such that $\varphi(\mathbf{x})(\mathbf{f}) \in \mathbb{F}$ for some $\mathbf{f} \in V^*$. The most natural way to define the mapping $\varphi(\mathbf{x})(\mathbf{f}) \in \mathbb{F}$ is to define it as $\varphi(\mathbf{x})(\mathbf{f}) \coloneqq \mathbf{f}(\mathbf{x}) \in \mathbb{F}$. This results in the map φ as

$$\boxed{\begin{array}{ccccc} \varphi: & V & \to & V^{**} \\ & \cup & & \cup \\ & \mathbf{x} & \mapsto & \varphi(\mathbf{x}) \coloneqq \langle \bullet \,|\, \mathbf{x} \rangle \ . \end{array}} \tag{D.48}$$

Without defining additional structure, it can be shown that the map φ is linear and injective (one-to-one), also for *infinte*-dimensional vector spaces. For the *finite*-dimensional case, as assumned, φ is bijective. That means we can uniquely associate to each vector $\mathbf{x}$ a bidual vector $\langle \bullet \,|\, \mathbf{x} \rangle$ and therefore V is *isomoprhic* to V^{**} by the *natural isomorphism* φ, denoted as $V \cong V^{**}$. This is called *reflexivity* and all *finite* dimensional vector spaces are reflexive.

Proof: *(Reflexivity (finite case))*
With $\mathbf{x} = x^i \mathbf{a}_i \in V$ and $\mathbf{y} = y^i \mathbf{a}_i \in V$ and $a \in \mathbb{F}$ and basis $\bar{\bar{\mathbf{e}}}^{\mathrm{A}} = \{\mathbf{a}_i\}$ and dual basis $\bar{\bar{\boldsymbol{\epsilon}}}^{\alpha} = \{\boldsymbol{\alpha}_i\}$ and $\mathbf{f} \in V^*$ it follows that

$$\begin{aligned} \varphi(\mathbf{x}+\mathbf{y})(\mathbf{f}) &= \mathbf{f}(\mathbf{x}+\mathbf{y}) = \mathbf{f}(\mathbf{x}) + \mathbf{f}(\mathbf{y}) = \varphi(\mathbf{x})(\mathbf{f}) + \varphi(\mathbf{y})(\mathbf{f}) \\ \varphi(a\mathbf{x})(\mathbf{f}) &= \mathbf{f}(a\mathbf{x}) = a\mathbf{f}(\mathbf{x}) = a\varphi(\mathbf{x})(\mathbf{f}), \end{aligned}$$

from which follows linearity. Given $\varphi(\mathbf{x}) = \varphi(\mathbf{y})$, it follows $\varphi(\mathbf{x})(\boldsymbol{\alpha}_i) = \varphi(\mathbf{y})(\boldsymbol{\alpha}_i) \quad \Rightarrow \boldsymbol{\alpha}_i(\mathbf{x}) = x^i = y^i = \boldsymbol{\alpha}_i(\mathbf{y})\ \forall i \in [i\,;\, n]$, which shows injectivity, and because $\dim_{\mathbb{F}}(V) = \dim_{\mathbb{F}}(V^{**}) = n$ it is also bijective. ■

It is important to note that the duality pairing $\langle \hat{\mathbf{x}} \,|\, \boldsymbol{\alpha} \rangle$ with $\hat{\mathbf{x}} = \varphi(\mathbf{x}) \in V^{**},\ \boldsymbol{\alpha} \in V^*,\ \mathbf{x} \in V$ is equivalent to the pairing $\langle \boldsymbol{\alpha} \,|\, \mathbf{x} \rangle$.

D.9.2 Injective Map into the Dual Space

For the injection (D.48) into the bidual space V^{**}, it was not necessary to rely on any sort of basis. Defining a map from the primal space V to the dual space V^* is in general not possible without relying on a basis. That is exactly where the role of a *bilinear form* comes into play. Abstracting the notion of an *inner product* in a vector space V over the field $\mathbb{F}$ is that of a *bilinear form* or *bilinear pairing*.

A *bilinear form* ϕ is in general a function $\phi : V \times W \to \mathbb{F}$ with vector spaces V and W over the same field $\mathbb{F}$. A bilinear form is linear in both arguments. The duality pairing is a bilinear form $\langle \cdot \,|\, \cdot \rangle : V^* \times V \to \mathbb{F}$. A bilinear form $\phi : V \times V \to \mathbb{R}$ in a real vector space ($\mathbb{A} \coloneqq \mathbb{R}$) which is symmetric, that is, $\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{y}, \mathbf{x})$ and positive definite, that is, $\phi(\mathbf{x}, \mathbf{x}) > 0\ \forall \mathbf{x} \neq \mathbf{0}$, is called *inner product* in the vector space V. For a vector space V with a complex field $\mathbb{F} \coloneqq \mathbb{C}$, the concept of a bilinear form needs to be generalized to the notion of a sesquilinear form to become an inner product in V.

Any bilinear form $\phi : V \times V \to \mathbb{F}$ can be used to define a map ϕ_V as

$$\begin{array}{rccl} \phi_V : & V & \to & X \subseteq V^* \\ & \cup & & \cup \\ & \mathbf{x} & \mapsto & \phi_V(\mathbf{x}) \coloneqq \phi(\mathbf{x}, \bullet)\ , \end{array} \tag{D.49}$$

which maps a primal vector $\mathbf{x}$ to a dual vector $\phi_V(\mathbf{x})$. If ϕ_V is bijective then ϕ_V defines an isomorphism from the primal space V to a subspace X in the dual space V^* for *infinite*-dimensional vector spaces and to the whole space V^* for *finite*-dimensional vector spaces.

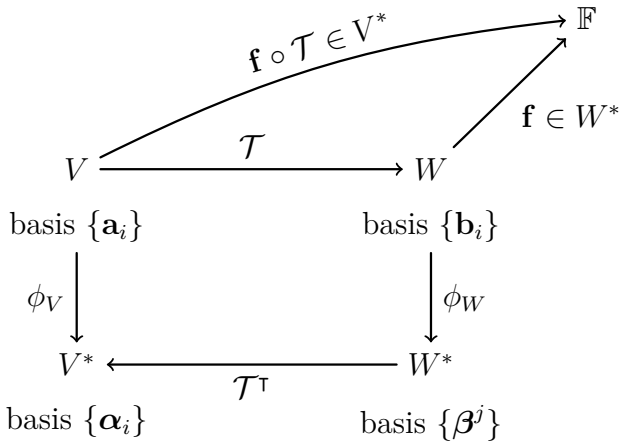
Any inner product therefore defines an isomorphism from V to its dual V^* in the finite case. An inner product also induces a norm by $\|\mathbf{x}\| = \sqrt{(\mathbf{x} \,|\, \mathbf{x})}$ which is useful to measure lengths. Specifying an inner product is done by representing it in some basis, lets say A and B, which yields $(\mathbf{x} \,|\, \mathbf{y}) = ({}_{\mathrm{A}}x^i \mathbf{a}_i \,|\, {}_{\mathrm{B}}y^j \mathbf{b}_j) = {}_{\mathrm{A}}x^i {}_{\mathrm{B}}y^j (\mathbf{a}_i \,|\, \mathbf{b}_j) = {}_{\mathrm{A}}\mathbf{x}^{\top}\ {}_{\alpha,\mathrm{B}}\mathbf{M}\ {}_{\mathrm{B}}\mathbf{y}$, where ${}_{\alpha,\mathrm{B}}\mathbf{M} = [M_{ij}] \in \mathbb{R}^{n \times n}$ with $M_{ij} = (\mathbf{a}_i \,|\, \mathbf{b}_j)$ denotes the metric and $\mathbf{M} = M_{ij} \boldsymbol{\alpha}^i \otimes \boldsymbol{\beta}^j$ is the metric tensor.

D.9.3 Transpose of a Linear Map

The transpose $\mathcal{T}^\intercal : W^* \to V^*$ of a linear map $\mathcal{T} : V \to W$ is defined on the relationship between the duality pairing as

$$\boxed{\langle\, \mathcal{T}^\intercal(\mathbf{f}) \,|\, \mathbf{v} \,\rangle = \langle\, \mathbf{f} \,|\, \mathcal{T}(\mathbf{v}) \,\rangle \qquad \forall \mathbf{f} \in W^*, \mathbf{v} \in V \ .} \tag{D.50}$$

This can also be written as $\mathcal{T}^\intercal(\mathbf{f}) = \mathbf{f} \circ \mathcal{T}$ which is visualized in the following commutative diagram:



Remark that, the adjoint map $\mathcal{T}^\star : W \to V$ of a linear map is defined on the inner product and not the same as the transpose defined above. The next step is to derive the matrix representation $\mathbf{mat}_{\alpha\leftarrow\beta}(\mathcal{T}^\intercal)$ of the transpose map. This is done the same way as for any linear map as described in appendix D.4. Assume $\mathbf{x} \in V$ and $\mathbf{y} = \mathcal{T}(\mathbf{x}) \in W$ and $\mathbf{x} = {}_{\mathrm{A}}x^i\ \mathbf{a}_i$ and $\mathbf{y} = {}_{\mathrm{B}}y^i\ \mathbf{b}_i$. Note also that ${}_{\mathrm{B}}\mathbf{y} = {}_{\mathrm{B,A}}\mathbf{T}\ {}_{\mathrm{A}}\mathbf{x}$ or ${}_{\mathrm{B}}y^i = T^i{}_j\ {}_{\mathrm{A}}x^j$ and accordingly the basis as $\mathcal{T}(\mathbf{a}_i) = \mathbf{b}_j\ T^j{}_i$ and by (D.12) we have

$$(\mathcal{T}^\intercal(\boldsymbol{\beta}^i))(\mathbf{x}) = \boldsymbol{\beta}^i \circ \mathcal{T}(\mathbf{x}) = \boldsymbol{\beta}^i(\mathbf{y}) = {}_B y^i = T^i{}_j\ {}_{\mathrm{A}}x^j = \boldsymbol{\alpha}^j(\mathbf{x})\ T^i{}_j \ , \tag{D.51}$$

which holds for all $\mathbf{x}$. Therefore, $\mathcal{T}^\intercal(\boldsymbol{\beta}^i) = \boldsymbol{\alpha}^j\ T^i{}_j$ and this is exactly the transpose of the relationship $\mathcal{T}(\mathbf{a}_i) = \mathbf{b}_j\ T^j{}_i$ in (D.11). Therefore, the matrix representation ${}_{\alpha,\beta}(\mathbf{T}^\intercal) \coloneqq \mathbf{mat}_{\alpha\leftarrow\beta}(\mathcal{T}^\intercal)$ yields

$$\boxed{{}_{\alpha,\beta}\mathbf{T}^\intercal = \left({}_{\mathrm{B,A}}\mathbf{T}\right)^\top \qquad \text{or} \qquad {}_{\alpha,\beta}(\mathbf{T}^\intercal) = \left[T^\intercal{}_i{}^j\right] = \left[T^j{}_i\right] .} \tag{D.52}$$

The reader should note the difference of the transpose of a matrix denoted by ${}_{\mathrm{A,B}}\mathbf{T}^\top$ and the representation of the transpose map $\mathbf{T}^\intercal$ as ${}_{\alpha,\beta}\mathbf{T}^\intercal$. Relation (D.52) gives the true meaning of taking the transpose of a matrix: if a matrix represents an operator on some finite-dimensional space, then its tranpose represents the dual operator. According to appendix D.4, the matrix representation ${}_{\alpha,\beta}\mathbf{T}^\intercal$ maps a coordinate tuple ${}_\beta\mathbf{g}$ in basis β to a coordinate tuple ${}_\alpha\mathbf{f}$ in basis α.

