A Coupled Contact-Mechanics Computational Model for Studying Deformable Human-Artifact Contact

by

Christopher David King

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2018

Author .....
Department of Mechanical Engineering
May 23, 2018

Certified by .....
Leia Stirling
Assistant Professor
Charles Stark Draper Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .....
Raúl Radovitzky
Professor

Certified by .....
Kenneth Kamrin
Thesis Reader, Associate Professor

Accepted by .....
Rohan Abeyaratne
Chairman, Department Committee on Graduate Theses

A Coupled Contact-Mechanics Computational Model for Studying Deformable Human-Artifact Contact

by

Christopher David King

Submitted to the Department of Mechanical Engineering
on May 23, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

Gas-pressurized spacesuits are necessary for human spaceflight, most notably for extravehicular activity (EVA). Legacy EVA suits have been primarily rigid, and operation in such suits can result in significant metabolic expense, or even injury, for the wearer. To reduce these effects, modern spacesuits are more flexible, through the incorporation of more softgood materials and specially designed joint interfaces such as hip bearings. However, modeling the effects of human-suit interaction for these softgood materials is challenging due to the highly deformable nature of the suit coupled with the deformable nature of the human. To enable improved analysis and design of modern spacesuits, a computational model that can resolve the structural deformations of the suit and human resulting from contact interactions is developed.

This thesis details the development and validation of a coupled contact-mechanics solver architecture for use in studying the effects of human-artifact interaction, particularly with respect to pressurized softgood exosuit design. To resolve contact and structural mechanics interactions for a deformable human and artifact, a finite element model is developed. First, the SUMMIT computational framework is employed for resolving the structural deformations of the system, and is coupled to an explicit contact mechanics scheme. The explicit contact scheme is implemented so as to resolve both external- and self-contact problems. Next, the model architecture is integrated to enable parallelization of both the structural deformation and contact systems, and computational scaling investigated. A computational trade study is performed to benchmark the coupled contact-mechanics method against a simpler rigid body model employing a penalty method. Following this, the model is validated against experimental data for various artifact contact problems. The explicit coupled contact-mechanics model is found to effectively capture contact interactions of the experimental data, with improved fidelity for deformable contact interactions. With careful tuning of the system properties, the coupled contact-mechanics model enables an architecture for an integrated human-suit analysis and design model.

Thesis Supervisor: Leia Stirling
Title: Assistant Professor
Charles Stark Draper Professor of Aeronautics and Astronautics

Acknowledgments

I want to thank Leia Stirling for giving me the opportunity to work on this project, and join the Man Vehicle Lab at MIT. I'm incredibly thankful that she chose to reach out to me when I started graduate study at MIT, as this work has not only been exciting and challenging, but her guidance, knowledge, and patience as an advisor is incredible. Even when I was in the weeds of reviving legacy code, or stretched too thin due to deadlines at my job, she was always understanding and able to parse the problems into realizable increments to keep the project on track and help me balance conflicts. I'd also like to thank Conor, Aditi, Patrick and Alvin, from the MVL who worked on this NRI project for all the help, especially in experimental methods, and all the fun car rides out to DCCI.

I also want to thank Raúl Radovitzky for allowing me to work alongside his research team in the ISN using the computational framework SUMMIT, which formed the primary basis for the model. His knowledge of computational structural mechanics, Euler-Lagrangian mechanics, contact methods, and parallel computation schemes are unparalleled, and the model could not have been successful without his guidance. Even though I was not an official member of his research team, he allowed me to be an active member of the software development team, and was always available to help me with any problem, whether as high-level as modeling philosophy, or as low-level as repository management. The rigorous software development methods he instilled in me over the course of this project have been beneficial to me in both school and my career, and I am incredibly thankful for his continued efforts to help me grow. I'd also like to thank the members of the RRGroup which have helped me along the way, especially Ryadh, Biana, Anwar, Mohammad, Brian and Tom, who have helped me sort out either particularly tricky bugs, or my particularly tricky self-made mistakes. I'd also like to apologize to them for remoting into Nimble at any given hour and lighting up the entire lab without warning. I also want to thank my in-department reader, Ken Kamrin, for agreeing to read my thesis for the Mechanical Engineering department.

I'd also like to thank Paul Lopiccolo, Tom Martyn, and Paul Patoulidis for not only allowing me the flexibility to work on my thesis and travel to MIT concurrent with my work obligations during the work week, but for going far above and beyond in helping me with GE thesis procedures. This work has given me experience in project and software development, along with extensive computational modeling methods that will continue to help me throughout my career.

My parents and sister deserve special thanks for always being in my corner and picking up the phone at any random hour to talk, and for helping me balance work and life when I felt overwhelmed. I also want to thank Jessie for going through this whole experience with me, both at MIT and at work, for being an incredible friend, and for helping me stay (mostly) sane. Finally, I want to thank my wife, Kaitlin, for giving me the courage to pursue graduate study at MIT, and for supporting me through all the long nights and crazy schedules. I couldn't have done it without you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The advancement of human spaceflight predicates the need for gas-pressurized spacesuits to protect and enable astronauts in extravehicular activity (EVA) and launch, entry and abort (LEA) scenarios. Activities such as repairing hardware during a spacewalk, or exploration of the lunar surface, have all necessitated the use of suits such as NASA's Extravehicular Mobility Unit (EMU). These legacy EVA suits, such as the EMU, are often comprised of a combination of softgood materials for the limb regions, and rigid materials for the trunk region of the suit. The intent of this construction is aimed at maximizing the performance and efficiency of the human operator, while minimizing the potential for injury [7]. While these suits, such as the Apollo Program A7LB lunar suit, have been effective in enabling the astronaut to complete their objectives in short-term lunar and spacewalk scenarios in the past, astronauts indicated a need for improvement in suit mobility, as inflexibility at the hip and glove caused an increase in human workload, and in some cases resulted in hand injury [8]. The current EMU suit has also been reported to cause a variety of mild injuries in the hands and shoulders [9]. To compound these concerns, future EVA activity is expected to increase in frequency and length by orders of magnitude, going from 20 total EVAs across the entire Apollo Program, up to potentially 76 EVAs in a single mission [7]. Due to this expected increase in EVA, there is a need

to develop modern EVA suits capable of improving astronaut mobility and efficiency, while also reducing the risk for injury, to accommodate such an increased demand for EVA in future missions.

To improve astronaut performance and reduce injury risk during EVA, suits should be designed to maximize mobility and minimize deviations from the unsuited kinematic profiles of the astronaut [10]. The current EMU suit is comprised of a rigid fiberglass trunk, known as the Hard Upper Torso (HUT) assembly, combined with bearing assemblies attached to the softgood material regions along the limbs. While the softgood materials are highly flexible by material properties, when pressurized for operation in space, the effective rigidity of these sections increases. To move the suit, the astronaut must apply work to drive the pressurized suit, needing to overcome not only the hysteresis in the bearing assemblies, but also to drive the reduction in volume at the point of bending in the pressurized softgood region of the suit [11]. This additional work not only requires the astronaut exert more energy in driving motion, but also disturbs the kinematic profile. Modern spacesuit designs such as NASA's Mark III suit and DCCI's Demonstrator suit have been developed with the aim of mitigating the cost associated with driving volume reduction at the point of bending in the pressurized softgood regions through unique bearings [12], or through a series of softgood convolutes [2], respectively. Figure 1-1 illustrates the NASA Mark III and DCCI Demonstrator Suit for reference.

Significant work has been undertaken to understand the nature of how both the legacy A7LB [13, 14], current EMU [15, 16] and modern suits [17] impact the kinematics and of the astronaut. However, how the astronaut interacts with and moves relative to the EVA suit in operation is not yet well understood.

1.2 Objective

Prior studies have attempted to capture astronaut-suit interaction effects through external measurements such as motion capture [18–20], but external measurements can only evaluate the total astronaut-suit system, and are unable to decompose the

Figure 1-1: NASA Mark III EVA Suit [1] (left) and DCCI Demonstrator LEA Suit [2] (right)

individual contributions of the astronaut and suit. Additional work has considered tracking body joint angles during suited operation [14, 21], but without being able to resolve the astronaut-suit contact locations and contact pressure magnitudes, the impact on the astronaut biomechanics cannot be understood. Recent work has focused on the use of in-suit pressure sensing systems to transiently detect and measure human-suit contact interactions during suited operation [22, 23]. Such in-suit measurements provide a means of measuring the effect of human-suit interactions on astronaut biomechanics, and also as a means for guiding future EVA suit design. While such an in-suit sensing system can evaluate a given suit design, a means by which such interactions can be modeled and resolved does not yet exist. Such a coupled human-suit interaction model could be used to inform and guide suit design prior to fabrication.

The focus of this work is aimed at the development of such a multi-tiered computational framework by which the human-suit interactions can be modeled, and used to evaluate operational effects of suit design parameters. The overall project from which this work is motivated has three primary aims:

1) Development of a coupled finite element and musculoskeletal model of the integrated human-suit system.

2) Validation of the integrated human-suit model with component level and human-system experimental data.
3) Evaluation of operational performance sensitivity to changes in material stiffness, muscle activation assumptions, and strategically placed actuation.

This work is aimed at addressing the first aim: the development of a coupled mechanics-musculoskeletal model of an integrated human-suit system. The mechanics portion of this model is of particular note, as it must be capable of resolving contact between the human and the suit, as well as self-contact of the suit, as suit softgood materials fold on themselves during motion, as illustrated in Figure 1-2.

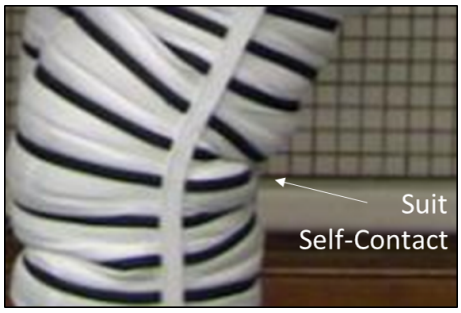


Figure 1-2: DCCI Demonstrator Suit Knee Region experiencing Self-Contact [2]

To satisfy this aim, the objective of this work is to develop, implement, and validate a coupled contact-mechanics model for connection with a human musculoskeletal model. This process involves the selection of an overall contact-mechanics architecture, computational framework, and the development and validation of the low-level contact and structural mechanics models.

1.3 Overall Model Connectivity and Architecture

The first step in the development of a model capable of capturing the contact interactions in an integrated human-suit system is the construction of an overall ar-

chitecture. This high-level architecture needs to be specified to identify the primary inputs and outputs necessary at each stage of the model, and what validation is necessary for each sub-component of the model. From a purely black box perspective, the integrated model needs to be able to take in a geometric representation of the human-suit system, and given some expected motion profile for the human, output the resultant kinematics of the human and suit to predict form and fit, the stresses applied to both human and suit to predict potential for injury or damage to the suit, and joint torques and muscle activations required to drive such a motion to capture musculoskeletal impacts to the human. Given these high level expectations, the interior of this theoretical black box model must be populated with methods capable of providing the necessary outputs. To meet these requirements, the nominal high-level model architecture was identified as shown in Figure 1-3.

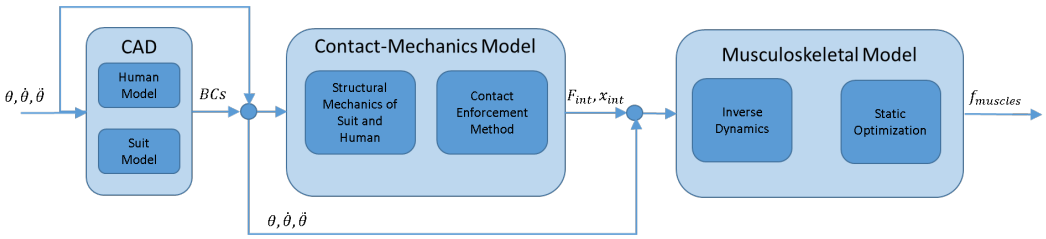


Figure 1-3: High-Level, Preliminary Integrated Human-Suit Model Architecture

A Computer-Aided Design (CAD) model representation of the suit and human, with input motion and posture profiles, represents the first sub-component of the integrated human-suit model. Within the CAD model, geometries representing the suit and human are contained, along with necessary material property definitions and assignments. The CAD model is provided the initial posture for the simulation, and the prescribed motion profile is used to dictate corresponding boundary conditions acting on the geometries. The combined CAD model information for both the human and suit are then provided along with these boundary conditions to the contact-mechanics model. The contact-mechanics model will compute the transient

response of the human and suit to the prescribed motion profiles, while enforcing contact constraints so as to resolve the contact pressures and interactions between the human and suit. The structural mechanics component of this coupled contact-mechanics (CCM) model must resolve and track the internal energy associated with both the suit and human throughout the transient to report internal stress and forces, as well as reporting displacement data associated with both the human and suit models. These internal forces and displacements are translated into effective loads on the joints of the human, which are provided, along with the prescribed motion profiles, to a musculoskeletal model. The musculoskeletal model will use the applied joint torques to compute the forces required by the muscles to achieve the resultant motion using inverse dynamics and static optimization. It should be noted that this architecture predicates an "open-loop" concept with regards to the motion profiles. This open-loop concept implies that the prescribed motion profiles of the human or suit are not stopped or modified during the transient; human neural control will not attempt to modify trajectory based on impedance from the suit. This architecture does not disqualify adding an additional internal feedback loop in the future, but for the purposes of this model to aid in decision support of suit design parameter selection, identification of human effort required to achieve a motion in a suit design, assessing risk of injury, or suit fit checks, this open-loop structure is capable.

While the identification and selection process of methods and models for each of these subcomponents will be detailed further in Chapter 2, for completeness, the overall model architecture with calibration inputs is illustrated in Figure 1-4.

To tune and calibrate the integrated human-suit model, two external calibration and validation input steps are necessary. First, component-level calibrations of the modeling methods, material properties, and representative human anthropometry is integrated with the output from the CCM model to update the CAD models of the suit and human as necessary. For example, to capture the stress-strain relationship of the softgood convolutes of the DCCI Demonstrator Suit, iteration on constitutive material models and properties may be necessary to represent the suit. In addition to these component-level calibrations, which act on the CAD models, human-system

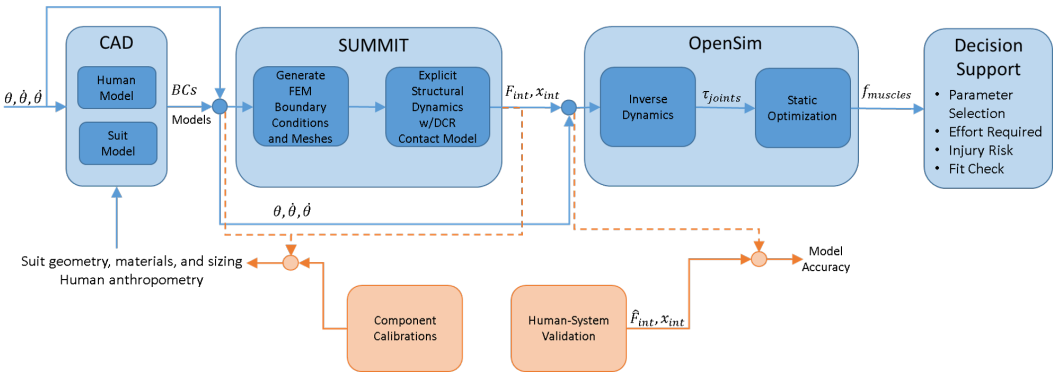


Figure 1-4: High-Level Integrated Human-Suit Model Architecture with selected component methods and calibration inputs

validation is necessary downstream of the CCM model output to validate the accuracy of the CCM model results based on experimental human-suit interation data, such as that collected with in-suit pressure sensing systems [23]. This validation step is necessary to bound the accuracy of the model, and to inform the decision support processes based on the accuracy of the model results. Furthermore, this human-suit validation can be used to guide component-level calibrations, to improve the fidelity of the CCM model and open pathways for datamatching the model to experimental data if desired.