D.9.4 Coordinate Transformation in the Primal and Dual

If we take the indentiy map $\mathbb{1}_V : V \to V$ with $\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathbb{1}_V) = \mathbf{A}_{\mathrm{BA}} = [A^i{}_j]$, then the matrix representation of the transpose operation $\mathbf{mat}_{\alpha\leftarrow\beta}(\mathbb{1}_V^\intercal) = \mathbf{A}_{\alpha\beta}$ of the transpose map $\mathbb{1}_V^\intercal : V^* \to V^*$ results in the map ${\mathbf{A}_{\mathrm{BA}}}^\top$ which transforms from basis $\bar{\bar{\boldsymbol{\epsilon}}}^\beta = \{\boldsymbol{\beta}_i\}$ to the basis $\bar{\bar{\boldsymbol{\epsilon}}}^\alpha = \{\boldsymbol{\alpha}_i\}$ and vice versa with its inverse. The transformation relationships between the dual and the primal space can be summarized in the following:

primal space V	dual space V^*	
${}_\mathrm{B}\mathbf{x} = \mathbf{A}_{\mathrm{BA}}\ {}_\mathrm{A}\mathbf{x} \quad \Leftrightarrow \quad {}_\mathrm{B}y^i = {A_{\mathrm{BA}}}^i{}_j\ {}_\mathrm{A}x^j$ ${}_\mathrm{A}\mathbf{x} = \mathbf{A}_{\mathrm{AB}}\ {}_\mathrm{B}\mathbf{x} \quad \Leftrightarrow \quad {}_\mathrm{A}x^i = {A_{\mathrm{AB}}}^i{}_j\ {}_\mathrm{B}x^j$	${}_\beta\mathbf{f} = {(\mathbf{A}_{\mathrm{AB}})}^\top{}_\alpha\mathbf{f} \quad \Leftrightarrow \quad {}_\beta f_i = {}_\alpha f_j\ {A_{\mathrm{AB}}}^j{}_i$ ${}_\alpha\mathbf{f} = {(\mathbf{A}_{\mathrm{BA}})}^\top{}_\beta\mathbf{f} \quad \Leftrightarrow \quad {}_\alpha f_i = {}_\beta f_j\ {A_{\mathrm{BA}}}^j{}_i$	tuples
$\mathbf{a}_i = \mathbf{b}_j\ {A_{\mathrm{BA}}}^j{}_i$ $\mathbf{b}_i = \mathbf{a}_j\ {A_{\mathrm{AB}}}^j{}_i$	$\boldsymbol{\alpha}^i = \boldsymbol{\beta}^j\ {A_{\mathrm{AB}}}^i{}_j$ $\boldsymbol{\beta}^i = \boldsymbol{\alpha}^j\ {A_{\mathrm{BA}}}^i{}_j$	bases

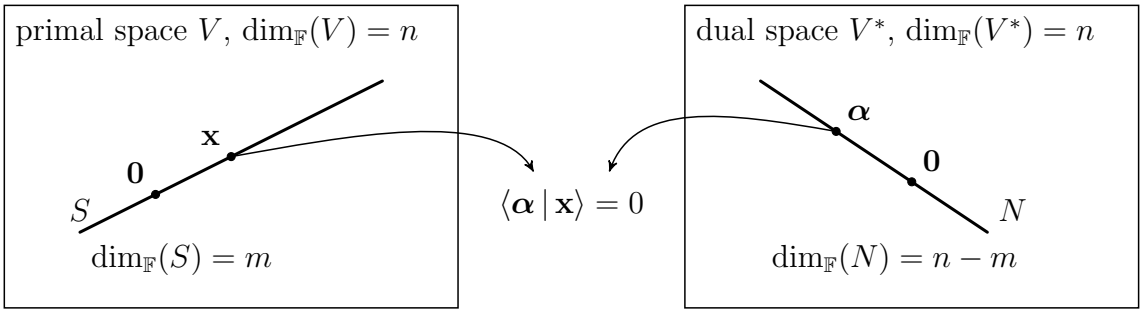
In multilinear algebra, the transformation in the dual space is often called *covariant basis transformation*. The reason for this terminology comes from the fact that the primal basis transforms as $\mathbf{b}_i = \mathbf{a}_j\ {A_{\mathrm{AB}}}^j{}_i$ whereas the coordinate tuple transforms as ${}_\beta\mathbf{f} = {(\mathbf{A}_{\mathrm{AB}})}^\top\ {}_\alpha\mathbf{f}$. On the contrary, the primal tuple transforms with the inverse of $\mathbf{A}_{\mathrm{AB}}$ as ${}_\mathrm{B}\mathbf{x} = {\mathbf{A}_{\mathrm{AB}}}^{-1}\ {}_\mathrm{A}\mathbf{x}$, which is called *contravariant basis transformation*. If the coordinate tuple is written as a row vector, the transpose vanishes and the row tuple transforms with $\mathbf{A}_{\mathrm{AB}}$ which is identical to the transformation of the bases. Therefore, it makes sense to write dual vectors in their coordinate representation as row tuples.

D.9.5 Annihilator

The *annihilator* N of a subspace $S \subseteq V$ is defined as:

$$N \coloneqq \{\boldsymbol{\alpha} \mid \langle \boldsymbol{\alpha} \,|\, \mathbf{x} \rangle = 0 \quad \forall \mathbf{x} \in S\}\ . \tag{D.53}$$

The annihilator is depicted in the following diagram:



If the subspace N has dimension $\dim_\mathbb{F}(S) = m < n$ with $\dim_\mathbb{F}(V) = n$, then the annihilator has dimension $\dim_\mathbb{F}(N) = n - m$.

Proof:
Assume a basis $\bar{\bar{\mathbf{e}}}^{\mathrm{A}} = \{\mathbf{a}_i\}$ for V with $\dim_{\mathbb{F}}(V) = n$. Assume that the subspace S is spanned by the first m basis vectors $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ of V. The dual basis of V for V^* is given as $\bar{\bar{\boldsymbol{\epsilon}}}^{\alpha} = \{\boldsymbol{\alpha}_i\}$.

Every vector $\mathbf{x}$ in S can be written as $\mathbf{x} = x^i \mathbf{a}_i$ with $i \in [1\,;\, m]$. A functional $\mathbf{f} \in V$ can be written in V^* as $\mathbf{f} = f_i \boldsymbol{\alpha}^i$ with $i \in [1\,;\, n]$. For $\mathbf{f}$ to be in the annihilator N, we need $\langle \mathbf{f} \,|\, \mathbf{x} \rangle = 0 \;\forall \mathbf{x} \in S$. Which implies that $f_i x^i = 0, \quad \forall x^i$ with $i \in [1\,;\, m]$. This means that the first m coordinate components f_i are zero which implies further that any $\mathbf{f} \in N$ can be expressed by $\mathbf{f} = f_i \boldsymbol{\alpha}^i, i \in [m+1\,;\, n]$. Which means that the basis vectors $\boldsymbol{\alpha}^i$ with $i \in [m+1\,;\, n]$ span the annihilator and therefore $\dim(N) = n - m$. ■

D.10 Used Notation

$x \in \mathbb{F}$	a scalar in the field $\mathbb{F}$.
$(V, \mathbb{F})$ or $(V, \mathbb{F}, \oplus, \odot)$	vector space or linear space V to the field $\mathbb{F}$ with addition $\oplus$ and multiplication $\odot$.
$\mathcal{T}$, $\mathcal{T}^{\intercal}$, $\mathcal{T}^{-1}$, $\mathcal{A}$,	general maps.
$\mathcal{T} : V \to W$	a map from vector space V to W.
$\mathcal{T}^{-1} : W \to V$	inverse map of $\mathcal{T}$ from vector space W to V.
$\mathcal{T}^{\intercal} : W^* \to V^*$	transpose map of $\mathcal{T}$ from vector space W^* to V^*.
$\mathbb{1}_V : V \to V$ and $\mathbb{1}_W : W \to W$	linear identity map in vector space V and W.
$\mathbf{x} \in V$	vector in the vector space V.
$\bar{\bar{\mathbf{e}}}^{\mathrm{A}} \coloneqq \{\mathbf{a}_i\} \in V$	set of basis vectors for the vector space V.
${}_{\mathrm{A}}\mathbf{x} \in \mathbb{F}^n$	coordinate tuple of a vector $\mathbf{x} \in V$ in the basis A.
$\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathcal{T})$ or ${}_{\mathrm{B,A}}\mathbf{T}$ or $[T^i{}_j] \in \mathbb{F}^{n\times m}$	matrix representation of the linear map $\mathcal{T}$ from basis A to basis B.
$\mathbf{mat}_{\alpha\leftarrow\beta}(\mathcal{T}^{\intercal})$ or ${}_{\alpha,\beta}\mathbf{T}^{\intercal}$ or $[T^{\intercal}{}_i{}^j] \in \mathbb{F}^{n\times m}$	matrix representation of the map $\mathcal{T}^{\intercal}$ from basis β to basis α
$\mathbf{mat}_{\mathrm{A}\leftarrow\mathrm{B}}(\mathcal{T}^{-1})$ or ${}_{\mathrm{A,B}}\mathbf{T}^{-1}$ or $[T^{-1\,i}{}_j] \in \mathbb{F}^{n\times m}$	matrix representation of the linear map $\mathcal{T}^{-1}$ from basis B to basis A
$\mathbf{mat}_{\mathrm{B}\leftarrow\mathrm{A}}(\mathbb{1}_V)$ or ${}_{\mathrm{B,A}}\mathbf{1}$ or $\mathbf{A}_{\mathrm{BA}}$ or $[A^i{}_j] \in \mathbb{F}^{n\times n}$	coordinate transformation from basis A to basis B.
$(\mathbf{A}_{\mathrm{BA}})^{\top}$ or $\mathbf{A}_{\mathrm{BA}}{}^{\top}$	the transpose of the matrix $\mathbf{A}_{\mathrm{BA}}$
$(\mathbf{A}_{\mathrm{BA}})^{-1}$ or $\mathbf{A}_{\mathrm{BA}}{}^{-1}$	the inverse of the matrix $\mathbf{A}_{\mathrm{BA}}$