While Figure 1-4 highlights the selections that were made for the CCM model, it is relevant to consider the pedigree of modeling techniques for resolving problems in the class of human-artifact contact problems, as discussed in Section 1.4, as these techniques were evaluated and traded based on the requirements of the model.

The CAD application used in this work was selected to be SolidWorks [24], but this can enable the use of any upstream CAD and pre-processing software as desired. A set of utilities used to easily communicate CAD mesh data, material properties, and boundary conditions was developed to support the connectivity between SolidWorks and the computational framework used for the CCM model. These utilities are detailed in Appendix D for reference. The selection of the computational framework and the internal methods used by the CCM model is detailed in Sections 2.4.2 and 2.5. Finally, the musculoskeletal model used for the integrated human-suit model is the open-source biomechanical modeling software OpenSim [25], which will be detailed

further in Section 1.5.

While the high-level of the integrated human-suit modeling environment has been detailed here, the primary focus of this work was on the development, implementation and validation of the CCM model used to resolve the human-suit contact interactions, illustrated in the Contact-Mechanics Model block of Figure 1-3. Future development of this model, as noted in Section 4.2, will involve further development of the pipeline between the CCM and OpenSim, along with component calibration and human-suit validation testing.

1.4 Human-Artifact Contact Models

While the target application of the Coupled Contact-Mechanics (CCM) model developed in this work is to resolve the interactions between a human operator and EVA space suit, potentially made of rigid or softgood materials, it is relevant to consider the pedigree of human-artifact contact methods employed across a variety of domains when selecting a formulation for the CCM model. Furthermore, to enable the model to scale into geometries and materials of arbitrary definition, the CCM model should be capable of resolving the contact interactions between a human and any arbitrary contact artifact. The terminology of a contact "artifact" is used to represent any abstract entity with which the human may experience a contact interaction, such as vehicles, orthotic devices, or garments. This terminology is adopted from the work of Krüger and Wartzack [26], which considered multi-body contact dynamics between a human operator and an training device. In this work, the concept of the contact artifact is any arbitrary geometric entity which is contacted by the human, or some exogenous "impactor" entity.

When considering the development of the CCM model for use in the overall integrated human-suit model, it was necessary to consider the mathematical formulation in which the CCM model will be represented. Human-artifact contact, and more generally, impactor-artifact contact, is a heavily studied field, and a wide variety of mathematical- and physics-based models have been developed across a wide range

of technical domains such as human training and ergonomics [26], garment design for form and fit [5, 6, 27–31], human-exoskeleton design [32], orthotic device development [33], generalized structural mechanics [34–36], studying neural control [37], and space suit design such as this work [38].

Surveying the formulations used across these domains highlights three primary classes of human-artifact contact formulations: multi-body surrogate models, finite element method (FEM) models, and particle-based models. These classes of contact model formulations will be discussed in detail, including the primary domains that utilize each, the general theory of each formulation, and the benefits and drawbacks of each class relative to one another. These benefits and drawbacks are particularly germane to the selection of a contact model formulation to implement for the CCM model that is the subject of this work, and will be considered further in Section 2.1.

1.4.1 Multi-Body Surrogate Models

The class of human-artifact contact models referred to herein as Multi-Body Surrogate models are those which represent the human and artifact through a construction of multiple simplicial geometric entities such as ellipsoids, coupled with a contact enforcement mechanism to prevent geometric penetration between the impactor and artifact geometric representations. The term surrogate is used as the geometric representations do not resolve unique or low-level geometric details of a human or artifact model, and instead represent the human or artifact with a simplified mathematical or geometric model, as exemplified in Figure 1-5.

This class of contact formulation has been employed primarily in musculoskeletal kinematic studies [26,39–41], with such methods used in the musculoskeletal dynamics toolkits SimTK [26, 42] and DART [43], and is the primary contact formulation class employed by the Todorov:2014 physics engine [4].

As this class of contact formulations represent the human and artifact as geometric entities and do not aim to resolve low-level geometric details or internal energy states, they are computationally inexpensive relative to FEM and Particle methods [28, 40], and as a result are often used in the gaming and animation industries [44]. However,

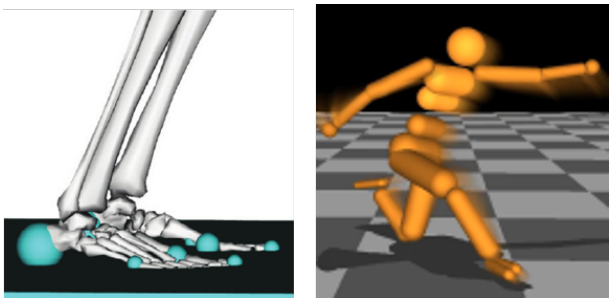


Figure 1-5: Illustration of Multi-Body Surrogate Contact Models for Dynamic Musculoskeletal Models as employed in SimTK [3] (left) and Todorov:2014 [4] (right)

due to the nature of not resolving low-level geometric details and internal energy states at a discretized level for the system, the fidelity of this formulation is limited to the macroscopic dynamic details of the individual geometric entities, and the internal energy information is restricted to the complexity of the surrogate mathematical model employed. Furthermore, while some codes allow the use of arbitrary geometric entities [41] through Non-Uniform Rational Basis Splines (NURBS), most codes use more simplicial geometric entities for resolving contact, such as Hunt and Crossley force spheres [42].

1.4.2 Finite Element Method Models

Finite element method formulations of human-artifact contact problems have been implemented due to their ability to resolve the deformation and stress at varying levels of fidelity through the discretization of the human or artifact into well-defined tetrahedral or quadrilateral elements. Stemming from its basis of representing any entity through discretized elements, FEM has a distinct advantage in that it can describe highly complex geometries for simulation of deformation and vibration associated with dynamics, without restriction to geometric continuity or thickness [45]. From this advantage, FEM contact formulations have been applied to a wide array of human-artifact contact problems, such as the deformation associated with a hand grasping a ball [46], or deformation of a human foot in response to contact with a rigid cleat [33]. The FEM formulation for human-artifact contact events has furthermore been argued as the ideal formulation of the three classes for contact problems where

both the human and artifact can deform for two reasons, as argued by Gourret and Magnenat-Thalmann [46]:

1) FEM formulations permit resolution of contact events with sliding and sticking in addition to repulsive forces. Without a means of tracking the deformations of localized regions of an entity, such as afforded by FEM, the shape of the human or artifact would not be representative to capture these modified shape changes during a contact event.
2) FEM formulations permit capturing deformations in both the human and artifact models at localized regions, where a simplicial or rigid body approximation may neglect such deformations on the human or artifact, depending on their relative rigidity. Even surrogate models with detailed NURBS representations are limited in their ability to resolve localized deformations, as the contact enforcement strategy would require the discretization of the NURBS geometry to resolve localized events - essentially driving towards a FEM formulation.

However, while FEM formulations provide a high degree of fidelity in modeling structural and contact mechanics for arbitrary, complex geometries, and can account for deformation of both human and artifact, they come with the drawback of a significantly higher computational expense [28] than the other classes. Despite this computational expense, applications of FEM contact formulations have been successfully applied in capturing the effects of contact interactions between a human and flexible garment, as illustrated in Figure 1-6, which presents some similar challenges in modeling softgood human-suit interactions [5, 31].

1.4.3 Particle-Based Models

The third class of human-artifact contact formulations are those referred to as Particle-based methods. These methods represent the contact artifact as a collection of discrete particles, often with some form of a mathematical constitutive equation between the particles to maintain continuity between the particles [6, 47], as illustrated in Fig-

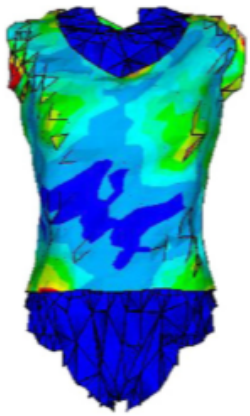


Figure 1-6: Illustration of FEM Human-Artifact Contact Problem resolving pressure distributions across a donned cotton shirt using Mindlin-Reissner Shell elements, per Wang et. al. [5]

ure 1-7.

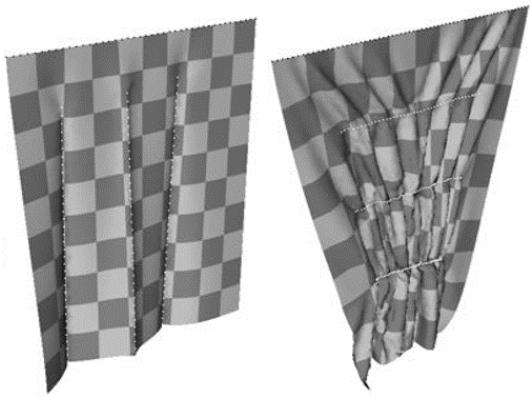


Figure 1-7: Illustration of Particle-based method as adapted from the work of Volino and Magnenat-Thallman [6], representing anisotropic bending stiffness in a flexible cloth garment, while capturing the rest posture curvature of the garment in self-contact

This formulation shares some striking similarities with shell-based FEM approaches, as Particle-based schemes cannot realistically resolve the internal energy of non-thin contact artifacts by virtue of the "surface cloud" of particles, with the exception of computationally expensive peridynamic schemes [47]. However, unlike shell-based

FEM models, Particle methods do not necessarily require physics-driven constitutive relationships between the particle clouds that represent the contact artifact, and can offer significant advantages in computational speed as a result [48]. As a result of this thin surface contact artifact requirement, Particle-based methods have seen success in the garment prototyping industry, and an array of virtual "try-on" environments have been developed leveraging such methods for problems such as drape prediction and fit checks [27, 29, 48]. However, it should be noted that in these representations, while the contact artifact is deformable, the model representation of the human is typically approximated with a rigid-body definition [29], which neglects the effects of deformation on the human body. While such an assumption is valid for garment draping, for a pressurized EVA suit, this assumption has less merit resulting from the increased rigidity of the suit in pressurized conditions [23].

While the discrete nature of the Particle-based formulation does result in higher computational cost relative to a Multi-Body Surrogate representation, it is significantly less expensive relative to a FEM formulation [28, 49]. Additionally, the ability to drive either mathematical- or physics-based inter-particle constitutive relationships provides a great degree of flexibility in driving the transient and static response of a contact artifact. However, without implementing a physics-based constitutive law between the particles, such as a mass-spring-damper approximation of elastic material properties, information regarding the stresses acting within the contact artifact will not be resolved. Furthermore, if the deformation of the human is also intended to be captured in a simulation, a Particle-based method will approach the effectiveness of a shell-based FEM model of the human due to its shell-like representation of the human. While this assumption may be valid while the contact artifact is less rigid relative to the human, as the contact artifact rigidity increases, the assumption that the human can be represented as an inflexible shell body weakens, as the human body deformation increases without accounting for the three-dimensional stiffness effects of the skeletal structure.

1.5 Relevance to Musculoskeletal Forward Dynamics

The musculoskeletal model component of the overall integrated human-suit model is responsible for taking in the resultant loads on the human as computed by the CCM model and determining the muscle forces necessary to achieve the motion profile. While the focus of this work is primarily on the CCM model, it is relevant to discuss the selection of the musculoskeletal modeling environment, how the CCM model interfaces with the musculoskeletal model, and a high-level view of the methodology of the musculoskeletal model.

The open source biomechanical modeling software, OpenSim [25] was selected as the component software for the musculoskeletal model portion of the integrated human-suit model . OpenSim is a powerful modeling environment for biomechanical problems, as it contains toolkits for not only musculoskeletal kinematics and kinetics, but also a means by which to use inverse dynamics [50] and static optimization [51] to compute the muscle recruitment and activation required to achieve a prescribed motion for the human model. Given the open-loop nature of the overall human-suit model noted in Section 1.3, this capability to determine the requirements on the human operator to achieve a given motion, under the external loads from the donned EVA suit as calculated by the CCM model, provides results which can be used to inform suit design to reduce risk of injury and improve efficiency for a given motion. Given the open-source nature of OpenSim, connectivity between the CCM model and the OpenSim environment can be developed to support the pipeline illustrated in Figure 1-4.

The terminology of inverse dynamics refers to the concept of taking prescribed motion data for a human along with any exogeneous forces acting on the human during that motion, the dynamics of that motion can be inverted along the lines of the common $\boldsymbol{F} = m\boldsymbol{a}$ to determine the forces the human must generate to accomplish the motion. For joint kinematics, the general equations of motion can be expressed

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \boldsymbol{G}(\boldsymbol{\theta}) + \boldsymbol{\tau_{exo}} = \boldsymbol{\tau_{required}} \tag{1.1}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{C}$ is the Coriolis jacobian, $\mathbf{G}$ is the gravitational force vector, τ_{exo} are the exogeneous torques acting on the human, such as those computed by the CCM model from human-suit interactions, and $\tau_{required}$ are the unknown torques the human must generate to achieve the motion profile through θ, $\dot{\theta}$, and $\ddot{\theta}$.

Once the required torques to achieve the motion profile of the human under the exogenous loads prescribed from the CCM model have been determined, a process known as static optimization is employed to determine the muscle activation and recruitment required of the human to achieve these torques [51]. OpenSim can represent the muscles as either idealized force generators or as muscles subject to force-length-velocity constraint properties [52]. Using static optimization, the number of recruited muscles and their activation levels can be determined. This information can be used to infer fatigue and injury risk for the human and, coupled with localized pressures acting on the human as calculated by the CCM model, can potentially be used to guide suit design decisions to minimize injury and reduce the effective impedance of the suit on the human.

Further development of the pipeline between the CCM model and OpenSim is recommended, as noted in Section 4.2. However, the primary focus of this work was aimed at the CCM model component of the integrated human-suit model. The identification, selection, development, and testing of the CCM model is discussed in Chapters 2 and 3, and the structure of this discussion is reviewed in Section 1.6.