Appendix E

Additional References

E.1 Source Code References

The following table helps the reader to locate the relevant source code of the GRS framework [141] for all discussed software implementations throughout this thesis. The following path abbreviations are used:

- [CS] : Source code directory of the common source code files of the GRS framework.
- [MPI] : Source code directory of the MPI application GRSFSimMPI.
- [CV] : Source code directory of the converter application GRSFConverter.
- [A] : Source code directory of the library ApproxMVBB [139].

nr.	abbreviation ▶ C++ class/namespace	file path
[1]	SimulationManager	[SIM]/include/GRSF/states/simulationManager/SimulationManager.hpp
[2]	SimulationManagerMPI	[MPI]/include/GRSF/states/simulationManager/SimulationManagerMPI.hpp
[3]	TimeStepper ▶TimeStepperBase	[CS]/include/GRSF/dynamics/general/TimeStepperBase.hpp
[4]	MoreauTimeStepperMPI	[MPI]/include/GRSF/dynamics/general/MoreauTimeStepperMPI.hpp
[5]	DynamicsSystem	[CS]/include/GRSF/dynamics/general/DynamicsSystem.hpp
[6]	StateRecorder	[CS]/include/GRSF/dynamics/buffers/StateRecorder.hpp
[7]	StateRecorderMPI	[MPI]/include/GRSF/dynamics/buffers/StateRecorderMPI.hpp

[8]	RigidBody	[CS]/include/GRSF/dynamics/general/RigidBody.hpp
[9]	CollisionSolver	[CS]/include/GRSF/dynamics/collision/CollisionSolver.hpp
[10]	CollisionSolverMPI	[MPI]/include/GRSF/dynamics/collision/CollisionSolverMPI.hpp
[11]	InclusionSolver	[CS]/include/GRSF/dynamics/Inclusion/InclusionSolver.hpp
[12]	InclusionSolverMPI	[MPI]/include/GRSF/dynamics/Inclusion/InclusionSolverMPI.hpp
[13]	ContactGraph	[CS]/include/GRSF/dynamics/inclusion/ContactGraph.hpp
[14]	ContactGraphMPI	[MPI]/include/GRSF/dynamics/inclusion/ContactGraphMPI.hpp
[15]	SceneParser	[CS]/include/GRSF/systems/SceneParser.hpp
[16]	MultiBodySimFile	[CS]/include/GRSF/dynamics/general/MultiBodySimFile.hpp
[17]	ProcessTopology	[CS]/include/GRSF/Dynamics/General/MPIToplogy.hpp
[18]	GridTopology ▶ProcessTopologyGrid	[MPI]/include/GRSF/dynamics/general/MPITopologyGrid.hpp
[19]	GridTopologyBuilder	[MPI]/include/GRSF/dynamics/general/MPITopologyBuilder.hpp
[20]	KdTreeTopology ▶ProcessTopologyKdTree	[MPI]/include/GRSF/Dynamics/General/MPIToplogyKdTree.hpp
[21]	KdTreeTopologyBuilder	[MPI]/include/GRSF/dynamics/general/MPITopologyBuilder.hpp
[22]	Collider ▶ColliderBody	[CS]/include/GRSF/dynamics/collision/Collider.hpp
[23]	ColliderOOBB	[CS]/include/GRSF/dynamics/collision/Collider.hpp
[24]	ColliderAABB	[CS]/include/GRSF/dynamics/collision/Collider.hpp
[25]	TopologyBuilder	[MPI]/include/GRSF/Dynamics/General/MPIToplogyBuilder.hpp
[26]	NearestNeighbourFilter	[A]/include/ApproxMVBB/KdTree.hpp
[27]	KdTree ▶Tree	[A]/include/ApproxMVBB/KdTree.hpp
[28]	BodyCommunicator	[MPI]/include/GRSF/dynamics/general/BodyCommunicator.hpp

[29]	InclusionCommunicator	[MPI]/include/GRSF/dynamics/inclusion/InclusionCommunicator.hpp
[30]	ProcessCommunicator	[MPI]/include/GRSF/dynamics/general/MPICommunication.hpp
[31]	MPIMessages	[MPI]/include/GRSF/dynamics/general/MPIMessages.hpp
[32]	NMessageWrapperBodies ▶NeighbourMessage WrapperBodies	[MPI]/include/GRSF/dynamics/general/MPIMessages.hpp
[33]	TBMessageWrapperBodies ▶TopologyBuilderMessage WrapperBodies	[MPI]/include/GRSF/dynamics/general/MPIMessages.hpp
[34]	TBMessageWrapperResults ▶TopologyBuilderMessage WrapperResults	[MPI]/include/GRSF/dynamics/general/MPIMessages.hpp
[35]	ContactGraphVisitors ▶*	[CS]/include/GRSF/dynamics/inclusion/ContactGraphVisitors.hpp
[36]	LogicNode	[CS]/include/GRSF/logic/LogicNode.hpp
[37]	LogicSocketTypes	[CV]/include/GRSF/logic/LogicTypes.hpp
[38]	LogicSocket	[CS]/include/GRSF/logic/LogicSocket.hpp
[39]	ExecutionTree ▶ExecutionTreeInOut	[CS]/include/GRSF/logic/ExecutionTreeInOut.hpp
[40]	ColorList ▶ColorList	[CV]/include/GRSF/converters/renderer/RenderExecutionGraphNodes.hpp
[41]	BxdfMaterial ▶BxdfDisneyMaterial	[CV]/include/GRSF/converters/renderer/RenderExecutionGraphNodes.hpp

E.2 File References

The following table lists some relevant files discussed throughout this thesis. The following path abbreviations are used:

- [GRSF] : The root directory of the GRS framework.

nr.	file path
[1]	[GRSF]/simulations/examples/jobs/simulationStudies/avalanche1M-Tree-SimStudy/analyzeStartJob/analyzerLogic/FindStart.xml
[2]	[GRSF]/simulations/examples/jobs/simulationStudies/avalanche1M-Tree-SimStudy/simulationJobs/config/SceneFile.xml

[3] [GRSF]/simulations/examples/jobs/correlationImageVelocimetry/experiments-3Passes/scripts/exp5/3Passes-4FramesFine/accivParametersPass{1,2,3}.ascii

Bibliography

[1] Abadi, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: http://tensorflow.org/ [cit. on p. 134]

[2] Acary, V. "The Siconos software. A open-source software platform for the modeling, the simulation and the control of nonsmooth mechanical and electrical systems." In: *Proc. of the Second International Conference on Advanced COmputational Methods in ENgineering - ACOMEN*. Liège, Belgium, May 2008. URL: https://hal.inria.fr/inria-00423535 [cit. on pp. 5, 81]

[3] Acary, V. and Brogliato, B. "Numerical Methods for the Frictional Contact Problem." In: *Numerical Methods for Nonsmooth Dynamical Systems*. Vol. 35. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2008, pp. 403–440. DOI: 10.1007/978-3-540-75392-6_13 [cit. on pp. 3, 73]

[4] Acary, V. and Jean, M. "Numerical simulation of monuments by the contact dynamics method." In: *Proc. of the Workshop on seismic perfomance of monuments - Monument-98*. Ed. by DGEMN-LNEC-JRC. Laboratório Nacional de engenharia Civil (LNEC). Lisbon, Portugal, Nov. 1998, pp. 69–78. URL: https://hal.inria.fr/inria-00425359 [cit. on p. 3]

[5] Aeberhard, U. P. "Geometrische Behandlung idealer Stösse." PhD thesis. Institute for Mechanical Systems, ETH Zurich, Switzerland, 2008. DOI: 10.3929/ethz-a-005684478 [cit. on p. 127]

[6] Alart, P. and Curnier, A. "A Mixed Formulation for Frictional Contact Problems Prone to Newton Like Solution Methods." In: *Computer Methods in Applied Mechanics and Engineering* **92**(3), 1991, pp. 353–375. DOI: 10.1016/0045-7825(91)90022-X [cit. on pp. 53, 54]

[7] Alart, P., Iceta, D., and Dureisseix, D. "A nonlinear Domain Decomposition formulation with application to granular dynamics." In: *Computer Methods in Applied Mechanics and Engineering* **205-208**, 2012. Special Issue on Advances in Computational Methods in Contact Mechanicsdedicated to the memory of Professor J.A.C. Martins, pp. 59–67. DOI: 10.1016/j.cma.2011.04.024 [cit. on pp. 130, 166]

[8] Allard, J. et al. "SOFA - an Open Source Framework for Medical Simulation." In: *Proc. of the Medicine Meets Virtual Reality - MMVR 15*. Vol. 125. Studies in Health Technology and Informatics. IOP Press, Palm Beach, USA, Feb. 2007, pp. 13–18. URL: https://hal.inria.fr/inria-00319416 [cit. on p. 4]

[9] Altmann, S. L. *Rotations, Quaternions and Double Groups*. Dover Publications, Mineola, NY, USA, 2005 [cit. on p. 13]

[10] AOS Technology AG. *S-PRI F1 Product Leaflet*. June 2012. URL: http://www.aostechnologies.com/fileadmin/user_upload/PDFs/Highspeed/S-PRI_F1_ProductLeaflet_en.pdf [cit. on p. 150]

[11] Arnold, P. and Glocker, Ch. "Bobsleigh Optimization: A Customized Dynamic Vibration Absorber with Limit Stops." In: *Proc. of the 7th European Nonlinear Dynamics Conference (ENOC 2011)*. Ed. by Bernardini, D., Rega, G., and Romeo, F. 7th European Nonlinear Dynamics Conference, Rome, Italy, July 2011 [cit. on p. 3]