1.6 Overview of Thesis

This thesis is divided into three chapters. Chapter 2 details the methodology, development, and implementation of the Coupled Contact-Mechanics (CCM) model. It first details the selection of the class of deformable-deformable human-artifact contact formulation for the CCM model, and the identification of a computational framework in which to develop the model. The structural mechanics and contact mechanics methods considered and selected are discussed, with additional detail provided with

regards to the methodology of the explicit contact enforcement scheme used by the CCM model. The numerical algorithm which couples the structural and contact mechanics domains is described, and the software implementation of this coupled highlighted. In addition, Chapter 2 also provides a review of the efforts to enable the CCM model for parallel computation for runtime improvements. Finally, Chapter 2 concludes with a trade study identifying functional domains in which the CCM model should be employed, versus a numerically simple rigid-body contact approximation as a benchmark.

Chapter 3 discusses the methods and results of a CCM model validation study against experimental data. The contact problem domain of interest is developed into a test plan, and executed using an experimental benchtop rig. The CCM model is employed to replicate the benchtop results, and a comparative analysis of the experimental results against the CCM simulation results provided to highlight the capability of the CCM model.

Chapter 4 documents a summary of the conclusions from the trade study of Chapter 2 and the validation study of Chapter 3. From this, future work, and additional developments and improvements for the integrated human-suit model are recommended.

Chapter 2

Methodology

This section details the methodology and development of the CCM model for use in the integrated human-suit model architecture. This involves the selection of a primary contact model formulation as noted in Section 1.4, and the identification of a computational framework in which to implement the model. With a framework and model architecture selected, this section identifies the constitutive model selected for resolving structural mechanics, as well as the contact enforcement method used to model the deformable-deformable human-artifact contact problems germane to this work. A primer on contact enforcement methods is provided for background on alternate contact methods, along with a detailed discussion of the contact enforcement method selected for the CCM model. The means and algorithm by which the CCM model couples structural and contact mechanics is addressed, and the implications for this software development highlighted. Furthermore, the extension of the CCM model to enable parallel computation is reviewed, with computational scaling studies performed. Finally, to identify the problem domains in which the CCM model should be employed over a simplified, less expensive rigid body contact model, a trade study is performed. This trade study considers a range of relative model parameters, such as the ratio of Young's Moduli between an impactor and artifact, to identify the computational cost versus simulation accuracy associated with each model. From the results of this trade study, recommendations can be made regarding when the full fidelity CCM model should be used for improved accuracy in exchange for runtime

versus a simplified rigid body assumption.

2.1 Model Architecture Selection

One of the first steps in the development of a computational model for deformable human-artifact contact was the selection of a modeling approach. In Section 1.4, three primary model architectures were discussed, namely Multi-Body Surrogate models, Finite Element Method-based models and Particle-based models. As each architecture has merits and drawbacks, a trade had to be performed to select an approach for modeling human-suit contact problems.

Section 1.4 details the fundamentals of each of these methods and highlights some of the benefits and drawbacks associated with each method. To trade between these architectures, a set of desired model attributes were defined to score each architecture. The selected attributes are:

1) Computational Expense
2) Fidelity of Contact Enforcement
3) Fidelity of Structural Mechanics
4) Arbitrary Model Geometry Capability

The first attribute, computational expense, as defined for the sake of this trade is the total computational runtime required to fully resolve a transient problem. Note that this distinction for runtime, and not also computational complexity, was explicitly selected to not artificially debit methods that scale in parallel computation. The second attribute, fidelity of contact enforcement, represents both the accuracy of the contact model and the ability to interrogate low-level data from the model. The third attribute, fidelity of structural mechanics, represents the accuracy of the model to resolve resultant deformations and stresses, and also the ability to interrogate low-level structural data from the model. Lastly, the fourth attribute, arbitrary model geometry capability, represents the ability for the model to capture any arbitrary geometric

shape for the human or artifact.

With these metrics in mind, and based on the literature discussed in Section 1.4, a comparative trade stackup was completed. The results of this qualitative trade are represented quantitatively in Table 2.1, where a higher numerical ranking represents the more capable model.

Metric	Multi-Body Surrogate Models	FEM-based Models	Particle-based Models
Computational Expense	3	1	2
Fidelity of Contact Enforcement	1	3	3
Fidelity of Structural Mechanics	2	3	1
Arbitrary Model Geometry Capability	1	3	3
Total Score	7	10	9

Table 2.1: Summary of Model Architecture Trade

While FEM-based models may be known to incur the highest computational expense of the architectures, it offers a framework that can account for arbitrary geometries while resolving both contact and structural mechanics with variable resolution. Particle-based methods, while less computationally expensive, cannot resolve structural mechanics beyond shell approximations. Due to the capability to resolve arbitrary geometries, and improved fidelity in structural mechanics, a FEM-based architectural approach was selected for development of this human-artifact contact model. With a FEM-based architecture selected, the development of the model could be decomposed into the selection or development of two components: a structural mechanics solver, and a contact enforcement method.

2.2 Structural Mechanics Framework and Model

To resolve the internal stresses in the human and artifact that result from a contact impulse or other exogeneous force, a structural mechanics computational framework is necessary. A wide variety of frameworks exist, including a wide range of commercially available codes such as ANSYS. Some commercial codes already contain contact resolution methods as well [34]. However, the ability to integrate with or modify these codes are limited due to their closed-source nature. Furthermore, their ability to scale in parallel computation may be limited [34] or restricted to specific contact enforcement methods [35]. This is not to say that these codes are not effective and capable, but for the purposes of this project and due to these limitations, an open-source computational framework was desired.

The computational framework SUMMIT, developed at MIT by Professor Raul Radovitzky, is an open-architecture structural and multiphysics environment. It supports both continuous and discontinuous Galerkin methods for multiphysics problems [53], and a wide array of constitutive material models [54, 55]. It also has extensive parallelization support for its structural methods through METIS and ParMETIS [56]. Lastly, it contains both rigid body contact mechanics methods, and a legacy explicit contact mechanics model that will be discussed in Section 2.4.2. Due to the open architecture design of SUMMIT allowing users to directly interact with and modify the source code to integrate external models, such as contact, coupled with the pre-existing capabilities of SUMMIT in parallelization and modeling methods, SUMMIT was selected as the computational framework for this work.

2.3 Constitutive Material Models

The SUMMIT computational framework supports a wide variety of constitutive material models. While the intent of this work is not to identify the appropriate constitutive model to represent the various materials used in the Mark III or Demonstrator Suit, it is germane to identify a constitutive model that is appropriate for simulat-

ing large displacement simulations with coupled contact interactions. Two primary material models were considered for use in this work: a linear elastic model and a non-linear Neo-Hookean model.

The first constitutive model considered in this study was the application of an isotropic linear elastic model. The stress response of this constitutive model can be expressed by

$$\sigma_{ij} = \lambda \delta_{ij} \epsilon_{kk} + \mu(\epsilon_{ij} + \epsilon_{ji}) \tag{2.1}$$

where σ is the Cauchy stress, λ is the first Lamé coefficient, δ is the Kronecker delta, ϵ is the strain, and μ is the second Lamé coefficient, with subscripts i, j, and k representing the principal axes [57].

This linear constitutive model was initially evaluated for the baseline material model for the structural mechanics system. However, due to the nature of the large deformations present in the transient simulations considered for contact, this material model would unrealistically deform entities. This issue becomes apparent when, as strain increases with deformation, stress increases in a similar linear fashion. For large deformations, this results in unrealistic stress-strain relationships, and as such, was not selected for the primary constitutive model in this study.

The second constitutive model considered in this work, and the model that was selected and used, is the nonlinear Neo-Hookean hyperelastic model. It is relevant to note that in hyperelastic formulations, a nonlinear relationship between stress and strain is permitted, which is advantageous to large deformation problems, such as those involving contact. For this constitutive model, the primary governing equations are for the strain energy density function

$$W = \frac{\lambda}{2}\ln^2(J) - \mu \ln(J) + \frac{\mu}{2}(\mathrm{tr}(C) - 3) \tag{2.2}$$

where λ and μ are the first and second Lamé coefficients, J is the determinant of the deformation gradient, and C is the right-Cauchy Green deformation tensor [58]. From this strain energy density definition, the response can be written as

$$S_{ij} = C_{ij}^{-1}\lambda \ln(J) + \mu(\delta_{ij} - C_{ij}^{-1}) \tag{2.3}$$

where S represents the second Piola Kirchhoff stress [54]. The Neo-Hookean constitutive model was found to be an effective baseline for this contact-mechanics model study, as unlike the linear elastic model, the Neo-Hookean model could resolve large deformations without generating unrealistic stress responses within the material. However, it should be noted that a full analysis of appropriate constitutive models will be necessary to accommodate the unique material properties associated with softgood material weaves in EVA suits. Further investigation into additional constitutive models, including superelastic models, is recommended to consider models which can accommodate the multiple static equilibrium postures of softgood suits such as the Demonstrator suit. However, for the purposes of the development of this model, the nonlinear Neo-Hookean constitutive model is sufficient.

2.4 Contact Mechanics Model

As highlighted in Section 1.2, to resolve the interactions between a human operator and a donned EVA suit, contact between the suit and the human, and self-contact within the suit must be captured by this model. As a result, a contact enforcement scheme that is capable of enforcing contact for arbitrary geometries in both human-artifact and artifact-artifact interactions must be identified or developed. This identification is a nontrivial process. To implement a nonlinear contact mechanics solver within the context of a finite element analysis, special care must be considered in addressing the two primary steps in resolving contact, beyond considering the integration of these methods into the formulation for structural mechanics:

1) Contact Search
2) Contact Constraint Enforcement

The first step, the contact search, involves the process of interrogating the compu-

tational domain for any geometric penetration that violates the defined contact constraint. A wide range of formulations exist, the simplest being the nearest-neighbor approach [59], in which a master surface is checked against a slave surface to enforce the geometric contact constraint. However, this approach is ineffective for problems involving self-contact or nondeterministic contact pairings. To resolve the ineffectiveness of nearest-neighbor, methods such as a global search method [59], or a two step iteration can be employed [60]. Additionally, the geometry which is considered for determining geometric proximity can vary to use a single or combination of the finite element nodes, edges, centroids, or quadrature points to detect geometric penetration. Furthermore, for parallel computation, the manner by which the domain is partitioned plays a large role in the selection of the method and complexity of the contact search, which will be discussed further in Section 2.7.2.

Once a contact constraint is found to be violated through the contact search, the nature by which the constraint is enforced is the Contact Constraint Enforcement step. Several categories of contact enforcement schemes have been proposed and implemented in finite element applications. In general, however, these schemes can be categorized into one of three categories, namely, contact penalty methods, Lagrangian multiplier methods, augmented Lagrangian methods [60]. While an extensive review is not necessary for the intent of this work, it is relevant to provide a basic summary of these methods when introducing the method employed in this model. To begin, however, it should be noted that the the general equation of motion for a problem involving contact can be expressed by

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \frac{\partial \boldsymbol{Q}(\dot{\boldsymbol{x}}, \boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{f}_{ex} + \boldsymbol{f}_{contact} \tag{2.4}$$

where M is the mass matrix, Q is the internal energy, f_{ex} are exogenous forces, $f_{contact}$ is the force necessary to comply with the contact constraint, and x is the position vector. The means by which this contact force is calculated and introduced into the equation of motion is the main differentiation between these methods.

2.4.1 Pedigree of Contact Enforcement Methods

Contact Penalty Methods

For a contact penalty method, the contact force of Equation 2.4 can be generally represented by a mechanical spring, expressed as

$$\boldsymbol{f}_{contact} = K_P \boldsymbol{\Delta} \tag{2.5}$$

Here, x is the finite location of interest, x_Γ is the nearest location to x on the contact constraint, and K_P is the contact penalty parameter. Figure 2-1 provides an illustration of this contact force representation.

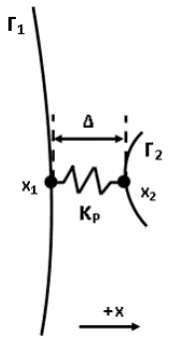


Figure 2-1: Contact Penalty Method Force Formulation

The distance between Γ_1 and Γ_2 form the penetration parameter Δ in this frame by

$$\begin{aligned} \boldsymbol{\Delta} &= (\boldsymbol{x}_1 - \boldsymbol{x}_2), \\ \boldsymbol{\Delta} &\geq 0 \end{aligned} \tag{2.6}$$

It should be noted that Δ is a nonlinear piecewise defined function, effectively representing a compression-only spring which ensures when the contact constraint is not violated, Δ is 0 and thus $f_{contact}$ is also 0. While this formulation is relatively straightforward to conceptualize, the challenge arises in the tuning of the penalty

coefficient K_P. If K_P is too large, the contact enforcement will dominate the forces in a given time step in the simulation, and in an explicit scheme, can contribute to instability [36]. Furthermore, the contact constraint can only be exactly represented as K_P tends to infinity, which is not realizable in computational practicality. As a result, the contact constraint is not necessarily completely enforced, and some geometric penetration may occur. Regardless, penalty methods are relatively common [36, 60, 61], and SUMMIT also includes a contact penalty scheme for resolving rigid body contact problems, which will be discussed further in Section 2.8.

Lagrangian Multiplier Methods

Methods that fall into the Lagrangian Multiplier class take on the general form of Equation 2.4, with $f_{contact}$ being replaced by

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \frac{\partial \boldsymbol{Q}(\dot{\boldsymbol{x}}, \boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{f}_{ex} + \boldsymbol{G}^T \boldsymbol{\lambda} \tag{2.7}$$

where G represents the constrained nodal displacements along the contact boundary, and must satisfy

$$\boldsymbol{G}[\boldsymbol{x} + \boldsymbol{u}] = 0 \tag{2.8}$$

where x are the nodal positions and u are the nodal displacement vectors [60]. Equations 2.7 and 2.8 are solved as a system to find λ, which is the Lagrange multiplier from which these methods derive their name [60]. Many computational frameworks incorporate Lagrangian Multiplier methods for contact enforcement, including ANSYS [34] and ADINA [62]. Lagrange multiplier methods are more numerically intensive to satisfy relative to penalty methods [36], but are notably more effective for exactly enforcing the contact constraint [63].

Augmented Lagrangian Methods

To address the limitations of the non-infinte K_P used in contact penalty methods, an augmentation was made to the formulation for the contact penalty noted in Equation

2.5, given by

$$\boldsymbol{f}_{contact} = K_P \boldsymbol{\Delta} + \boldsymbol{\lambda} \tag{2.9}$$

where the additional term λ is the Lagrangian adder. This augmented formulation of the contact penalty method represents a class of contact enforcement known as Augmented Lagrangian methods [64]. Here, the Lagrangian adder must be calculated iteratively for each node to guarantee the contact constraint is respected, given the user-defined K_P and geometric penetration at that time step. Augmented Lagrangian methods have the unique advantage of enjoying the relatively easier numerical scheme of the penalty method for estimating contact energy, while using the Lagrangian adder to mitigate erroneous penetration resulting from the penalty method and enable exact contact constraint enforcement [60].