[12] Asay-Davis, X. S., Marcus, P. S., Wong, M. H., and Pater, I. de. "Jupiter's shrinking Great Red Spot and steady Oval BA: Velocity measurements with the 'Advection Corrected Correlation Image Velocimetry' automated cloud-tracking method." In: *Icarus* **203**(1), 2009, pp. 164–188. DOI: 10.1016/j.icarus.2009.05.001 [cit. on pp. 151, 152, 154]

[13] Asay-Davis, X. S. and Wong, M. H. *ACCIV:: Advection-Corrected Correlation Image Velocimetry*. 2009. URL: https://github.com/xylar/acciv [cit. on p. 151]

[14] Auslender, A. and Teboulle, M. "Interior Gradient and Proximal Methods for Convex and Conic Optimization." In: *SIAM Journal on Optimization* **16**(3), 2006, pp. 697–725. DOI: 10.1137/S1052623403427823 [cit. on p. 73]

[15] Azéma, E., Radjai, F., Peyroux, R., and Saussine, G. "Force transmission in a packing of pentagonal particles." In: *Phys. Rev. E* **76**, Iss. 1, 2007. DOI: 10.1103/PhysRevE.76.011301 [cit. on p. 3]

[16] Azéma, E. et al. "Discrete simulation of dense flows of polyhedral grains down a rough inclined plane." In: *Phys. Rev. E* **86**(3), 2012, pp. 1–14. DOI: 10.1103/PhysRevE.86.031303 [cit. on p. 3]

[17] Baasch, T. "Computing Large-Scale Non-Smooth Dynamics on the GPU." MA thesis. Institute for Mechanical Systems, ETH Zurich, Switzerland, 2014. DOI: 10.3929/ethz-a-010585362 [cit. on pp. 4, 77, 81]

[18] Baraff, D. "Analytical Methods for Dynamic Simulation of Non-penetrating Rigid Bodies." In: *Proc. of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '89. ACM, New York, NY, USA, 1989, pp. 223–232. DOI: 10.1145/74333.74356 [cit. on p. 4]

[19] Barequet, G. and Har-Peled, S. "Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions." In: *Journal of Algorithms* **38**(1), 2001, pp. 91–109. DOI: 10.1006/jagm.2000.1127 [cit. on p. 108]

[20] Bauchau, O. A. and Laulusa, A. "Review of Contemporary Approaches for Constraint Enforcement in Multibody Systems." In: *Journal of Computational and Nonlinear Dynamics* **3**(1), 2007. DOI: 10.1115/1.2803258 [cit. on p. 68]

[21] Beck, A. and Teboulle, M. "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems." In: *SIAM Journal on Imaging Sciences* **2**(1), 2009, pp. 183–202. DOI: 10.1137/080716542 [cit. on p. 73]

[22] Bender, J., Erleben, K., and Trinkle, J. "Interactive Simulation of Rigid Body Dynamics in Computer Graphics." In: *Computer Graphics Forum* **33**(1), 2014, pp. 246–270. doi: 10.1111/cgf.12272 [cit. on p. 4]

[23] Bertsekas, D. "On the Goldstein-Levitin-Polyak gradient projection method." In: *1974 IEEE Conference on Decision and Control including the 13th Symposium on Adaptive Processes*. Phoenix, AZ, USA, Nov. 1974, pp. 47–52. doi: 10.1109/CDC.1974.270399 [cit. on p. 43]

[24] Blanco, J. L., Muja, M., and Lowe, D. G. *The nanoflann library: A C++ library for nearest neighbor search with kd-trees and a subset of the FLANN library*. url: https://github.com/jlblancoc/nanoflann [cit. on p. 109]

[25] Boman, E. G., Catalyurek, U. V., Chevalier, C., and Devine, K. D. "The Zoltan and Isorropia Parallel Toolkits for Combinatorial Scientific Computing: Partitioning, Ordering, and Coloring." In: *Scientific Programming* **20**(2), 2012, pp. 129–150. doi: 10.1155/2012/713587 [cit. on pp. 4, 102]

[26] Boost Development Community. *The Boost C++ Library*. 2015. url: www.boost.org [cit. on pp. 114, 134]

[27] Borwein, J. M. and Corless, R. M. "The Encyclopedia of Integer Sequences (N. J. A. Sloane and Simon Plouffe)." In: *SIAM Review* **38**(2), 1996, pp. 333–337. doi: 10.1137/1038058 [cit. on p. 91]

[28] Boyd, S. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2010 [cit. on p. 40]

[29] Bratberg, I., Radjai, F., and Hansen, A. "Dynamic rearrangements and packing regimes in randomly deposited two-dimensional granular beds." In: *Phys. Rev. E* **66**, Iss. 3, 2002. doi: 10.1103/PhysRevE.66.031303 [cit. on p. 3]

[30] Brogliato, B. *Nonsmooth Mechanics: Models, Dynamics and Control*. 3rd ed. Springer International Publishing, Switzerland, 2016. doi: 10.1007/978-3-319-28664-8 [cit. on p. 3]

[31] Bugnion, L. "Measurements of velocity, velocity fluctuations and density in dry granular flows." PhD thesis. ETH Zurich, Switzerland, 2012. doi: 10.3929/ethz-a-009770799 [cit. on pp. 159, 168]

[32] Burke, J. V., Ferris, M. C., and Qian, M. "On the Clarke subdifferential of the distance function of a closed set." In: *Journal of Mathematical Analysis and Applications* **166**(1), 1992, pp. 199–213. doi: 10.1016/0022-247X(92)90336-C [cit. on p. 37]

[33] Cadoux, F. "An optimization-based algorithm for Coulomb frictional contact." In: *CANUM 2008 - 39e Congrès National d'Analyse Numérique*. Ed. by Besse, C., Goubet, O., Goudon, T., and Nicaise, S. Vol. 27. EDP Science, Saint Jean de Monts, France, May 2008, pp. 54–69. doi: 10.1051/proc/2009019 [cit. on p. 60]

[34] Carisch, L. "Stopping Behavior of Dry and Wet Granular Avalanches." MA thesis. Institute of Fluid Dynamics, ETH Zurich, Switzerland, 2013 [cit. on pp. 144, 149, 150]

[35] Chang, C.-T., Gorissen, B., and Melchior, S. "Fast Oriented Bounding Box Optimization on the Rotation Group $SO(3, \mathbb{R})$." In: *ACM Transactions on Graphics (TOG)* **30**(5), 2011, 122:1–122:16. DOI: 10.1145/2019627.2019641 [cit. on p. 107]

[36] Coplien, J. O. "Curiously Recurring Template Patterns." In: *C++ Report* **7**(2), 1995, pp. 24–27. URL: http://dl.acm.org/citation.cfm?id=229227.229229 [cit. on p. 134]

[37] Corbalan, J., Duran, A., and Labarta, J. "Dynamic Load Balancing of MPI+OpenMP Applications." In: *Proc. of the 2004 International Conference on Parallel Processing*. ICPP '04. IEEE Computer Society, Washington, DC, USA, 2004, pp. 195–202. DOI: 10.1109/ICPP.2004.29 [cit. on p. 102]

[38] Cormen, T. H., Stein, C., Rivest, R. L., and Leiserson, C. E. *Introduction to Algorithms*. 3rd ed. MIT Press, Cambridge, MA, USA, 2009 [cit. on p. 96]

[39] Coumans, E. et al. *Bullet Physics Engine*. URL: http://bulletphysics.org [cit. on pp. 4, 81]

[40] Courtecuisse, H. and Allard, J. "Parallel Dense Gauss-Seidel Algorithm on Many-Core Processors." In: *Proc. of the 2009 11th IEEE International Conference on High Performance Computing and Communications*. IEEE Computer Society, Washington, DC, USA, 2009, pp. 139–147. DOI: 10.1109/HPCC.2009.51 [cit. on pp. 4, 111]

[41] Cundall, P. A. and Strack, O. D. L. "A discrete numerical model for granular assemblies." In: *Géotechnique* **29**(1), 1979, pp. 47–65. DOI: 10.1680/geot.1979.29.1.47 [cit. on p. 2]

[42] Cundall, P. A. "A computer model for simulating progressive large scale movements of blocky rock systems." In: *Proc. of the Symposium of the International Society of Rock Mechanics*. Vol. 1. Nancy, France, 1971, pp. 132–150 [cit. on p. 2]

[43] De Gennes, P.-G. "From Rice to Snow." In: *Nishina Memorial Lectures*. Vol. 746. Lecture Notes in Physics. Springer, Japan, 2008, pp. 297–318. DOI: 10.1007/978-4-431-77056-5_14 [cit. on p. 1]

[44] Delenne, J.-Y., Soulié, F., Youssoufi, M. S. E., and Radjai, F. "From liquid to solid bonding in cohesive granular media." In: *Mechanics of Materials* **43**(10), 2011, pp. 529–537. DOI: 10.1016/j.mechmat.2011.06.008 [cit. on pp. 3, 169]

[45] Derouet-Jourdan, A., Bertails-Descoubes, F., Daviet, G., and Thollot, J. "Inverse dynamic hair modeling with frictional contact." In: *ACM Transactions on Graphics (TOG)* **32**(6), 2013, p. 159. DOI: 10.1145/2508363.2508398 [cit. on p. 4]

[46] Dimitrov, D. *Geometric Applications of Principal Component Analysis*. Suedwestdeutscher Verlag fuer Hochschulschriften, Saarbrücken, Germany, 2012 [cit. on p. 106]

[47] Dreisigmeyer, D. *Direct Search Algorithms Over Riemannian Manifolds*. Tech. rep. LA-UR-06-7416. Los Alamos National Laboratory, Los Alamos, NM, USA, 2007 [cit. on p. 107]

[48] Dubois, F. and Jean, M. "The non smooth contact dynamic method: recent LMGC90 software developments and application." In: *Analysis and Simulation of Contact Problems*. Ed. by Wriggers, P. and Nackenhorst, U. Vol. 27. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2006, pp. 375–378. doi: 10.1007/3-540-31761-9_44 [cit. on p. 5]

[49] Eckart, W., Gray, J. M. N. T., and Hutter, K. "Particle Image Velocimetry (PIV) for Granular Avalanches on Inclined Planes." In: *Dynamic Response of Granular and Porous Materials under Large and Catastrophic Deformations*. Ed. by Hutter, K. and Kirchner, N. Vol. 11. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2003, pp. 195–218. doi: 10.1007/978-3-540-36565-5_6 [cit. on p. 153]

[50] Ericson, C. *Real-Time Collision Detection*. Ed. by Ericson, C. CRC Press, Boca Raton, Florida, USA, 2004 [cit. on p. 92]