While the three noted contact enforcement categories noted here are effective and commonly used, the contact constraint enforcement method selected for the model developed in this work does not directly fall within any of the above categories. For this project, the contact constraint enforcement method selected was the Contact Decomposition Response method, a hybrid Lagrange multiplier method that can be resolved explicitly with its similarities to Particle-based schemes noted in Section 1.4.3.

2.4.2 Explicit Decomposition Contact Response Method

While the contact enforcement methods noted in Section 2.4.1 are commonly used formulations to resolve nonlinear contact mechanics, each comes with notable advantages and drawbacks. Contact penalty methods are numerically simple to implement, but do not exactly enforce the contact constraint, and also become increasingly numerically unstable as the penalty K_P is increased. Lagrangian Multiplier schemes exactly conserve the contact constraint, but the need to implicitly solve the coupled Lagrangian multiplier equation at each node during a given iteration significantly increases the computational expense of the method. Augmented Lagrangian methods

represent a compromise of the methods, but must suffer from the need for an implicit nodal calculation to determine the Lagrangian adder in a given time step.

The explicit contact enforcement method developed by Cirak and West, known as the Decomposition Contact Response (DCR) method, was identified as it is a purely explicit scheme that also ensures exact enforcement of the contact constraint using an enforcement method similar to a Lagrange multiplier method. The fundamentals of this method as noted by Cirak and West are summarized and described herein, but their work should be consulted if greater detail is desired [65]. This method is unique relative to the methods noted in Section 2.4.1, in that in separates the enforcement of the contact constraint from the momentum conservation of the method, which is what enables this method to maintain these advantages. The DCR method takes the equations of motion noted in Equation 2.4, and represents the impact force as a discrete integral evaluated before and after impact,

$$\boldsymbol{f}_{contact} = \left[\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{x}}}\Delta \boldsymbol{x} + \boldsymbol{L}\Delta t\right]_{t_-}^{t^+} \tag{2.10}$$

where L is the semi-discrete Lagrangian for the problem, expressed as

$$\boldsymbol{L} = \dot{\boldsymbol{x}}^T \boldsymbol{M}\dot{\boldsymbol{x}} - \boldsymbol{Q} + \boldsymbol{f}_{ex}\boldsymbol{x} \tag{2.11}$$

Here, $\mathbf{M}$ is the mass matrix, $\mathbf{Q}$ is the internal energy, $\Delta\mathbf{x}$ and $\dot{\boldsymbol{x}}$ are the nodal displacements and velocities, respectively, and $\mathbf{f}_{\text{ex}}$ is the exogenous force vector. To enforce the contact constraint, the following must be satisfied, where

$$\nabla \boldsymbol{g}(\Delta \boldsymbol{x} + \dot{\boldsymbol{x}}\Delta t) = 0 \tag{2.12}$$

To accomplish this, either of the following conditions can be met, where

$$\Delta \boldsymbol{x} = -\dot{\boldsymbol{x}}\Delta t \tag{2.13}$$

or

$$\nabla \boldsymbol{g}\Delta \boldsymbol{x} = 0, \quad as\ \Delta t = 0 \tag{2.14}$$

The use of either Equation 2.13 or 2.14 in 2.11 results in

$$\left[\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{x}}}\right]_{t^-}^{t^+} = \lambda \nabla \boldsymbol{g} \tag{2.15}$$

which takes the form of a Lagrange multiplier, and

$$\left[\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{x}}}\dot{\boldsymbol{x}} - \boldsymbol{L}\right]_{t^-}^{t^+} = 0 \tag{2.16}$$

At this point, attention should be called to the significance of the partial of the semi-discrete Lagrangian, noted in Equation 2.11, with respect to nodal velocities reflecting the momentum vector of the system, $\boldsymbol{p} = \boldsymbol{M}\dot{\boldsymbol{x}}$, so that

$$\left[\boldsymbol{p}\right]_{t^-}^{t^+} = \lambda \nabla \boldsymbol{g} \tag{2.17}$$

which takes on a Lagrange multiplier form previously identified in Equation 2.7, and

$$\left[\boldsymbol{p}^T \boldsymbol{M}^{-1} \boldsymbol{p}\right]_{t^-}^{t^+} = 0 \tag{2.18}$$

where λ in Equation 2.17 is any real scalar value. The implications of Equation 2.17 are that the only changes in momentum permitted are those normal to the contact constraint, ∇g. Similarly, Equation 2.18 highlights that, for an elastic process, momentum is conserved. Furthermore, if the momentum immediately prior to impact, $p(t^-)$ is known, then these equations can be used to compute the resultant momentum after the impact event, $p(t^+)$. Equations 2.17 and 2.18 can be solved explicitly across a given contact event, providing purely explicit scheme to compute momentum changes associated with an impact event, while enforcing geometric impenetrability through the constraint, g.

It should be noted that $\mathbf{g}$ has been explicitly undeclared, as this formulation is agnostic to the nature of the contact constraint. This is to say, $\mathbf{g}$ can be selected to be the gap $\boldsymbol{\Delta}$ from 2.6, the intersection volume, or any other preferred metric for enforcing geometric impenetrability. The only requirements for $\mathbf{g}$ are that

$$g(\boldsymbol{x}) = 0 \;\; on \; \Gamma \tag{2.19}$$

where the contact domain, Ω is defined by

$$\Omega = \left\{ \boldsymbol{x} \in Q | g(\boldsymbol{x}) \leq 0 \right\} \tag{2.20}$$

With Equations 2.17 and 2.18 defined for a given contact constraint, $\mathbf{g}$, the explicit contact discretization approach can be considered. In general, a contact event will occur between two time steps, such that $[t_{i-1} < t_c < t_i]$. To exactly compute the resulting momentum changes from contact, it could be considered to modify the time step to compute two separate intervals, namely $[t_{i-1}, t_c^-]$ and $[t_c^+, t_i]$, where i is the current time step, and exactly resolve Equations 2.17 and 2.18 across the contact event. However, this is not practical for a finite element approach, as a large number of individual contact events may occur during any given time step. Furthermore, for the DCR formulation, this approach is not necessary. Rather than resolve the contact event exactly, the current time step t_i can instead be "corrected" to enforce the contact constraint and impart momentum changes associated with the impact [65]. This approach is illustrated in Figure 2-2.

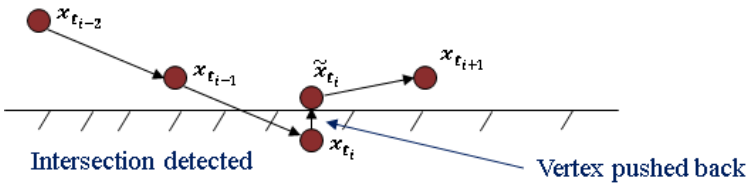


Figure 2-2: Explicit DCR Lumped Mass Vertex Momentum Conservation Concept - Node at position x at time t_{i-1} crosses the contact boundary Γ at time t_i, and is pushed back by contact enforcement to a corrected position $\tilde{x}$ at time t_i, conserving momentum across the contact event

This approach results in, for the consideration of the governing equations, the assumption that contact occurred at t_i. While a generalized momentum balance

could be considered at this point, the definition of the contact constraint, g, should be revisited due to its effect on the governing equations. As noted by Cirak and West, while this formulation permits the use of global geometric constraints, such as intersection volumes, to improve computational scaling and ease parallelization, local geometric quantities should be used [65]. For a generalized three-dimensional contact problem considered at the local element level, the types of contact can either be a node impacting an element face, or an edge impacts with another edge. In both cases, the constraint g can be defined as the signed volume of the tetrahedron formed from the combination of node and face, or edge and edge, for each respective contact condition. Figure 2-3 illustrates these contact types and the resultant tetrahedron for each.

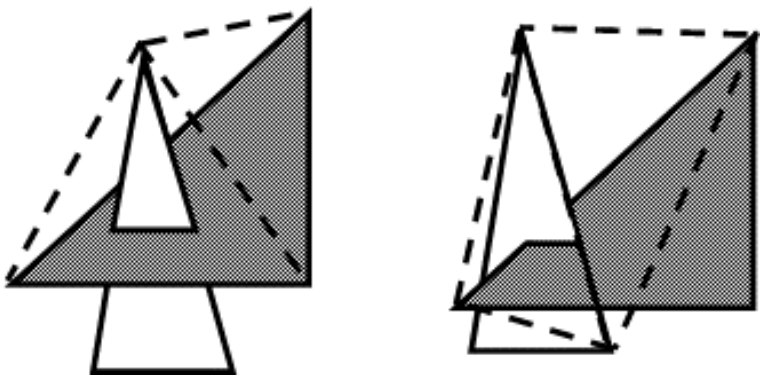


Figure 2-3: Considered Finite Element Contact Types and Resultant Intersection Tetrahedrons for DCR Method are node-face intersection (left) and edge-edge intersection (right)

With the contact constraint, **g**, defined as the intersection tetrahedron volume resulting from node-face or edge-edge contact of two triangular finite elements, the approach for enforcing the constraint can be implemented. As shown in Figure 2-2, for node-face contact, this involves projecting the vertex "back" to the limit of the contact surface, Γ, using closest-point projections [65]. The edge-edge contact condition projects the edge back to the closest point on the other edge in a similar

fashion. This process of projecting back the node or edge results in an increase in the internal energy of the system, and must be accounted for in the governing equations of motion for the contact event. In the finite domain, the internal energy increase associated with the projection results in the approximate definition of

$$(\lambda\nabla \boldsymbol{g} + \boldsymbol{p}_i^-)^T \boldsymbol{M}^{-1}(\lambda\nabla \boldsymbol{g} + \boldsymbol{p}_i^-) - \boldsymbol{p}_i^- \boldsymbol{M}^{-1}\boldsymbol{p}_i^- \approx (\boldsymbol{f}_{internal} - \boldsymbol{f}_{ex})(\boldsymbol{x}_i - \widetilde{\boldsymbol{x}}_i) \tag{2.21}$$

where $\widetilde{\boldsymbol{x}}_i$ denotes the nodal displacements after the "push back" step illustrated in Figure 2-2. With this formulation, the contact event can be presented as a momentum balance, but it remains to determine how to calculate the momentum component modifications during an arbitrary contact event. However, as noted by Cirak and West, these components can be readily calculated through a momentum decomposition, from which the method derives its name, wherein

$$\boldsymbol{p} = \boldsymbol{p}_t + \boldsymbol{p}_n \tag{2.22}$$

where $\boldsymbol{p}_t$ is the momentum tangential to the contact boundary, Γ, and $\boldsymbol{p}_n$ is the momentum normal to Γ. We know that, for a frictionless system, the normal momentum increases in the direction of $\nabla \boldsymbol{g}$, and by conservation of momentum

$$\boldsymbol{p}_n^- = -\boldsymbol{p}_n^+ \tag{2.23}$$

and

$$\boldsymbol{p}_t^- = \boldsymbol{p}_t^+ \tag{2.24}$$

Furthermore, as we know that $\boldsymbol{p}_t$ and $\boldsymbol{p}_n$ are by definition orthogonal, it follows that across a given contact event,

$$(\boldsymbol{p}_n^- + \boldsymbol{p}_t^-)^T \boldsymbol{M}^{-1}(\boldsymbol{p}_n^- + \boldsymbol{p}_t^-) = (\boldsymbol{p}^+)^T \boldsymbol{M}^{-1}(\boldsymbol{p}^+) \tag{2.25}$$

simplifies to

$$\boldsymbol{p}^{+} = \boldsymbol{p}_t^{-} - \boldsymbol{p}_n^{-} \tag{2.26}$$

From Equation 2.26, the momentum in the normal and tangential directions after an impact event at arbitrary time t_i can be found explicitly. Friction models and inelastic collisions can be accommodated within this framework as well [65]. At this point, the means by which a contact event is detected and enforced is complete for the DCR method. The remaining step is to determine a means by which to couple the contact detection and enforcement of the DCR method with the structural mechanics system described in Section 2.2.

The explicit DCR method presents several noteworthy advantages to its use. First, by nature of being an explicit method, this allows the method to be readily parallelized across multiple processors in parallel computation, which will be discussed futher in Section 2.7.2, as opposed to implicit methods which are not well-suit for parallelization as blocking processes must wait until the method converges. Secondly, it ensures the contact constraint is exactly enforced through the closest-point projection method, eliminating erroneous geometric penetration at any time step. Finally, and most importantly, the DCR method shares commonality with the Particle-based approaches noted in Section 1.4.3, in that each node and edge is projected independently, enabling contact enforcement for both external- and self-contact problems; a necessary capability for human-suit contact problems.

However, the DCR method does experience some limitations relative to other contact enforcement methods. First, as it ensures exact enforcement of the contact constraint, and *each* vertex and edge must be checked and enforced in every time step, it suffers from a greater computational cost relative to a simple penalty method, as will be highlighted in Section 2.8. Secondly, the contact constraint is enforced for each node as it is encountered in the simulation - this implies that, depending on mesh ordering and orientation, arbitrary nodes in two intersecting contact surfaces may be projected in the order in which the contact search finds each node. While

this does not violate the contact constraint, it represents a potential inconsistency in the method as a function of the contact search path. Finally, as the DCR method must track nodes and edges along the lines of a Particle-based method, it must store data structures containing information regarding the contact boundaries, where a simple contact penalty method would not, resulting in relatively higher memory consumption. Despite these limitations, the advantages offered by the DCR method, especially for self-contact and parallelization, drove its selection as the contact enforcement method for the CCM model.

2.5 Coupled Contact-Mechanics Architecture

With the nonlinear Neo-Hookean constitutive material model selected for structural mechanics, and the explicit DCR method selected for contact enforcement, the next step was to develop the coupled model architecture that can be used to resolve deformable human-artifact contact problems. The DCR method is inherently explicit, and for parallel computation it is ideal to construct a purely explicit solver sequence. Additionally, given that the model pipeline noted in Section 1.3 intends to drive motion profiles open-loop (no human feedback response to the motion), an explicit numerical scheme is preferable over an implicit scheme.

2.5.1 Explicit Integration Scheme

Explicit numerical integration schemes are numerous, ranging from the simplest first-order method in Forward Euler, to the commonly known class of explicit Runge-Kutta methods. For the coupled contact-mechanics problem at hand, an explicit scheme that lends itself well to the ability to inject the updated momentum after a contact event is desired. One such class of explicit numerical integrations schemes that is separable into such a multi-step solution procedure is the second-order Newmark-Beta method [66]. This method is widely used in finite element structural analysis due to its flexibility to account for nearly any constitutive material model, in both elastic and plastic deformation. The explicit Newmark-Beta method was employed

as the baseline numerical integration scheme for this model.