[51] Eugster, S. R. *Geometric Continuum Mechanics and Induced Beam Theories*. Vol. 75. Lecture Notes in Applied and Computational Mechanics. Springer International Publishing, Cham, Switzerland, 2015. doi: 10.1007/978-3-319-16495-3 [cit. on pp. 9, 11, 25, 26, 28]

[52] Fares, Ch. and Hamam, Y. "Collision detection for rigid bodies: A state of the art review." In: *Proc. of International Conference Graphicon 2005*. Novosibirsk Akademgorodok, Russia, 2005. url: http://graphicon.ru [cit. on p. 4]

[53] Fischer, K. et al. "Bounding Volumes." In: *CGAL User and Reference Manual*. 4.6.2. CGAL Editorial Board, 2015. url: http://www.cgal.org/ [cit. on p. 109]

[54] G. Saussine et al. "Modelling ballast behaviour under dynamic loading. Part 1: A 2D polygonal discrete element method approach." In: *Computer Methods in Applied Mechanics and Engineering* **195**(19–22), 2006, pp. 2841–2859. doi: 10.1016/j.cma.2005.07.006 [cit. on p. 3]

[55] Gallier, J. H. *Geometric Methods and Applications: For Computer Science and Engineering*. 2nd ed. Vol. 38. Texts in Applied Mathematics. Springer, New York, NY, USA, 2011. doi: 10.1007/978-1-4419-9961-0 [cit. on p. 13]

[56] Gao, D., Zhang, Y., and Zhao, Y. "The Application of kd-tree in Astronomy." In: *ASP Conf. Ser.* **394**, 2008, p. 525. arXiv: 0801.2004 [cit. on p. 94]

[57] García, X. and Medina, E. "Acoustic response of cemented granular sedimentary rocks: Molecular dynamics modeling." In: *Phys. Rev. E* **75**, Iss. 6, 2007. doi: 10.1103/PhysRevE.75.061308 [cit. on p. 2]

[58] Gehring, C. et al. "An Evaluation of Moreau's time-stepping scheme for the simulation of a legged robot." In: *10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*. Vol. 6. Proc. of the ASME IDETC/CIE 2014. Buffalo, New York, NY, 2014. doi: 10.1115/DETC2014-34374 [cit. on p. 3]

[59] Gilabert, F. A., Roux, J.-N., and Castellanos, A. "Computer simulation of model cohesive powders: Influence of assembling procedure and contact laws on low consolidation states." In: *Phys. Rev. E* **75**, Iss. 1, 2007. doi: 10.1103/PhysRevE.75.011303 [cit. on pp. 2, 169]

[60] Glocker, Ch. *Set-Valued Force Laws: Dynamics of Non-Smooth Systems*. 1st ed. Vol. 1. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2001. doi: 10.1007/978-3-540-44479-4 [cit. on pp. 21, 22, 49, 52, 57, 58, 191]

[61] Glocker, Ch. "An Introduction to Impacts." In: *Nonsmooth Mechanics of Solids*. CISM International Centre for Mechanical Sciences. Springer, Vienna, Austria, 2006, pp. 45–101. doi: 10.1007/978-3-211-48243-8_2 [cit. on pp. 57, 61, 64, 172]

[62] Glocker, Ch. "Energetic consistency conditions for standard impacts: Part I." In: *Multibody System Dynamics* **29**(1), 2013, pp. 77–117. doi: 10.1007/s11044-012-9316-9 [cit. on p. 65]

[63] Glocker, Ch. "Energetic consistency conditions for standard impacts: Part II." In: *Multibody System Dynamics* **32**(4), 2014, pp. 445–509. doi: 10.1007/s11044-013-9387-2 [cit. on p. 65]

[64] Goddard, J. D. "Continuum Modeling of Granular Media." In: *Applied Mechanics Reviews* **66**(5), 2014. doi: 10.1115/1.4026242 [cit. on p. 4]

[65] Goldstein, A. A. "Convex programming in Hilbert space." In: *Bull. Amer. Math. Soc.* **70**(5), 1964, pp. 709–710. doi: 10.1090/S0002-9904-1964-11178-2 [cit. on p. 43]

[66] González-Vélez, H. and Leyton, M. "A survey of algorithmic skeleton frameworks: high-level structured parallel programming enablers." In: *Software: Practice and Experience* **40**(12), 2010, pp. 1135–1160. doi: 10.1002/spe.1026 [cit. on p. 113]

[67] Gropp, W., Lusk, E., and Skjellum, A. *Using MPI : Portable Parallel Programming with the Message-Passing Interface*. 3rd ed. MIT Press, Cambridge, MA, USA, 2014 [cit. on p. 112]

[68] Guennebaud, G., Jacob, B., et al. *Eigen v3*. 2010. url: http://eigen.tuxfamily.org [cit. on p. 110]

[69] Hamilton, W. R. "XXII. On quaternions; or on a new system of imaginaries in algebra." In: *Philosophical Magazine Series 3* **29**(192), 1846, pp. 113–122. doi: 10.1080/14786444608645590 [cit. on p. 13]

[70] Hapala, M. and Havran, V. "Review: Kd-tree Traversal Algorithms for Ray Tracing." In: *Computer Graphics Forum* **30**(1), 2011, pp. 199–213. doi: 10.1111/j.1467-8659.2010.01844.x [cit. on p. 94]

[71] Heimsch, T. "Variational Integration of Hamiltonian Systems by Discontinuous Approximations." Diss. ETH No. 22952. 2015 [cit. on p. 67]

[72] Herrmann, H. J. "Computer simulations of granular media." In: *Disorder and Granular Media*. Ed. by D., B. and A., H. Elsevier Sci. Publ., Amsterdam, 1993 [cit. on p. 2]

[73] Iglberger, K. "Software Design of a Massively Parallel Rigid Body Framework." PhD thesis. Friedrich-Alexander-University, Erlangen, Germany, 2010 [cit. on p. 4]

[74] Iglberger, K. and Rüde, U. *The pe Rigid Multibody Physics Engine*. Tech. rep. Friedrich-Alexander-University, Erlangen, Germany, 2009. URL: https://www10.cs.fau.de/publications/reports/TechRep09-9.pdf [cit. on pp. 4, 81, 115, 118]

[75] Iglberger, K. and Rüde, U. "Large-scale rigid body simulations." In: *Multibody System Dynamics* **25**, Iss. 1, 2011, pp. 81–95. DOI: 10.1007/s11044-010-9212-0 [cit. on pp. 4, 115, 116, 118, 130, 166]

[76] Jean, M. "Frictional contact in collections of rigid or deformable bodies: numerical simulation of geomaterial motions." In: *Mechanics of Geomaterial Interfaces*. Ed. by Selvadurai, A. and Boulon, M. Vol. 42. Studies in Applied Mechanics. Elsevier, 1995, pp. 463–486. DOI: 10.1016/S0922-5382(06)80022-X [cit. on p. 2]

[77] Jean, M., Jourdan, F., and Tathi, B. "Numerical Dynamics for the Simulation of Deep-Drawing." In: *Proc. of IDDRG 1994 - 18th Biennial Congress*. Lisboa, Portugal, May 1994, pp. 425–435 [cit. on p. 2]

[78] Jean, M. and Moreau, J.-J. "Unilaterality and dry friction in the dynamics of rigid body collections." In: *Proc. of Contact Mechanics International Symposium*. Presses Polytechniques et Universitaires Romandes, Lausanne, Switzerland, 1992, pp. 31–48 [cit. on pp. 2, 61]

[79] Jean, M. and Pratt, E. "A system of rigid bodies with dry friction." In: *International Journal of Engineering Science* **23**(5), 1985, pp. 497–513. DOI: 10.1016/0020-7225(85)90060-6 [cit. on p. 2]

[80] Jean, M. "Unilateral contact with dry friction: time and space discrete variables formulation." In: *Archives of Mechanics* **40**(5-6), 1988, pp. 677–691 [cit. on p. 2]

[81] Jean, M. "The non-smooth contact dynamics method." In: *Computer Methods in Applied Mechanics and Engineering* **177**(3–4), 1999, pp. 235–257. DOI: 10.1016/S0045-7825(98)00383-1 [cit. on p. 2]

[82] Kapoulkine, A. *pugixml: Light-weight, simple and fast XML parser for C++ with XPath support*. 2006. URL: http://pugixml.org [cit. on p. 110]

[83] Kaufman, D. M., Edmunds, T., and Pai, D. K. "Fast frictional dynamics for rigid bodies." In: *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2005* **24**(3), 2005, pp. 946–956. DOI: 10.1145/1186822.1073295 [cit. on p. 4]

[84] Kendall, W., Nath, D., Bland, W., et al. *MPI Tutorial*. 2015. URL: https://github.com/wesleykendall/mpitutorial.git [cit. on p. 112]

[85] Koziara, T. and Bićanić, N. "A distributed memory parallel multibody Contact Dynamics code." In: *International Journal for Numerical Methods in Engineering* **87**(1-5), 2011, pp. 437–456. DOI: 10.1002/nme.3158 [cit. on pp. 4, 81]

[86] Krause, R. H. "Monotone Multigrid Methods for Signorini's Problem with Friction." PhD thesis. Freie Universität Berlin, FB Mathematik und Informatik, Berlin, Germany [cit. on p. 51]

[87] Kuipers, J. B. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*. Princeton University Press, Princeton, NJ, USA, 2002 [cit. on p. 13]

[88] Kumar, K., Soga, K., and Delenne, J.-Y. "Multi-scale modelling of granular avalanches." In: *AIP Conf. Proc.* Vol. 1542. July 2013, pp. 1250–1253. doi: 10.1063/1.4812165 [cit. on p. 3]

[89] Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. "Convergence Properties of the Nelder–Mead Simplex Method in Low Dimensions." In: *SIAM Journal on Optimization* **9**(1), 1998, pp. 112–147. doi: 10.1137/S1052623496303470 [cit. on p. 107]

[90] Lanier, J. and Jean, M. "Experiments and numerical simulations with 2D disks assembly." In: *Powder Technology* **109**(1–3), 2000, pp. 206–221. doi: 10.1016/S0032-5910(99)00237-5 [cit. on p. 3]

[91] Larsson, T., Akenine-Müller, T., and Lengyel, E. "On Faster Sphere-Box Overlap Testing." In: *Journal of Graphics, GPU, and Game Tools* **12**(1), 2007, pp. 3–8. doi: 10.1080/2151237X.2007.10129232 [cit. on p. 92]

[92] Lee, J. D., Sun, Y., and Saunders, M. A. "Proximal Newton-type methods for minimizing composite functions." Version 13. In: *SIAM Journal on Optimization* **24**(3), 2013, pp. 1420–1443. doi: 10.1137/130921428 [cit. on p. 45]