To generally describe the integration procedure that defines the Newmark-Beta method, we begin with the definition of the first temporal derivative, velocity in the case of our system, as

$$\dot{x}_{i+1} = \dot{x}_i + \Delta t(1-\gamma)\ddot{x}_i + \Delta t\gamma\ddot{x}_{i+1} \tag{2.27}$$

where γ is a scalar bounded by $0 \leq \gamma \leq 1$. The value selected for γ highlights how much to bias the next velocity in favor of either the prior acceleration ($\gamma < 0.5$) or the next acceleration ($\gamma > 0.5$). Newmark demonstrated that selecting γ to be 0.5 is generally advisable, as erroneous numerical damping will be introduced proportional to $\gamma - 0.5$. Additionally, as acceleration will not be constant for human-suit contact problems, the displacements must also be computed by

$$x_{i+1} = x_i + \dot{x}_i\Delta t + (0.5-\beta)\ddot{x}_i\Delta t^2 + \beta\ddot{x}_{i+1}\Delta t^2 \tag{2.28}$$

While Newmark noted that values for β could be selected such that $0 < \beta < 0.5$ for a $\gamma = 0.5$. However, for structural mechanics, selecting $\beta = 0$ recovers the central difference formulation of the Newmark-Beta method. As this class of the Newmark-Beta method is known to be symplectic, or a method which conserves the relationship between displacement and velocity, and momentum-conserving [67], which are primary objectives for the purposes of this model, the Newmark-Beta method with $\gamma = 0.5$ and $\beta = 0$ was selected. However, simply implementing the central difference Newmark algorithm is insufficient for our coupled contact-mechanics system. Without a means of correcting the computed displacements and velocities of the mechanical system to account for the contact constraint, this method would be ineffective. However, the Newmark-class of algorithms, and particularly the central difference class of Newmark-Beta methods, lend themselves well to a Predictor-Corrector form that is well-suited for coupled, nonlinear systems [68, 69].

2.5.2 Contact-Mechanics PEC Formulation

To integrate the contact enforcement step into the Newmark integration method, the Newmark algorithm is adapted into a Predictor-Corrector form. This restructuring results in a two-step process: prediction, given by

$$x_{i+1}^{p} = \dot{x}_i \Delta t + 0.5 \Delta t^2 \ddot{x}_i \tag{2.29}$$

and

$$\dot{x}_{i+1}^{p} = (1-\gamma) \Delta t \ddot{x}_i \tag{2.30}$$

From here, the residuals resulting from application of these displacements and velocities on the constitutive equations are computed and the predicted acceleration term $\ddot{x}_{i+1}^{p}$ found. In a pure two-step Predictor-Corrector method, the corrector step would be applied next. However, for the coupled contact-mechanics system, it is at this step *after* the nodal displacements, velocities, accelerations and residuals have been computed, that the DCR method is used to compute updated nodal displacements and velocities per the method described in Section 2.4.2. The distinction that the predicted accelerations and residuals are computed prior to application of the DCR method is significant, as it highlights a primary difference in that the nodal displacements and velocities will be *exactly* enforced on the structural mechanics system. If the ordering were to be reversed, then the contact enforcement mechanism would need to augment into computing the necessary nodal forces required to achieve the nodal displacements and velocities prescribed by the DCR method. Otherwise, the exact satisfaction of the contact constraint would be violated. This internal step where the DCR method is evaluated represents the construction of this method into one of a Predict-Evaluate-Correct (PEC) class.

After updating the nodal displacements and velocities with those from the DCR method, the final corrector step is applied to the velocity such that

$$\dot{x}_{i+1} = \gamma \ddot{x}_{i+1} \Delta t \tag{2.31}$$

where $\ddot{x}_{i+1}$ represents the accelerations prescribed from the resolution of the constitutive equations. After correcting the velocities, the model is advanced to the next time step, and the process repeated. To summarize this algorithm,

1) Nodal displacements and velocities predicted based on prior step acceleration, independent of the contact constraint

2) Constitutive model updates accelerations and residuals

3) DCR method checks for contact constraint violation, and updates nodal displacements and velocities to enforce contact and conserve momentum

4) Nodal velocities are corrected to account for next step accelerations from constitutive law

5) Time step advances, process repeats

This PEC method describes the coupled contact-mechanics model developed and implemented in SUMMIT for studying deformable human-suit contact problems.

2.6 Software Architecture and Communication for Serial Computation

While Section 2.5 provides a review of the numerical architecture for the coupled contact-mechanics model, the primary effort associated with the development of this model was in the integration of the SUMMIT structural mechanics domain with the DCR contact mechanics domain. The implementation of the DCR method as described by Cirak and West [65] has been developed in preexisting software [70], as are the constitutive models and structural system formulations in SUMMIT. This legacy software employed a similar PEC method to couple an explicit dynamics code with the DCR method, but this legacy code was found to suffer from memory leaks, and shared data between the structural and contact solvers at a global scope; a nonviable structure for interacting with an arbitrary computational framework. As a result, the

legacy DCR code had no means to interact with or communicate with SUMMIT, nor any means for SUMMIT to interact with the DCR methods. Furthermore, the legacy code which implemented the DCR method had been unmaintained for some time, and needed to be updated to work with modern compilers and platforms. Additionally, extensive software verification needed to be performed on both the legacy code and the new software developed in this work to generate the coupled contact-mechanics model. This section details the nature of the software architecture developed to integrate the legacy DCR implementation with SUMMIT, how the PEC solver acts and communicates with the various data structures, and finally the software verification steps taken to enhance runtime and minimize memory consumption.

As noted in Section 2.4.2, to resolve the contact mechanics problem, a concept of the contact boundary, Γ, and the nodal displacements are necessary to check for geometric penetration of the boundary. Furthermore, the nodal lumped mass and velocities must be known to compute the updated nodal momentum during a contact event. From this information, a minimum set of data must be communicated between SUMMIT and the DCR method. Figure 2-4 illustrates the data structure, referred to as the "Contact Surface" object, that is used by SUMMIT to interact with and invoke the DCR method, and how it interacts with the two domains.

As the model is implemented in the SUMMIT computational framework, the Contact Surface object is initialized and constructed based on the knowledge of the discretized domain provided to it. This knowledge of the discretized domain is provided to it via a boundary object, which is a collection of discretized triangle surface elements from the full domain. This boundary object is arbitrary by design, and can represent either every outer surface present in a domain, or specifically isolated element regions. However, the Contact Surface object has no concept or need for tracking the global discretized domain, but only how the boundary elements provided to it are related. As such, a memory map, noted in Figure 2-4 as the Mechanics Nodal Mapping, is a vector pairing that identifies which nodes within the contact surface correspond with which nodes in the global structural domain. This mapping is critical to generate and correctly maintain, as it is the means by which nodal quan-

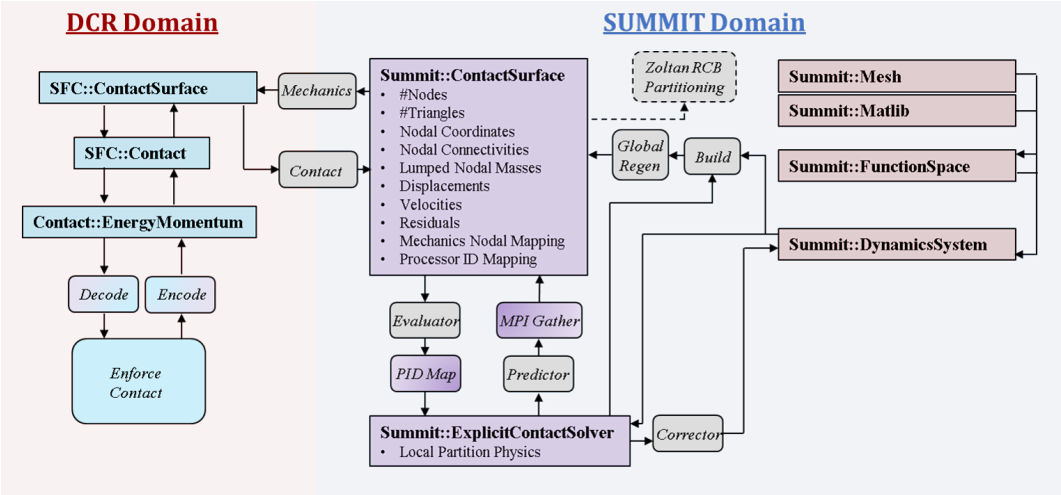


Figure 2-4: SUMMIT-Based Contact Surface Data Structure and I/O Connectivity - SUMMIT base code (pink), DCR base code (blue), Contact Surface developed code (purple), where bold blocks are classes, italicized blocks are functions, and blocks with gradient fills are parallel functions

tities such as displacement, mass, and velocity can be extracted from the structural domain and updated in the Contact Surface object during the Predictor step of the PEC algorithm.

To invoke the DCR contact method, the contact domain must be instantiated. The Contact Surface object is the owner of a generalized DCR Contact object, which is the DCR contact domain object. This object contains the contact search and enforcement methods of the DCR method, and maintains its own local contact surface "mesh." While this may initially appear to be double-dimensioning the contact domain, the separate maintenance of the contact domain mesh is particularly important for parallel partitioning, and will be discussed further in Section 2.7.2. The Contact Surface object uses the updated nodal quantities it extracts from the SUMMIT structural domain after the Predictor step, and communicates these updates to the DCR contact domain. The DCR contact search and enforcement methods are invoked by the Contact Surface object, after which the Contact Surface object then extracts the updated nodal displacements and velocities from the DCR contact mesh, forming the Evaluate step of the PEC algorithm. Finally, the Contact Surface communicates these updated nodal quantities to their corresponding nodal pairs in the SUMMIT

structural domain. At this time, the Corrector step of the PEC method is invoked to correct the nodal velocities based on the structural forces and ends the iteration for that time step. Figure 2-5 illustrates this PEC process through the two domains via the Contact Surface.

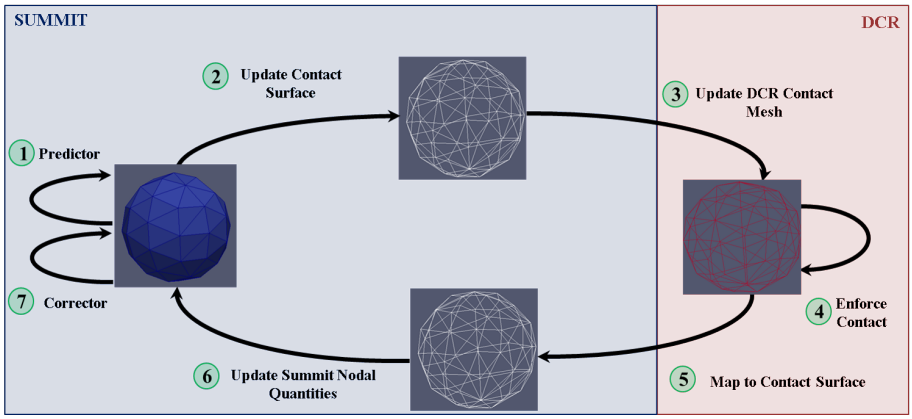


Figure 2-5: Coupled Contact-Mechanics Data Flow Representation in Serial Computation

Further description of the Contact Surface, including detailed description of its methods and input-output mapping can be found in Appendix A. Additionally, note that the methods "Global Regen," "Zoltan RCB Partitioning," and "MPI Gather," highlighted in Figure 2-4 are germane to the topic of domain partitioning and will be discussed in greater detail in Section 2.7.2.

Once the Contact Surface class was developed for SUMMIT and the legacy DCR code was updated to compile, build and link successfully, it was prudent to perform a detailed software analysis for memory consumption and identification of computational bottlenecks. To accomplish this, the execution-driven callgraph routines and dynamic binary analysis methods of the tool suite Valgrind [71–73] were employed. The callgraph functionality of Valgrind was used to diagnose and improve high-cost methods employed by the Contact Surface. In particular, this callgraph and cycle cost calculation method was used to isolate regions of the code that did and did not

benefit from code optimization methods during compile time. An example of such a back-to-back functional analysis is illustrated in Figure 2-6 for a test case simulation using the contact-mechanics model. For reference, the exact test case considered is detailed in Section 2.7.3. Note that the full function paths have been removed from Figure 2-6 and domains to which these functions belong highlighted instead. Note that the "Debug" results refer to compiling, linking and building the model using the Gnu Compiler Collection (GCC) with no optimizations or function in-lining to allow for traceable code. The "Optimized" results reference the CCM model built using GCC optimizations at level 3 [74], taking advantage of function in-lining and compilation optimizations to improve code runtime.

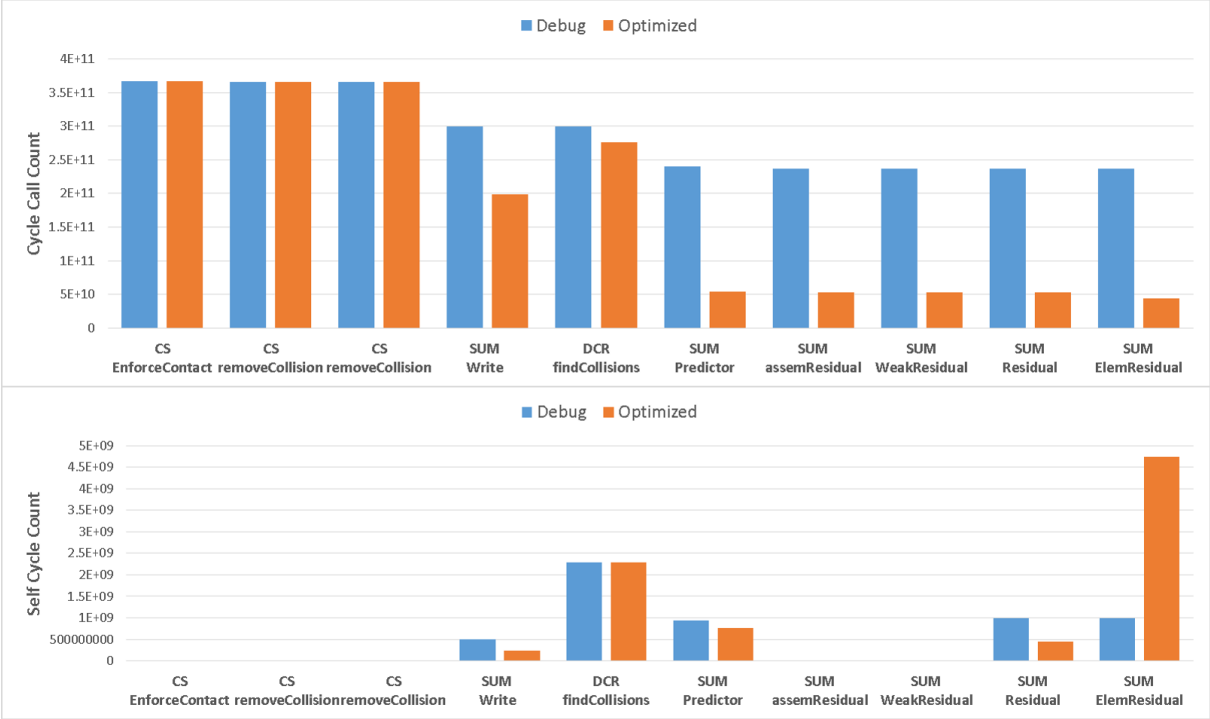


Figure 2-6: Example of Back-to-Back Software Cycle Comparison for Debug and Optimized Contact-Mechanics Model Code, Total Function Cycle Counts [top] versus Self-Cycle Counts [bottom], where SUM is a SUMMIT domain code, CS is a ContactSurface domain code, and DCR is a DCR domain code

Figure 2-6 provides an example of the quantitative results obtained from using callgraph functionality to track the cycle count of the CCM model. The top plot highlights ten of the functions in the CCM model with the highest cycle counts, implying these methods were called most frequently in Debug mode, and the corresponding cycle count reduction from optimization for these functions. The lower plot, however, highlights the number of "self cycles" or the number of internal function cycles those routines required. The total cycle count represents those functions that dominated the runtime of the CCM model, while the self cycle count highlighted methods with significant internal cost. These quantitative measures are not meant to provide a definitive means of highlighting code optimality, but to illustrate how the code was interrogated to identify costly functions.