[93] Leine, R. I. "On the stability of motion in non-smooth mechanical systems." Habilitation thesis. Institute for Mechanical Systems, ETH Zurich, Switzerland, June 2006 [cit. on p. 49]

[94] Leine, R. I. and Glocker, Ch. "A set-valued force law for spatial Coulomb–Contensou friction." In: *European Journal of Mechanics – A/Solids* **22**(2), 2003, pp. 193–216. doi: 10.1016/S0997-7538(03)00025-1 [cit. on p. 53]

[95] Leine, R. I., Le Saux, C., and Glocker, Ch. "Friction models for the rolling disk." In: *Proc. of the ENOC 2005 Conference*. Eindhoven, CD-ROM. Aug. 2005 [cit. on p. 53]

[96] Leine, R. I. and van de Wouw, N. *Stability and Convergence of Mechanical Systems with Unilateral Constraints*. Vol. 36. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2008. doi: 10.1007/978-3-540-76975-0 [cit. on pp. 57, 62]

[97] Levitin, E. and Polyak, B. "Constrained minimization methods." In: *USSR Computational Mathematics and Mathematical Physics* **6**(5), 1966, pp. 1–50. doi: 10.1016/0041-5553(66)90114-5 [cit. on p. 43]

[98] Luding, S. "Collisions & Contacts between Two Particles." In: *Physics of Dry Granular Media*. Ed. by Herrmann, H., Hovi, J.-P., and Luding, S. Vol. 350. NATO ASI Series. Springer, Netherlands, 1998, pp. 285–304. doi: 10.1007/978-94-017-2653-5_20 [cit. on p. 2]

[99] Malandain, G. and Boissonnat, J.-D. "Computing the Diameter of a Point Set." In: *International Journal of Computational Geometry & Applications* **12**(6), 2002, pp. 489–510. doi: 10.1007/3-540-45986-3_18 [cit. on p. 108]

[100] Mattson, T., Sanders, B., and Massingill, B. *Patterns for Parallel Programming*. 1st ed. Addison-Wesley Professional, Boston, USA, 2004 [cit. on p. 113]

[101] Matuttis, H., Luding, S., and Herrmann, H. "Discrete element simulations of dense packings and heaps made of spherical and non-spherical particles." In: *Powder Technology* **109**(1–3), 2000, pp. 278–292. doi: 10.1016/S0032-5910(99)00243-0 [cit. on p. 2]

[102] Mazhar, H., Heyn, T., Negrut, D., and Tasora, A. "Using Nesterov's Method to Accelerate Multibody Dynamics with Friction and Contact." In: *ACM Transactions on Graphics (TOG)* **34**(3), 2015, 32:1–32:14. doi: 10.1145/2735627 [cit. on pp. 5, 73, 78, 167]

[103] Mazhar, H. et al. "CHRONO: a parallel multi-physics library for rigid-body, flexible-body, and fluid dynamics." In: *Mechanical Sciences* **4**(1), 2013, pp. 49–64. doi: 10.5194/ms-4-49-2013 [cit. on pp. 4, 81]

[104] McNamara, S. and Herrmann, H. "Measurement of indeterminacy in packings of perfectly rigid disks." In: *Phys. Rev. E* **70**, Iss. 6, 2004. doi: 10.1103/PhysRevE.70.061303 [cit. on pp. 2, 3]

[105] Merkus, H. G. *Particle Size Measurements: Fundamentals, Practice, Quality*. 1st ed. Vol. 17. Particle Technology Series. Springer, Netherlands, 2009. doi: 10.1007/978-1-4020-9016-5 [cit. on p. 139]

[106] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard, Version 3.1*. High Performance Computing Center Stuttgart (HLRS), Stuttgart, Germany, Sept. 2012. url: http://www.mpi-forum.org [cit. on pp. 111, 112]

[107] Meurant, G. "A Review on the Inverse of Symmetric Tridiagonal and Block Tridiagonal Matrices." In: *SIAM Journal on Matrix Analysis and Applications* **13**(3), 1992, pp. 707–728. doi: 10.1137/0613045. eprint: http://dx.doi.org/10.1137/0613045 [cit. on p. 126]

[108] Möller, M. "Consistent Integrators for Non-Smooth Dynamical System." PhD thesis. Institute for Mechanical Systems, ETH Zurich, Switzerland, 2011. doi: 10.3929/ethz-a-006716243 [cit. on pp. 5, 13, 24, 29, 33, 42, 43, 53, 68, 69, 73, 75, 165]

[109] Möller, M. and Glocker, Ch. "Analogous Non-Smooth Models of Mechanical and Electrical Systems." In: *IUTAM Symposium on Multiscale Problems in Multibody System Contacts*. Ed. by Eberhard, P. Vol. 1. IUTAM Bookseries. Springer, Netherlands, 2007, pp. 45–54. doi: 10.1007/978-1-4020-5981-0_5 [cit. on p. 5]

[110] Möller, M. and Glocker, Ch. "Non-smooth modelling of electrical systems using the flux approach." In: *Nonlinear Dynamics* **50**(1), 2007, pp. 273–295. doi: 10.1007/s11071-006-9157-2 [cit. on p. 5]

[111] Moore, A. W. *Efficient memory-based learning for robot control*. Tech. rep. UCAM-CL-TR-209. University of Cambridge, Computer Laboratory, Nov. 1990. URL: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-209.pdf [cit. on p. 94]

[112] Moreau, J.-J. "Evolution problem associated with a moving convex set in a Hilbert space." In: *Journal of Differential Equations* **26**(3), 1977, pp. 347–374. DOI: 10.1016/0022-0396(77)90085-7 [cit. on p. 2]

[113] Moreau, J.-J. "Liaisons unilatérales sans frottement et chocs inélastiques." In: *Comptes-rendus des séances de l'Académie des sciences. Série 2, Mécanique-physique, chimie, sciences de l'univers, sciences de la terre* **296**(19), 1983, pp. 1473–1476 [cit. on p. 2]

[114] Moreau, J.-J. "New computation methods in granular dynamics." In: *Powders & Grains 93: Proc. of the 2nd International Conference on Micromechanics of Granular Media*. Ed. by Thornton, C. Birmingham, UK, 1993, p. 227 [cit. on p. 2]

[115] Moreau, J.-J. "Some numerical methods in multibody dynamics: Application to granular materials." In: *European Journal of Mechanics A/Solids* **13**, 1994, pp. 93–194 [cit. on p. 2]

[116] Moreau, J.-J. "Numerical Investigation of Shear Zones in Granular Materials." In: *Friction, Arching and Contact Dynamics*. Ed. by Wolf, D. E. and Grassberger, P. World Scientific, HLRZ, Forschungszentrum Julich, Germany, 1997, pp. 233–247 [cit. on p. 3]

[117] Moreau, J.-J. "Numerical aspects of the sweeping process." In: *Computer Methods in Applied Mechanics and Engineering* **177**(3-4), 1999, pp. 329–349. DOI: 10.1016/S0045-7825(98)00387-9 [cit. on p. 68]

[118] Moreau, J.-J. "The Stress Tensor in Granular Media and in other Mechanical Collections." In: ed. by Cambou, B., Jean, M., and Radjai, F. Micromechanics of Granular Materials. ISTE, 2010, pp. 51–100. DOI: 10.1002/9780470611616.ch2 [cit. on p. 168]

[119] Moreau, J.-J. "Bounded variation in time." In: *Topics in Nonsmooth Mechanics*. Ed. by Moreau, J. J., Panagiotopoulos, P. D., and Strang, G. Birkhäuser, Basel, 1988, pp. 1–74. DOI: 10.1002/zamm.19890691204 [cit. on pp. 2, 61]

[120] Moreau, J.-J. "Nonsmooth Mechanics and Applications." In: ed. by Moreau, J. J. and Panagiotopoulos, P. D. Springer, Vienna, Austria, 1988. Chap. Unilateral Contact and Dry Friction in Finite Freedom Dynamics, pp. 1–82. DOI: 10.1007/978-3-7091-2624-0_1 [cit. on pp. 61, 64, 68, 69]

[121] Mount, D. M. and Arya, S. *ANN: Library for Approximate Nearest Neighbor Searching*. College Park, MD 20742 USA. URL: http://www.cs.umd.edu/~mount/ANN/ [cit. on p. 109]

[122] Muja, M. and Lowe, D. G. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration." In: *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340. URL: https://github.com/mariusmuja/flann [cit. on p. 109]

[123] Muja, M. and Lowe, D. G. "Fast Matching of Binary Features." In: *Proc. of 9th Conference on Computer and Robot Vision (CRV)*. 2012, pp. 404–410. DOI: 10.1109/CRV.2012.60 [cit. on p. 109]

[124] Muja, M. and Lowe, D. G. "Scalable Nearest Neighbor Algorithms for High Dimensional Data." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36**(11), 2014, pp. 2227–2240. DOI: 10.1109/TPAMI.2014.2321376 [cit. on p. 109]

[125] Nagy, B. "Distance with generalized neighbourhood sequences in nD and ∞D." In: *Discrete Applied Mathematics* **156**(12), 2008, pp. 2344–2351. DOI: 10.1016/j.dam.2007.10.017 [cit. on p. 90]

[126] Nagy, Z. et al. "Modeling the Motion of Microrobots on Surfaces Using Non-smooth Multibody Dynamics." In: *IEEE Transactions on Robotics* **28**(5), 2012, pp. 1058–1068. DOI: 10.1109/TRO.2012.2199010 [cit. on p. 3]

[127] Negrut, D., Serban, R., Mazhar, H., and Heyn, T. "Parallel Computing in Multibody System Dynamics: Why, When, and How." In: *Journal of Computational and Nonlinear Dynamics* **9**(4), 2014. DOI: 10.1115/1.4027313 [cit. on p. 4]

[128] Nelder, J. A. and Mead, R. "A Simplex Method for Function Minimization." In: *The Computer Journal* **7**(4), 1965, pp. 308–313. DOI: 10.1093/comjnl/7.4.308 [cit. on p. 107]

[129] Nesterov, Y. "Smooth minimization of non-smooth functions." In: *Mathematical programming* **103**(1), 2005, pp. 127–152. DOI: 10.1007/s10107-004-0552-5 [cit. on p. 73]

[130] Nesterov, Y. "Gradient methods for minimizing composite functions." In: *Mathematical Programming* **140**(1), 2012, pp. 125–161. DOI: 10.1007/s10107-012-0629-5 [cit. on p. 73]

[131] Nesterov, Y. "A method of solving a convex programming problem with convergence rate O(1/sqr(k))." In: *Soviet Mathematics Doklady* **27**, 1983, pp. 372–376 [cit. on p. 73]

[132] Nesterov, Y. "On an approach to the construction of optimal methods of minimization of smooth convex functions." In: *Ekonomika i Mateaticheskie Metody* **24**(3), 1988, pp. 509–517 [cit. on p. 73]

[133] Nezamabadi, S., Radjai, F., Averseng, J., and Delenne, J.-Y. "Implicit frictional-contact model for soft particle systems." In: *Journal of the Mechanics and Physics of Solids* **83**, 2015, pp. 72–87. DOI: 10.1016/j.jmps.2015.06.007 [cit. on p. 3]

[134] Nguyen, N. S. and Brogliato, B. *Multiple Impacts in Dissipative Granular Chains*. Vol. 72. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2014. DOI: 10.1007/978-3-642-39298-6 [cit. on p. 65]

[135] Nica, B. "The Mazur–Ulam theorem." In: *Expositiones Mathematicae* **30**(4), 2012, pp. 397–398. DOI: 10.1016/j.exmath.2012.08.010 [cit. on p. 10]

[136] Niebler, E. *Meta: A tiny metaprogramming library*. 2015. URL: https://github.com/ericniebler/meta [cit. on p. 110]

[137] Nineb, S., Alart, P., and Dureisseix, D. "Approche multiéchelle des systèmes de tenségrité." In: *European Journal of Computational Mechanics* **15**(1-3), 2006, pp. 319–328. DOI: 10.3166/remn.15.319-328. eprint: http://www.tandfonline.com/doi/pdf/10.3166/remn.15.319-328 [cit. on p. 3]

[138] Nouguier-Lehon, C., Cambou, B., and Vincens, E. "Influence of particle shape and angularity on the behaviour of granular materials: a numerical analysis." In: *International Journal for Numerical and Analytical Methods in Geomechanics* **27**(14), 2003, pp. 1207–1226. DOI: 10.1002/nag.314 [cit. on p. 3]

[139] Nützi, G. *ApproxMVBB: Fast algorithms for minimal volume bounding box computations*. 2015. URL: www.github.com/gabyx/ApproxMVBB [cit. on pp. 6, 80, 106–108, 201]

[140] Nützi, G. *Job Configurator for High-Performance Cluster Computing*. URL: www.github.com/gabyx/HPCJobConfigurator [cit. on p. 153]

[141] Nützi, G. *GRSF: Granular Rigid Body Simulation Framework*. 2016. URL: www.github.com/gabyx/GRSFramework [cit. on pp. 6, 80, 81, 201]

[142] Nützi, G., Schweizer, A., Möller, M., and Glocker, C. "Projective Jacobi and Gauss Seidel on the GPU for Non Smooth Multibody Systems." In: *10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*. Vol. 6. Proc. of the ASME IDETC/CIE 2014. Buffalo, New York, NY, 2014. DOI: 10.1115/DETC2014-34606 [cit. on pp. 4, 73, 75, 81, 111]

[143] O'Rourke, J. "Finding minimal enclosing boxes." In: *International Journal of Computer & Information Sciences* **14**(3), 1985, pp. 183–199. DOI: 10.1007/BF00991005 [cit. on p. 105]

[144] O'Rourke, J. *Computational Geometry in C*. 2nd ed. Cambridge University Press, New York, NY, USA, 1998 [cit. on p. 109]

[145] Pacheco, P. S. *Parallel Programming with MPI*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 2009 [cit. on p. 112]

[146] Pixar Animation Studio. *The RiSpec 3.2*. Nov. 2005. URL: http://renderman.pixar.com/view/rispec/ [cit. on p. 133]

[147] Potters Europe. *EC-Certificate of Conformity*. Jan. 2008. URL: www.potterseurope.com [cit. on pp. 143, 144, 149]

[148] Pudasaini, S. P. and Hutter, K. *Avalanche dynamics: dynamics of rapid flows of dense granular avalanches*. Springer, Berlin, Germany, 2007. DOI: 10.1007/978-3-540-32687-8 [cit. on p. 3]

[149] Radjai, F., Brendel, L., and Roux, S. "Nonsmoothness, indeterminacy, and friction in two-dimensional arrays of rigid particles." In: *Phys. Rev. E* **54**, Iss. 1, 1996, pp. 861–873. DOI: 10.1103/PhysRevE.54.861 [cit. on p. 3]

[150] Radjai, F. and Dubois, F., eds. *Discrete-element Modeling of Granular Materials*. Wiley-ISTE, London, UK, 2011 [cit. on p. 5]

[151] Radjai, F., Evesque, P., Bideau, D., and Roux, S. "Stick-slip dynamics of a one-dimensional array of particles." In: *Phys. Rev. E* **52**, Iss. 5, 1995, pp. 5555–5564. doi: 10.1103/PhysRevE.52.5555 [cit. on p. 3]

[152] Radjai, F. and Roux, S. "Turbulentlike Fluctuations in Quasistatic Flow of Granular Media." In: *Phys. Rev. Lett.* **89**(6), Iss. 6, 2002. doi: 10.1103/PhysRevLett.89.064302 [cit. on p. 3]

[153] Radjai, F., Wolf, D. E., Jean, M., and Moreau, J.-J. "Bimodal Character of Stress Transmission in Granular Packings." In: *Phys. Rev. Lett.* **80**, Iss. 1, 1998, pp. 61–64. doi: 10.1103/PhysRevLett.80.61 [cit. on p. 3]

[154] Radjai, F. et al. "Force networks in dense granular media." In: *Powders & Grains 97: Proc. of the 3rd International Conference on Powders & Grains*. Ed. by Behringer, R. P. and Jenkins, J. T. Balkema, Rotterdam, 1997, pp. 211–214 [cit. on p. 3]

[155] Rempfler, G. S. and Glocker, Ch. "A bobsleigh simulator software." In: *Multibody System Dynamics*, 2015, pp. 1–22. doi: 10.1007/s11044-015-9450-2 [cit. on pp. 3, 4]

[156] Renouf, M. and Fillot, N. "Coupling electrical and mechanical effects in discrete element simulations." In: *International Journal for Numerical Methods in Engineering* **74**(2), 2008, pp. 238–254. doi: 10.1002/nme.2157 [cit. on p. 5]

[157] Renouf, M. and Alart, P. "Conjugate gradient type algorithms for frictional multi-contact problems: applications to granular materials." In: *Computer Methods in Applied Mechanics and Engineering* **194**(18–20), 2005, pp. 2019–2041. doi: 10.1016/j.cma.2004.07.009 [cit. on p. 73]

[158] Renouf, M., Dubois, F., and Alart, P. "A parallel version of the non smooth contact dynamics algorithm applied to the simulation of granular media." In: *Journal of Computational and Applied Mathematics* **168**(1–2), 2004, pp. 375–382. doi: 10.1016/j.cam.2003.05.019 [cit. on p. 4]

[159] Richefeu, V., Radjai, F., and El Youssoufi, M. S. "Stress transmission in wet granular materials." In: *The European Physical Journal E* **21**(4), 2006, pp. 359–369. doi: 10.1140/epje/i2006-10077-1 [cit. on p. 2]

[160] Ries, A., Wolf, D. E., and Unger, T. "Shear zones in granular media: Three-dimensional contact dynamics simulation." In: *Phys. Rev. E* **76**(5), Iss. 5, 2007. doi: 10.1103/PhysRevE.76.051301 [cit. on p. 3]

[161] Rockafellar, R. T. *Convex Analysis*. 1st ed. Princeton Landmarks in Mathematics and Physics. Princeton University Press, Princeton, NJ, USA, 1997 [cit. on pp. 37, 39, 40, 55]

[162] Roux, S., Hild, F., and Berthaud, Y. "Correlation Image Velocimetry: A Spectral Approach." In: *Applied optics* **41**, 2002, pp. 108–115. url: https://hal.archives-ouvertes.fr/hal-00002900 [cit. on p. 151]

[163] Rusu, R. B. and Cousins, S. "3D is here: Point Cloud Library (PCL)." In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011. DOI: 10.1109/ICRA.2011.5980567 [cit. on p. 109]

[164] Saint-Cyr, B., Radjai, F., Delenne, J.-Y., and Sornay, P. "Cohesive granular materials composed of nonconvex particles." In: *Phys. Rev. E* **87**(5), Iss. 5, 2013. DOI: 10.1103/PhysRevE.87.052207 [cit. on pp. 3, 169]

[165] Saint-Cyr, B. et al. "Effect of Particle Shape non-Convexity on the Rheology of Granular Media : 3D Contact Dynamics Simulations." In: *II International Conference on Particle-based Methods - Fundamentals and Applications*. Spain, 2011, pp. 1–7. URL: https://hal.archives-ouvertes.fr/hal-00686453 [cit. on p. 3]

[166] Saxcé, G. D. and Feng, Z.-Q. "The bipotential method: A constructive approach to design the complete contact law with friction and improved numerical algorithms." In: *Mathematical and Computer Modelling* **28**(4–8), 1998. Recent Advances in Contact Mechanics, pp. 225–245. DOI: 10.1016/S0895-7177(98)00119-8 [cit. on p. 55]

[167] Schäling, B. *The Boost C++ Libraries*. 2011. URL: http://theboostcpplibraries.com [cit. on p. 114]

[168] Schauer, J. and Nüchter, A. "Collision detection between point clouds using an efficient k-d tree implementation." In: *Advanced Engineering Informatics* **29**(3), 2015, pp. 440–458. DOI: 10.1016/j.aei.2015.03.007 [cit. on p. 94]

[169] Schindler, T. et al. "Analysing dynamical phenomenons: Introduction to MBSim." In: *Proc. of the 1st Joint International Conference on Multibody System Dynamics*. 2010. URL: www.github.com/mbsim-env [cit. on p. 5]

[170] Schweizer, A. P. "Ein nichtglattes mechanisches Modell für Steinschlag." PhD thesis. Institute for Mechanical Systems, ETH Zurich, Switzerland, 2014. DOI: 10.3929/ethz-a-010464319 [cit. on pp. 3, 9, 11, 13, 29, 68, 73, 75, 165]

[171] Seiler, Ch. *Cross Platform Floating Point Arithmetics*. URL: http://christian-seiler.de/projekte/fpmath/ [cit. on p. 109]