While the callgraph functionality of Valgrind was useful for identifying areas for improvement within the code, the memory leak detection offered in Valgrind was instrumental to the successful completion of the model. As the legacy DCR code had not been maintained for some time, this legacy release needed to be verified for proper memory allocation and deallocation when interacting with SUMMIT. Over the course of testing the code with Valgrind, significant memory leak issues were not only sorted out within the implementation of the SUMMIT Contact Surface interface and methods, but a memory leak issue was identified and corrected in the legacy code as well. Without this correction, expensive computations would consistently consume so much memory as to cause the operating system to forcibly kill the process. After completing the memory leak check and correction process for both the SUMMIT Contact Surface and the legacy DCR code, all memory leaks associated with the transient simulation, and thus those that can grow with time, were eliminated, enabling the model to be employed in HPC environments without risk to the cluster.

2.7 Computational Scaling and Parallelization

To evaluate the usability and computational cost associated with using this coupled contact-mechanics model, it is relevant to consider both computational complexity in

serial computation, and the capability for the model to scale in parallel computation. From understanding these computational cost studies, the end user can infer the computational runtime they can expect based on the complexity of their problem, and the advantages that can be gained with parallel computing for this model.

2.7.1 Serial Computational Complexity Studies

To study the computational complexity of evaluating a coupled contact-mechanics problem with this model, a numerical study was performed with increasingly refined finite element meshes. The intent of this approach is to highlight how the computational cost scales with increasing complexity of the model through higher element counts.

To study the computational cost scaling of the CCM model, a simple simulation was used, illustrated in Figure 2-7.

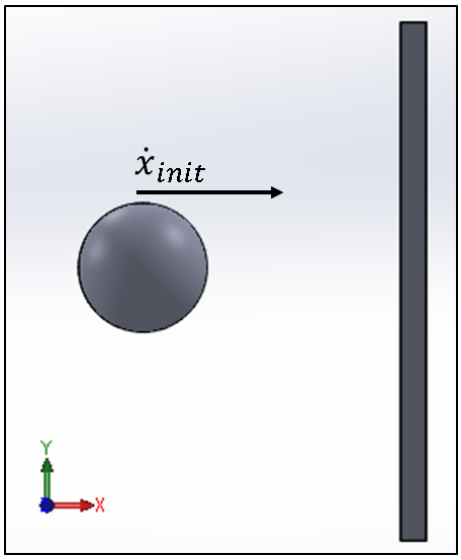


Figure 2-7: Simple Sphere-Plate Contact Problem for Serial Computational Cost Analysis

In this case, the sphere, with a radius of 0.006 meters (0.25 inches), is initially

separated from the plate, which has a width of 0.0095 meters (0.1 inches) and a length of 0.0095 meters (2 inches), by a distance of 0.0048 meters (0.75 inches). The sphere has an initial velocity of 2.0 inches per second. Additionally, the sphere was prescribed material properties of $\rho = 7000\ kg/m^3$, $E = 1000\ Pa$, $\nu = 0.3$, while the plate was prescribed $\rho = 1000\ kg/m^3$, $E = 70000\ Pa$, $\nu = 0.3$. While the exact values used here are not germane to the computational cost study, they were selected to reduce computational runtime at the coarsest mesh level, and they are documented for completeness and their effect on the simulation time step size.

To begin, a simplicial, coarse mesh was generated using the Solidworks meshing program [24]. This coarse mesh, combined with the noted material properties, were provided to the coupled contact-mechanics model and simulated for 1.0 seconds, with contact between the sphere and plate occuring at approximately 0.38 seconds, so as to ensure part of the problem involves only structural mechanics and the contact search, while the remainder contains structural mechanics, contact search, and contact enforcement. Figure 2-8 illustrates this transient simulation for reference.

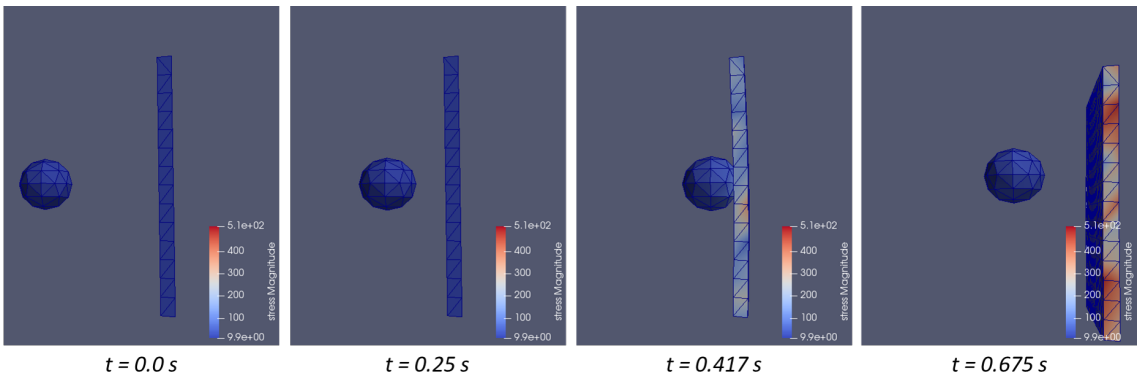


Figure 2-8: Transient Simulation Response for Serial Scaling Test Case, Coarse Mesh

From this coarse tetrahedral mesh, which contained 1527 tetrahedrons in the structural mechanics domain and 922 triangular faces on the outer boundaries, a series of mesh refinements were performed to increase the computational complexity

of the system. This mesh refinement was performed by subdividing each tetrahedron into eight separate tetrahedrons with equivalent local sizing, resulting in an eight-fold increase in the number of elements per subdivision. For each of these refined meshes, the same simulation highlighted in Figure 2-8 was run and the computational runtime associated with computing a given time step tracked at each iteration for both the contact and structural mechanics components of the integration scheme. The results of this study are provided in Figure 2-9.

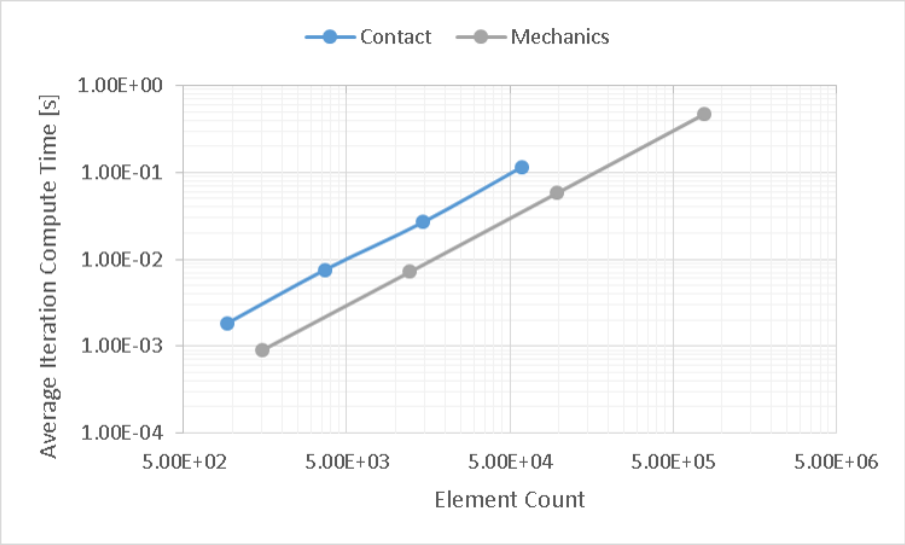


Figure 2-9: Computational Cost Scaling with Element Count in Serial Computation

Here, it was found that the relationship between the average iteration time for contact and structural mechanics for this system trended in a monomial fashion, with contact scaling by $t_{iteration} = 2 \times 10^{-6} N_e^{0.9878}$ and structural mechanics scaling by $t_{iteration} = 6 \times 10^{-7} N_e^{1.0033}$. The nearly 1.0 power of the monomial suggests a nearly log-log linear relationship between element count and computational cost for both the contact and structural mechanics method. This result is not unexpected given that the computational complexity of the contact is proportional to $\frac{N}{\Delta t}$ [65], where N is the number of elements and Δt is the time step, and highlights that

the computational cost will scale in a nearly linearly proportional fashion with the element count. However, it should be noted that the computational cost associated with checking for and enforcing contact is an order of magnitude more expensive than the structural mechanics component. This order of magnitude cost increase is primarily a function of the additional static cost of the contact search, which scales proportionally to N to check each node for contact, and potential cost of enforcement.

As the computational cost scales in a nearly proportional fashion with the complexity of the finite element domain, it follows that a point will be reached where the cost of using this model becomes too high for serial computation alone. Indeed, for these cases, as the element count decreased, the minimum element size decreased as well, resulting in a significant decrease in the stable time step size for the simulation. As a result, as the total number of iterations required to complete the simulation scaled with the minimum element size, and the average iteration time also scaled with the increase in element count, the cost of resolving increasingly refined finite element domains scales in a second-order fashion. While these effects are the result of the rigidity of the system in structural mechanics for maintaining the CFL number for numerical stability, this is a problem of practicality for complex or highly rigid systems. As a result, to enable a reduction in the total computational cost for the end user, enabling a parallel capability for the coupled contact-mechanics system is desirable.

2.7.2 Partitioning and Load Balancing Approaches

To enable a means by which to reduce the total computational cost of using this model for highly complex systems, parallel computation is desired. The general concept of parallel computation is relatively straightforward. If the work associated with computing the updated finite element data structures in a given iteration can be divided into equal, independent tasks, then the work can be distributed to multiple processors and computed simultaneously. While the total cost remains the same, the cost on a per processor basis reduces - ideally on a one-to-one basis with processor count. The effectiveness by which the computational workload can be evenly

split across multiple processors running in parallel, and the factor by which the run-time is reduced provide a metric for the scalability of model in parallel. Given that the relationship between finite element density and computational cost is essentially monomial in serial computation, there is significant motivation to offset this through parallelization.

To enable the model to run in parallel, the method by which the computational domain will be partitioned across processors must be determined. The SUMMIT computational framework was built with parallelism in mind, and for the nonlinear Neo-Hookean constitutive material model using Continuous Galerkin an equal-elements load balancing scheme is applied. Using the METIS partitioner [56], the finite element tetrahedrons are equally divided amongst the processors using a graph-based procedure, and a memory map between the processor boundaries constructed. In this fashion, each processor need only compute the elements within its own partition and to communicate with those processors which posses neighbor elements to resolve the constitutive equations during a given iteration. Figure 2-10 provides an illustration of how the equal-element partitioning method divides the domain in a 10 processor implementation.

It should be noted that the example domain shown in Figure 2-10 is using Continuous Galerkin tetrahedral elements, and that the plate and sphere objects possess differing material properties. Additionally, attention should be called to how spatial location and material properties may not always be consistent for elements in a given processor, as seen for core 9. However, the graph-based partitioning prefers to appropriate elements in consistent geometric regions when possible, as observed for all cores excluding 9 in 2-10.

This graph-based equal element load balancing method has been successfully applied to CG structural problems, and is known to scale well in parallel [75]. As a result, the graph-based, equal element partitioning scheme was employed for parallelizing the structural mechanics component of the model.

The contact mechanics domain must also be partitioned in such a way as to distribute the computational workload across the processors. As the coupled contact-

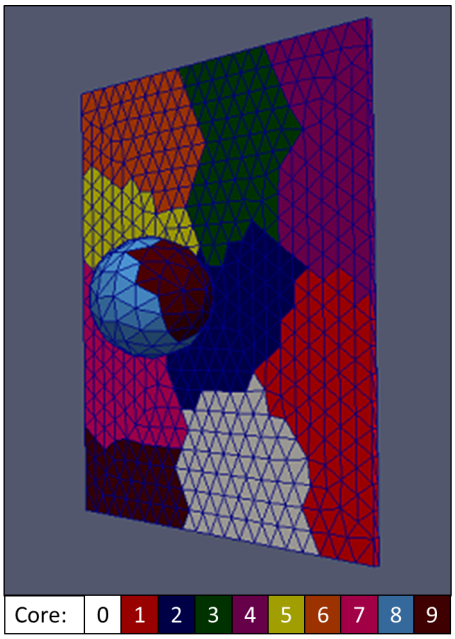


Figure 2-10: Structural Mechanics Domain Partitioning by Equal Element Load Balancing

mechanics model makes use of both a structural, solid finite element domain, and a contact, surface finite element domain, as referenced in Section 2.6, a separate partitioning scheme must be employed for the contact domain, independent of the structural mechanics domain.

As an initial attempt, an equal element approach can be implemented for the contact domain. In this method, the number of triangle elements are intended to be evenly distributed amongst the processors. Indeed, there is nothing inherently invalid about this process, but it is a flawed approach for properly balancing the workload in a contact problem. The primary issue with the equal element approach for contact is twofold:

1) Not every triangular element may necessarily be involved in a contact event.

2) Not every triangle needs to check for contact against every other triangle - only a comparison against those triangles in geometric proximity should be checked.

These two issues, while straightforward to highlight, represent the crux of the partitioning problem for contact. To guarantee that the contact constraint is exactly conserved with an equal elements approach, *every* triangle must be checked for penetration against every other node and edge. This is because processor X has no concept for the geometric location of processor Y's elements without querying all of those elements. Furthermore, if processor X contains all of the elements that are involved in contact events, while processor Y has elements that do not experience a contact impulse, the workload associated with only the contact check and not enforcement represents a workload imbalance across the processors.

To address these issues, a new partitioning scheme should be considered. In an ideal partitioning approach for contact mechanics, processors should be aware of which other processors represent those in its geometric proximity, and therefore those that could potentially come into contact. However, this is a dynamic problem, as over the course of a simulation, some elements may migrate closer or further away from others. Additionally, each processor should only need to communicate with those geometric neighboring processors to improve scaling. A partitioning scheme that is well-suited to such a task is known as the Recursive Coordinate Bisection (RCB) partitioning method [76], and is incorporated into the Zoltan Library [77, 78].

RCB partitioning is a conceptually intuitive partitioning method that effectively involves the recursive subdivision of the computational domain into 2^n subdomains, cut along cartesian coordinate axes. In greater detail, the method first detects which Cartesian axis (x, y, or z) of the domain spans the greatest length of the computational domain. The partitioner then generates a cutting plane orthogonal to that selected dimension, and subdivides the domain across that cutting plane. This process then repeats for each subdomain in a recursive divide-and-conquer scheme until the computational domain has been subdivided into a set of equivalently sized hyperrectangular subdomains. As the method of subdivision is coordinate based, this results in the creation of distinct geometric "neighbor" partitions. The Zoltan Library has an implementation of the RCB method as a dynamic, rather than static, partitioner method. Using this dynamic RCB partitioner in Zoltan, the Contact Surface

object noted in Section 2.6 can invoke RCB partitioning on the DCR computational domain. Figure 2-11 illustrates the result of RCB partitioning on the domain with 10 processors.

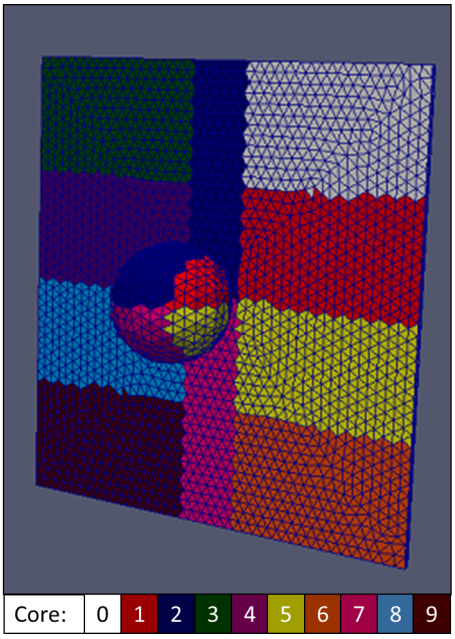


Figure 2-11: Contact DCR Domain Partitioning by Zoltan RCB Dynamic Partitioning

As noted, the RCB partitions form a series of orthogonal, hyperrectangular domains in Cartesian space. With this structure, partitions in the DCR contact domain need only communicate with those partitions that bound its perimeter. Additionally, as the Zoltan implementation of the RCB partitioning scheme is dynamic, this procedure will be repeated as the simulation progresses, ensuring that even as the domain deforms, communication at partition boundaries will remain geometrically relevant.

The coupled contact-mechanics model was developed to support both static equal-element partitioning and dynamic RCB partitioning through Zoltan. To enable parallel communication, the Open Message Passing Interface (Open MPI) [79] library was employed, and used to gather and distribute partition data structures between the structural partitions and the contact partitions. Figure 2-12 illustrates the up-

dated communication procedure used in one PEC integration iteration of the coupled contact-mechanics model.

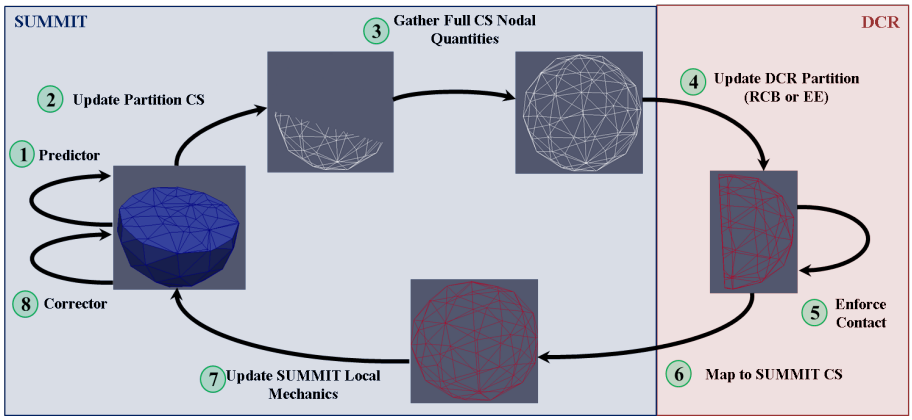


Figure 2-12: Contact-Mechanics PEC Algorithm Procedure in Parallel with MPI - relative to the Serial Procedure, step 3 is new and step 6 is modified to enable MPI gathering across the partitions

For the partitioning of the Contact Surface to be successful, the Contact Surface object must maintain a concept of the global contact surface geometry. This requirement to map to the global surface is due to the fact that structural mechanics has already partitioned the domain using the equal-elements graph-based method, and the concept of the global outer boundary has been lost. Enforcing contact across a structural mechanics partition boundary would be erroneous if the boundary was interior to a continuous geometric entity. Thus, an additional "reconstruction" step must be taken prior to attempting to implement either equal-elements (EE) or RCB partitioning on the contact domain. Note that this reconstruction step is not a part of the iterative loop illustrated in Figure 2-12, as it must only be performed once, prior to iteration, and is used to establish the mapping required for Step 3 of Figure 2-12. This reconstruction step uses Open MPI communication to gather all of the nodal quantities across the structural partitions which were contained within the original outer boundary object, especially the element connectivity. This allows for the origi-

nal outer boundary to be rebuilt, and provided to the contact domain for partitioning. As of now, this globally reconstructed surface acts as the means of communication between node X in the contact domain and node X in the structural mechanics domain, but alternative architectures can and should be considered, as mentioned in Section 4.2. A similar procedure has been applied previously for dual parallelization of structural mechanics and contact mechanics domains as described by Cirak et. al. [70]. However, this prior implementation maintained the mapping between the structural partition and the contact partition at a global scope level, whereas in this implementation, the partition mapping is owned by the Contact Surface of a given structural partition. This distinction does not impact the effect of parallelization of the CCM, but it is relevant to consider for the extensibility of the method. By maintaining the mapping between the SUMMIT structural domain and the DCR contact domain within the Contact Surface entity of each partition, two primary advantages are gained. First, it permits any structural mechanics framework to interact with the Contact Surface and the DCR domain, without need for globally-available data structures, which is ideal for protecting data of classes in general. Secondly, and more importantly, it allows the methods of the Contact Surface to be developed to minimize MPI communications to only the data structures required for both SUMMIT and DCR, while also minimizing memory consumption. However, at this point, a means of implementing the coupled contact-mechanics model in parallel computation with independent domain partitions is available. To determine the effectiveness and capability of this parallelization, a parallel scaling study was performed.

2.7.3 Parallel Scaling Capability

To verify that this scheme is effective for balancing the computational evenly amongst processors, and that this method will scale consistently with higher processor counts, a parallel scaling study was performed. In this study, a sphere and plate contact problem, similar to that used in Section 2.7.1 was discretized and refined using tetrahedral subdivision to increase the computational complexity. From this baseline case, the simulation was run in serial to establish a baseline, and then run again in parallel

with increasingly higher processor counts. The computation time required for each iteration was tracked and averaged across the entire run. Figure 2-13 illustrates the CAD model used for generation of the finite discretization of the system.

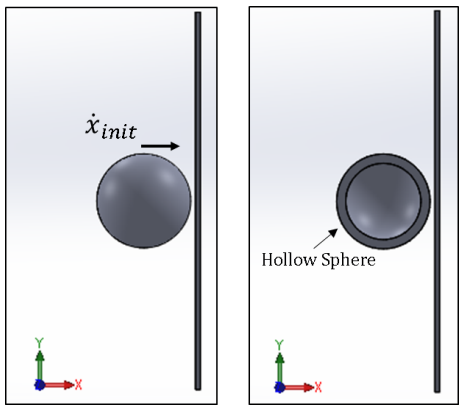


Figure 2-13: Hollow Sphere, Plate Contact Problem for Parallel Scaling Study

To briefly describe the model, the sphere is hollow, with an outer radius of $r_o = 0.00635$ m (0.25 in) and inner radius of $r_i = 0.00508$ m (0.2 in). The thin plate has a length of 0.0508 m (2.0 in) and a width of 0.000648 m (0.0255 in). The sphere was prescribed an initial velocity of $\dot{x}_{init} = 0.0254$ m/s (1.0 in/s), with a total gap between it and the plate of $d_{sep} = 0.00064$ m (0.254 in). Finally, the sphere was prescribed material properties of $E = 1000$ Pa, $\nu = 0.3$, and $\rho = 1000$ kg/m^3, while the plate was prescribed $E = 60000$ Pa, $\nu = 0.3$, and $\rho = 7000$ kg/m^3. While not inherently germane to the parallel scaling study, these values were selected to enable full transient simulations run in serial computation to complete within a reasonable period of time. Beyond this, the selected Young's Modulus is on the order of magnitude of human epidermis [45, 80], and serves as a suitable benchmark point for the purposes of the CCM model. The model was run for a total time of $t_f = 0.2$ s.

For this model, the domain was initially meshed simplicially using Solidworks. From there, the mesh was provided to the coupled contact-mechanics model in SUMMIT, and refined using tetrahedral subdivision one time. After this refinement step,

the total number of tetrahedral elements was approximately 25,000, while the total number of contact surface triangular elements was approximately 10,000. With the domain discretized and pre-processed, the simulation was run with a variable number of processors in parallel, and the average iteration time for both structural mechanics and contact mechanics tracked. These iteration times were averaged across each simulation, and plotted against the number of processors used to determine the parallel scalability of both structural mechanics and contact mechanics components within the model. Figure 2-14 illustrates the results from this parallel scaling study.

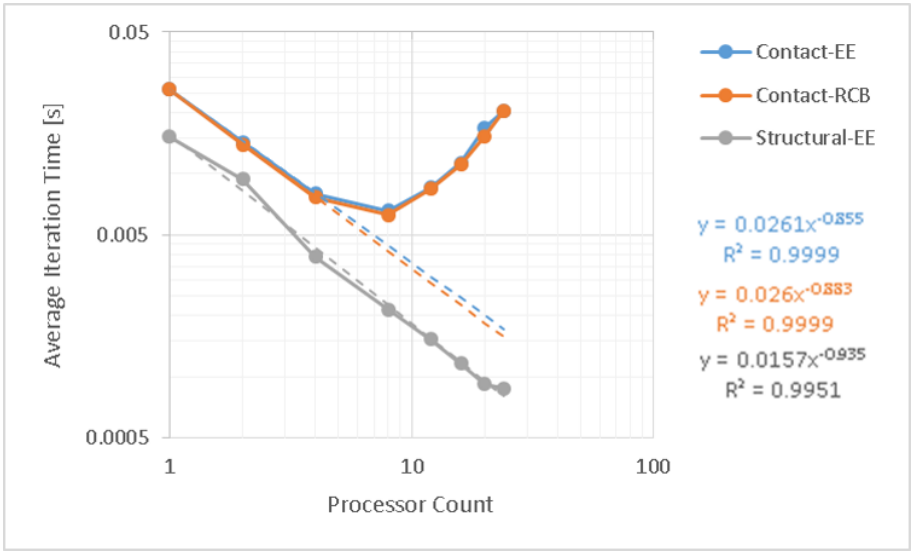


Figure 2-14: Parallel Scaling Study Results for Contact-Mechanics Model Components, with Contact Mechanics Iteration Time using Equal-Elements (EE) Static Partitioning [blue], Contact Mechanics Iteration Time using Recursive Coordinate Bisectioning (RCB) Dynamic Partitioning [orange], and Structural Mechanics Iteration Time using Equal-Elements (EE) Static Partitioning [grey]

From the results presented in Figure 2-14, a few conclusions can be drawn. First, the SUMMIT structural mechanics system scales strongly for the processor counts considered in this analysis, maintaining a monomial relationship of iteration time to processor count of $t_{struct_{iter}} = 0.0157 N_{procs}^{-0.935}$, representing an 93.5% scaling effectiveness. The results for contact mechanics include scaling results using the static

equal-elements partitioning scheme and the dynamic RCB partitioning scheme. It can be seen that initially, both methods scale in a similar monomial fashion as structural mechanics, with $t_{contact_{EE}} = 0.0261 N_{procs}^{-0.855}$ and $t_{contact_{RCB}} = 0.026 N_{procs}^{-0.883}$, highlighting a 85.5% and 88.3% scaling effectiveness for both schemes, respectively, for up to 4 processors. However, beyond this point, the improvements gained in parallelization begin to reach a point of diminishing returns, and eventually reverse beyond 10 processors. This reversal suggests that the parallelization scheme employed is currently suboptimal for data structure communication, as increasing processor count is creating an increase in processor work. This is unfortunate, but not entirely unexpected - the hypothesized cause and potential mitigation strategies are discussed in detail in Section 4.2. However, while the scalability response beyond 10 processors is not ideal for the contact mechanics component, for processor counts below this, the scaling is effective. This parallel capability enables a total runtime reduction that is nearly perfectly inversely proportional to the processor count. Additionally, it should be noted that for this case, the dynamic RCB partitioning scheme offered slight benefit over the static equal elements partitioning scheme, with a difference in effectiveness of approximately 3%. This benefit is not immense, but with a more complex geometric domain, the RCB partitioning scheme would be expected to offer further improvement over the equal elements method, as the graph-based partitioning scheme is effective for a simple test case such as this.

While the parallel scalability of the contact model is limited to $<$10 processors before diminishing returns set in, significant advantage is still gained by the extensibility into parallel computing. The transition from serial computation to parallel computation with four processors results in a runtime reduction of nearly 72%, significantly increasing the usability of this model.

2.8 Trade Comparison against Simplified Rigid-Body Approaches

While this coupled contact-mechanics model has been developed to enable modeling of interactions between two deformable entities with abstract geometries and varying levels of fidelity, this level of complexity may not always be necessary for the end user. Human epidermis, which has a Young's Modulus on the order of 10^{-1} MPa or lower [45,80] , appears highly deformable relative to a rigid EVA suit material such as fiberglass, which has a Young's Modulus on the order of GPa [81]. Similarly, a human surrogate model may appear rigid relative to a highly deformable softgood suit model, depending on the suit orientation and pressurization level. Due to this potential for relatively disparate ratios of effective Young's moduli between human and suit, it is not unreasonable to consider neglecting the deformation effects of the stiffer material and instead implement a simpler rigid-body approximation for resolving human-suit interactions. Such an approach would eliminate the finite element approximation of the stiffer entity, providing a benefit in terms of computational complexity, at the expense of both accuracy and a loss of internal energy data within that entity. However, as a design tool, it may be beneficial to obtain simulation results in a more expedient fashion during conceptual and preliminary design phases of an EVA suit, at the expense of some accuracy. Understanding the nature of this trade between computational complexity and accuracy is key to determine regimes in which the use of this coupled contact-mechanics model is necessary, and where a rigid body approximation would not be unreasonable.

To this end, a parametric study was developed and performed across a grid of normalized model parameters to identify the trade between computational complexity and accuracy for this model versus a simple-rigid body model in elastic collisions. The degrees of freedom identified for variation include the relative Young's Moduli, the relative length scales, and the relative kinetic energies of the impactor and artifact. A simple simulation, similar to those presented in Sections 2.7.1 and 2.7.3, was proposed as illustrated in Figure 2-15.