[172] Servin, M., Wang, D., Lacoursière, C., and Bodin, K. "Examining the smooth and nonsmooth discrete element approaches to granular matter." In: *International Journal for Numerical Methods in Engineering* **97**(12), 2014, pp. 878–902. DOI: 10.1002/nme.4612 [cit. on pp. 3, 168]

[173] Shewchuk, J. R. "Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates." In: *Discrete & Computational Geometry* **18**(3), 1997, pp. 305–363 [cit. on p. 109]

[174] Shojaaee, Z. et al. "An adaptive hierarchical domain decomposition method for parallel contact dynamics simulations of granular materials." In: *Journal of Computational Physics* **231**(2), 2012, pp. 612–628. DOI: 10.1016/j.jcp.2011.09.024 [cit. on pp. 5, 101, 130, 131]

[175] Silbert, L. E., Grest, G. S., and Landry, J. W. "Statistics of the contact network in frictional and frictionless granular packings." In: *Phys. Rev. E* **66**(6), Iss. 6, 2002. doi: 10.1103/PhysRevE.66.061303 [cit. on p. 3]

[176] Smith, R. *Open Dynamics Engine*. 2013. url: http://www.ode.org/ [cit. on p. 4]

[177] Staron, L., Vilotte, J.-P., and Radjai, F. "Preavalanche Instabilities in a Granular Pile." In: *Phys. Rev. Lett.* **89**(20), Iss. 20, 2002. doi: 10.1103/PhysRevLett.89.204302 [cit. on p. 3]

[178] Stephenson, I. D. *Essential RenderMan®*. 2nd ed. Springer, London, UK, 2007. doi: 10.1007/978-1-84628-800-5 [cit. on p. 133]

[179] Stiess, M. *Mechanische Verfahrenstechnik - Partikeltechnologie. 1*. 3rd ed. Springer, Berlin, Germany, 2009. doi: 10.1007/978/3-540-32552-9 [cit. on pp. 139, 142, 143]

[180] Studer, C. and Glocker, C. "Representation of Normal Cone Inclusion Problems in Dynamics Via Non-linear Equations." In: *Archive of Applied Mechanics* **76**(5–6), Iss. 5, 2006, pp. 327–348. doi: 10.1007/s00419-006-0031-y [cit. on p. 73]

[181] Studer, Ch. *Numerics of Unilateral Contacts and Friction. Modeling and Numerical Time Integration in Non-Smooth Dynamics*. Vol. 47. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2009. doi: 10.1007/978-3-642-01100-9 [cit. on pp. 3, 5, 73, 75]

[182] Taboada, A., Chang, K.-J., Radjaï, F., and Bouchette, F. "Rheology, force transmission, and shear instabilities in frictional granular media from biaxial numerical tests using the contact dynamics method." In: *Journal of Geophysical Research: Solid Earth* **110**(B9), 2005. doi: 10.1029/2003JB002955 [cit. on p. 3]

[183] Tasora, A., Negrut, D., and Anitescu, M. "Large-scale parallel multi-body dynamics with frictional contact on the graphical processing unit." In: *Proc. of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*. Vol. 222. 4. SAGE Publications, 2008, pp. 315–326. doi: 10.1243/14644193JMBD154 [cit. on pp. 4, 77, 111]

[184] Terze, Z., Mueller, A., and Zlatar, D. "Redundancy-Free Integration of Rotational Quaternions in Minimal Form." In: Vol. 6. 10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control. Buffalo, New York, USA. Buffalo, NY, USA, 2014. doi: 10.1115/DETC2014-35118 [cit. on p. 24]

[185] Terze, Z. and Naudet, J. "Geometric properties of projective constraint violation stabilization method for generally constrained multibody systems on manifolds." In: *Multibody System Dynamics* **20**(1), 2008, pp. 85–106. doi: 10.1007/s11044-008-9107-5 [cit. on p. 68]

[186] The CGAL Project. *CGAL User and Reference Manual*. 4.8. CGAL Editorial Board, 2016. url: http://doc.cgal.org/4.8/Manual/packages.html [cit. on p. 109]

[187] Thornton, C. "Computer simulation of impact fracture/fragmentation." In: *The first Nisshin Engineering Particle Technology International Seminar: Discrete Particle Simulations in Powder Technology*. Osaka, Japan, 1993, p. 17 [cit. on p. 2]

[188] Thornton, C. "Coefficient of restitution for collinear collisions of elastic-perfectly plastic spheres." In: *Journal of Applied Mechanics* **64**(2), 1997, pp. 383–386. doi: 10.1115/1.2787319 [cit. on p. 2]

[189] Thornton, C. and Yin, K. "Impact of elastic spheres with and without adhesion." In: *Powder Technology* **65**(1), 1991, pp. 153–166. doi: 10.1016/0032-5910(91)80178-L [cit. on p. 2]

[190] Thorsten Pöschel and Volkhard Buchholtz. "Molecular Dynamics of Arbitrarily Shaped Granular Particles." In: *J. Phys. I France* **5**(11), 1995, pp. 1431–1455. doi: 10.1051/jp1:1995208 [cit. on p. 2]

[191] Tokumaru, P. and Dimotakis, P. "Image correlation velocimetry." In: *Experiments in Fluids* **19**(1), 1995, pp. 1–15. doi: 10.1007/BF00192228 [cit. on p. 151]

[192] Tonge, R., Benevolenski, F., and Voroshilov, A. "Mass Splitting for Jitter-free Parallel Rigid Body Simulation." In: *ACM Transactions on Graphics (TOG)* **31**(4), 2012, 105:1–105:8. doi: 10.1145/2185520.2185601 [cit. on pp. 130, 166]

[193] Topin, V. et al. "Micro-rheology of dense particulate flows: Application to immersed avalanches." In: *Journal of Non-Newtonian Fluid Mechanics* **166**(1–2), 2011, pp. 63–72. doi: 10.1016/j.jnnfm.2010.10.006 [cit. on p. 5]

[194] Toussaint, G. "Solving geometric problems with the rotating calipers." In: *Proc. of IEEE MELECON '83*. A10.02. Athens, Greece, 1983, pp. 1–4 [cit. on p. 105]

[195] Tseng, P. "On accelerated proximal gradient methods for convex-concave optimization." In: *submitted to SIAM Journal on Optimization*, 2008 [cit. on p. 73]

[196] Visseq, V. et al. "Dense granular dynamics analysis by a domain decomposition approach." In: *Computational Mechanics* **49**(6), 2012, pp. 709–723. doi: 10.1007/s00466-012-0699-5 [cit. on pp. 130, 166]

[197] Wald, I. and Havran, V. "On building fast kd-Trees for Ray Tracing, and on doing that in O(N log N)." In: *Proc. of the 2006 IEEE Symposium on Interactive Ray Tracing*. Salt Lake City, UT, USA, 2006, pp. 61–69. doi: 10.1109/RT.2006.280216 [cit. on p. 96]

[198] Ward, J. P. *Quaternions and Cayley Numbers*. Vol. 403. Mathematics and Its Applications. Springer, Dordrecht, Netherlands, 1997. doi: 10.1007/978-94-011-5768-1 [cit. on p. 13]

[199] Weber-Jahnke, J. H. and Stier, J. "Virtual prototyping of automated manufacturing systems with Geometry-driven Petri nets." In: *Computer-Aided Design* **41**(12), 2009, pp. 942–951. doi: 10.1016/j.cad.2009.06.012 [cit. on p. 4]

[200] Wriggers, P. and Nackenhorst, U., eds. *Analysis and Simulation of Contact Problems*. Vol. 27. Lecture Notes in Applied and Computational Mechanics. Springer, Berlin, Germany, 2006. doi: 10.1007/3-540-31761-9 [cit. on p. 4]

[201] Wriggers, P. and Panatiotopoulos, P., eds. *New Developments in Contact Problems*. Vol. 384. CISM International Centre for Mechanical Sciences. Springer, Vienna, Austria, 2014. doi: 10.1007/978-3-7091-2496-3 [cit. on p. 169]

[202] Yan, X., Han, D., and Sun, W. "A modified projection method with a new direction for solving variational inequalities." In: *Applied Mathematics and Computation* **211**(1), 2009, pp. 118–129. doi: 10.1016/j.amc.2009.01.064 [cit. on p. 43]

[203] Žic, E., Bićanić, N., Koziara, T., and Ožanić, N. "The numerical modelling of suspended sediment propagation in small torrents with the application of the Contact Dynamics method." In: *Tehnički vjesnik* **21**(5), 2014, pp. 939–952 [cit. on p. 3]

Curriculum Vitae

Gabriel Nützi
born on 31.10.1986
citizen of Wolfwil SO, Switzerland

Education

05/2009 – 10/2011	MSc ETH in Mechanical Engineering, ETH Zurich Major: Robotics, System and Control
06/2009	BSc ETH in Mechanical Engineering, ETH Zurich Major: Mechatronics
July 2005	Higher School Certificate, Gymnasium Oberaargau Major: Physics and applied mathematics

Publications

[1] Nützi, G., Schweizer, A., Möller, M., and Glocker, C. "Projective Jacobi and Gauss Seidel on the GPU for Non Smooth Multibody Systems." In: *10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*. Vol. 6. Proc. of the ASME IDETC/CIE 2014. Buffalo, New York, NY, 2014. doi: 10.1115/DETC2014-34606

[2] Gehring, C. et al. "An Evaluation of Moreau's time-stepping scheme for the simulation of a legged robot." In: *10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*. Vol. 6. Proc. of the ASME IDETC/CIE 2014. Buffalo, New York, NY, 2014. doi: 10.1115/DETC2014-34374

[3] Nutzi, G., Feng, X., Siegwart, R., and Kock, S. "Optimal Contact Force Prediction of Redundant Single and Dual-Arm Manipulators." In: *Journal of the Chinese Society of Mechanical Engineers* **34**(3), 2013, pp. 251–258

[4] Nützi, G., Weiss, S., Scaramuzza, D., and Siegwart, R. "Fusion of IMU and Vision for Absolute Scale Estimation in Monocular SLAM." In: *Journal of Intelligent & Robotic Systems* **61**(1), 2010, pp. 287–299. doi: 10.1007/s10846-010-9490-z

Software

[1] Nützi, G. *GRSF: Granular Rigid Body Simulation Framework*. 2016. URL: www.github.com/gabyx/GRSFramework

[2] Nützi, G. *ApproxMVBB: Fast algorithms for minimal volume bounding box computations*. 2015. URL: www.github.com/gabyx/ApproxMVBB

[3] Nützi, G. *Job Configurator for High-Performance Cluster Computing*. URL: www.github.com/gabyx/HPCJobConfigurator