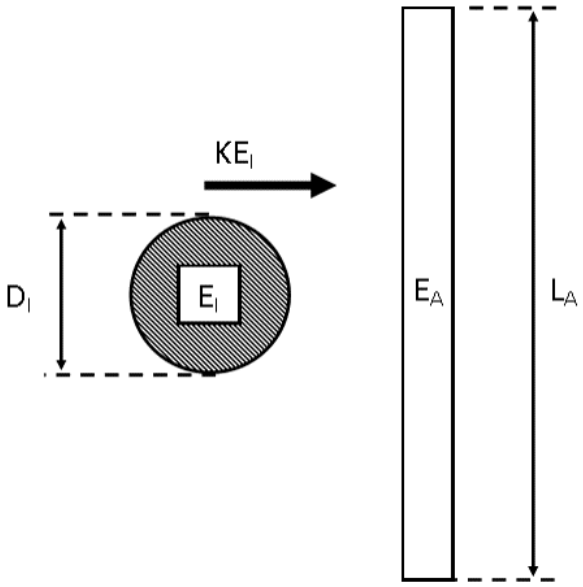


Figure 2-15: Trade Study Model with Primary Variable Degrees of Freedom

From Figure 2-15, the degrees of freedom to be parametrically varied are the Young's Modulus Ratio (ER), given by

$$ER = \frac{E_I}{E_A} \tag{2.32}$$

the Relative Length Scale (RLS), given by

$$RLS = \frac{D_I}{L_A} \tag{2.33}$$

and the impactor kinetic energy, KE_I. Table 2.2 highlights the grid points across which all combinations of these variables were simulated.

Variable	**Grid Points**
ER	10^0, 10^2, 10^4, 10^6
RLS	0.5, 1.0, 1.5
KE_I $[J]$	10, 50, 100

Table 2.2: CCM Model Parametric Trade Study Test Points

The intention and selection of these grid variables is aimed at providing a sensitivity mapping between accuracy and cost at a given simulation operating point. The ER selected span from equally deformable impactor and artifact (10^0), up to an apparently rigid impactor as seen by the artifact (10^6). Lower ratios were not considered, as in implementation, the rigid body approximation should always be applied to the more rigid entity. The RLS ranges are intended to highlight cases where the impactor is smaller, equivalent, or larger in size than the artifact, without delving into order of magnitude size separations as the finite element mesh refinement level would become an unintended dependent variable of the system. Finally, the kinetic energy values were selected to capture sensitivity to variable energy impactors. To provide constants by which to anchor these simulations, the artifact was prescribed a Young's Modulus of $E_A = 1000\ Pa$ and $\rho_A = 1000\ kg/m^3$. The impactor was also prescribed an identical density, and both impactor and artifact were given a Poisson's ratio of $\nu = 0.49$. Furthermore, the distance of separation between the impactor and artifact was explicitly calculated to ensure contact would occur at $t_c = 0.1\ s$, and the total simulation was run for $t_f = 0.15\ s$.

To simulate rigid body dynamics, the pre-existing rigid body contact methods employed in SUMMIT were used. These methods resolve a given impactor entity as a purely geometrical abstraction, with no finite element discretization representation. This rigid body contact formulation has been previously employed in ballistic simulations [55]. In this trade study, a sphere geometry was modeled and prescribed a given kinetic energy based on a constant material density, volume, and velocity vector. From this, the displacement and velocity of the rigid sphere geometry can be tracked with a negligible computational cost relative to resolving an equivalent sphere composed of tetrahedral finite elements. By comparison, the CCM model does fully resolve the impactor sphere in tetrahedral finite elements, and prescribes the stated material characteristics and velocity conditions to the sphere mesh accordingly.

To enforce the contact constraint across the rigid sphere entity in each iteration, all quadrature points in the artifact are checked for radial proximity from the center of the rigid sphere. If the radial distance is less than that of the rigid sphere's prescribed

radius, that distance of intersection is computed and a penalty method as described in Section 2.4.1 is enforced in a similar fashion to Equation 2.5. The contact penalty parameter, K_P was set to 10^6 for this study, as lower orders of magnitude notably did not preserve the contact constraint and higher values resulted in numerical instability, an unfortunate detriment of penalty methods [60]. Finally, it should be stressed that all cases were run in parallel using 10 processors, for both the CCM model and the rigid body model.

First, the total computational runtime, and the average iteration computation time, for each combination of variables listed in Table 2.2 was tracked and compared for both the CCM model and the rigid-body model. Figure 2-16 illustrates the total runtime for each case, while Figure 2-17 illustrates the average iteration computation time.

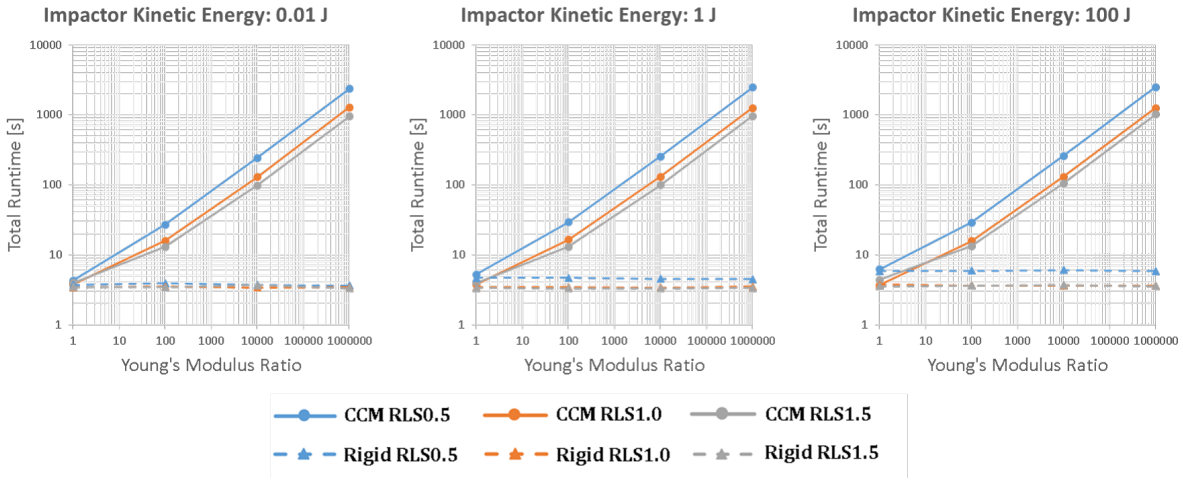


Figure 2-16: Total Computational Runtime for Trade Study Cases

The average iteration runtime shown in Figure 2-17 only serves to highlight that the average iteration time for the CCM was orders of magnitude lower than for the rigid body model, and that the variables had little appreciable effect on the average iteration time.

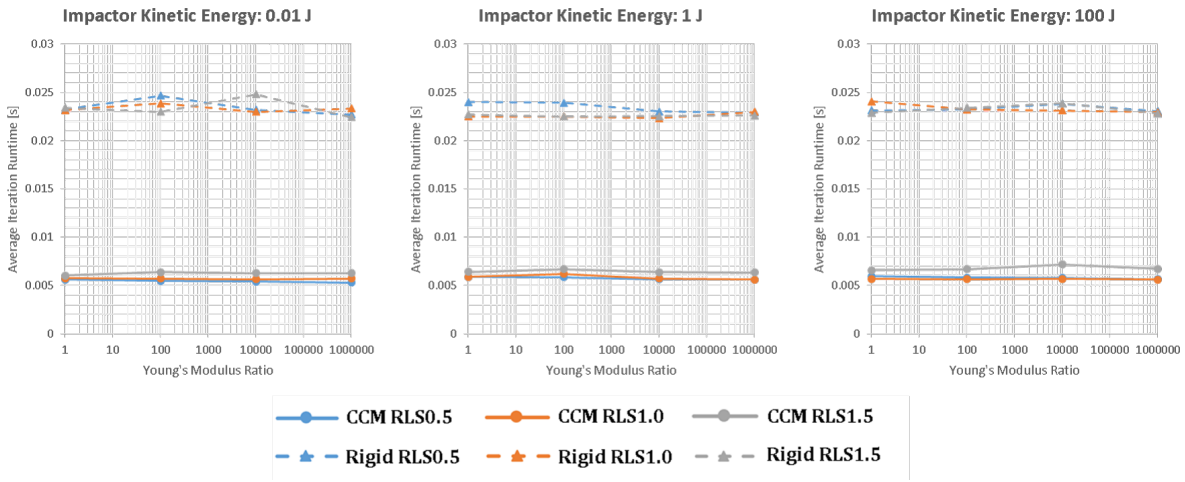


Figure 2-17: Average Iteration Computation Time for Trade Study Cases

The runtime results in Figure 2-16, however, shows the strong computational cost associated with use of the CCM versus the rigid body model. As ER decreases, the CCM total runtime decreases monomially until it converges to an effectively similar time as the rigid body model. However, as ER increases and the CCM total runtime increases significantly - up to 400 times more costly than the equivalent case using the rigid body model. This increase in computational cost is due to the need to resolve the sphere impactor with an increasing stiffness. This increased stiffness causes the minimum stable time step for the system to decrease equivalently, shown in Figure 2-18.

From these timing results, it can be seen that the CCM model must pay a notably higher computational price to resolve the mechanics associated with a rigid impactor object. However, as the CCM model is resolving the structural mechanics of the impactor as well, it can provide a more accurate representation of the system response for both impactor and artifact, especially as the impactor becomes less rigid. Essentially, as the ER increases, the accuracy of the rigid body relative to the CCM model should be expected to increase, as the rigid body assumption becomes increasingly valid with higher ER. In contrast, as ER decreases, it would be expected that the CCM model should capture a reduced transfer of kinetic energy to the artifact, while the rigid body introduces an incorrectly high amount of kinetic energy. To identify

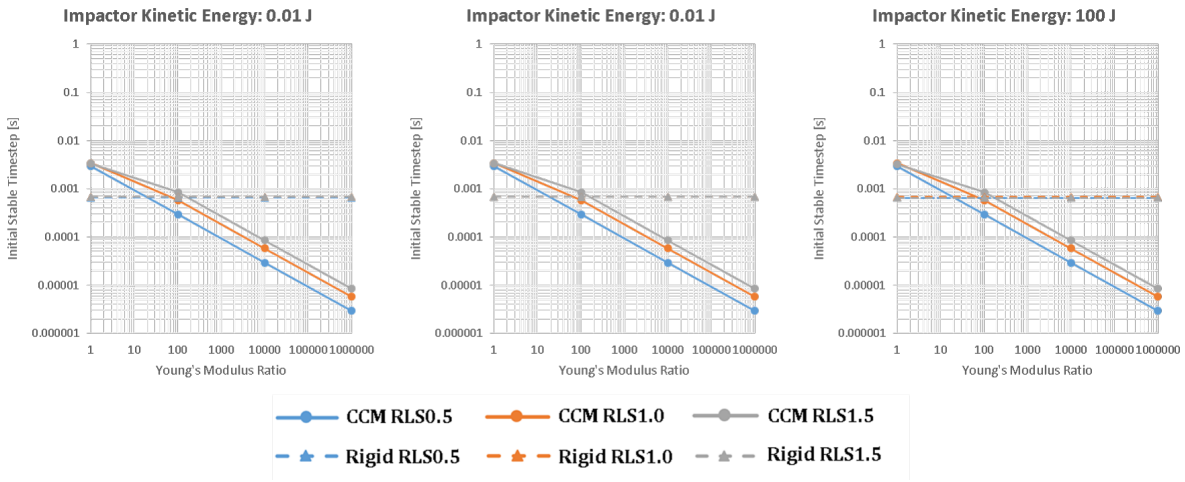


Figure 2-18: Initial Minimum Stable Time Step for Trade Study Cases

this effective accuracy impact, a handful of cases from the grid were interrogated and compared on a back-to-back basis. To provide insight into a few corner points, a case with high ER and a case with low ER were interrogated to observe differences in the transient kinetic energy of the artifact, as illustrated in Figure 2-19.

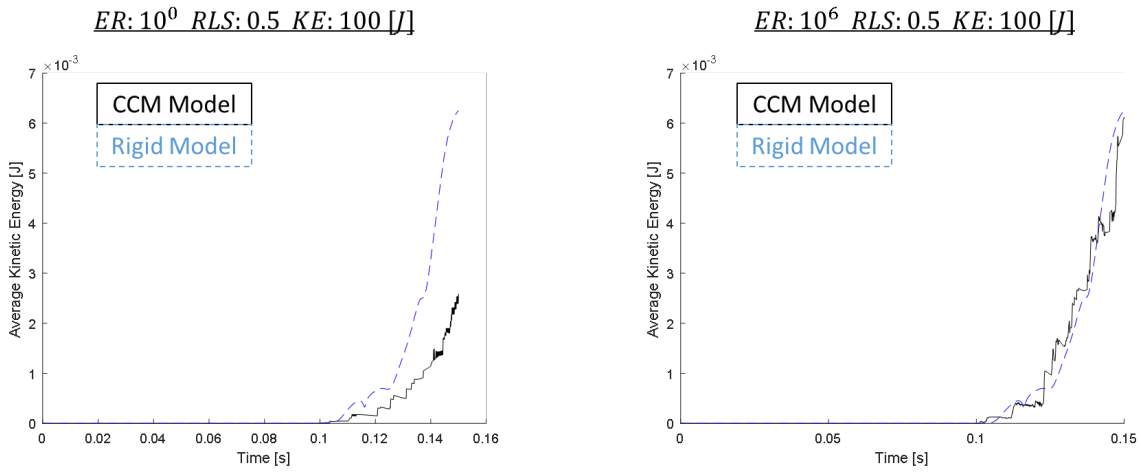


Figure 2-19: Comparison of CCM and Rigid Body Model Transient Results for Artifact Average Kinetic Energy with $t_c = 0.1\ s$

As posed previously, with high ER, the rigid body model and CCM model agree relatively well throughout the transient, as high ER makes the impactor appear

significantly more rigid relative to the artifact, which lends itself to the rigid body assumption. However, for low ER, the rigid body model response does not change, while the CCM model predicts notably lower imparted kinetic energy to the artifact, specifically 59.2% lower than the rigid body model. For an ER of 1.0, the impactor and artifact share nearly identical rigidities, and as a result, the impactor should absorb kinetic energy during contact into deformation, reducing the kinetic energy imparted to the artifact. Furthermore, the equivalent rigidities should be expected to yield an erroneous result for a rigid body assumption, where the impactor cannot deform. Highlighting this, Figures 2-20 illustrates the deformation of the artifact at the end of the simulation for both cases with $ER = 10^6$ and $RLS = 0.5$, while Figure 2-21 illustrates the deformation of the artifact at the end of the simulation with $ER = 10^0$ and $RLS = 0.5$.

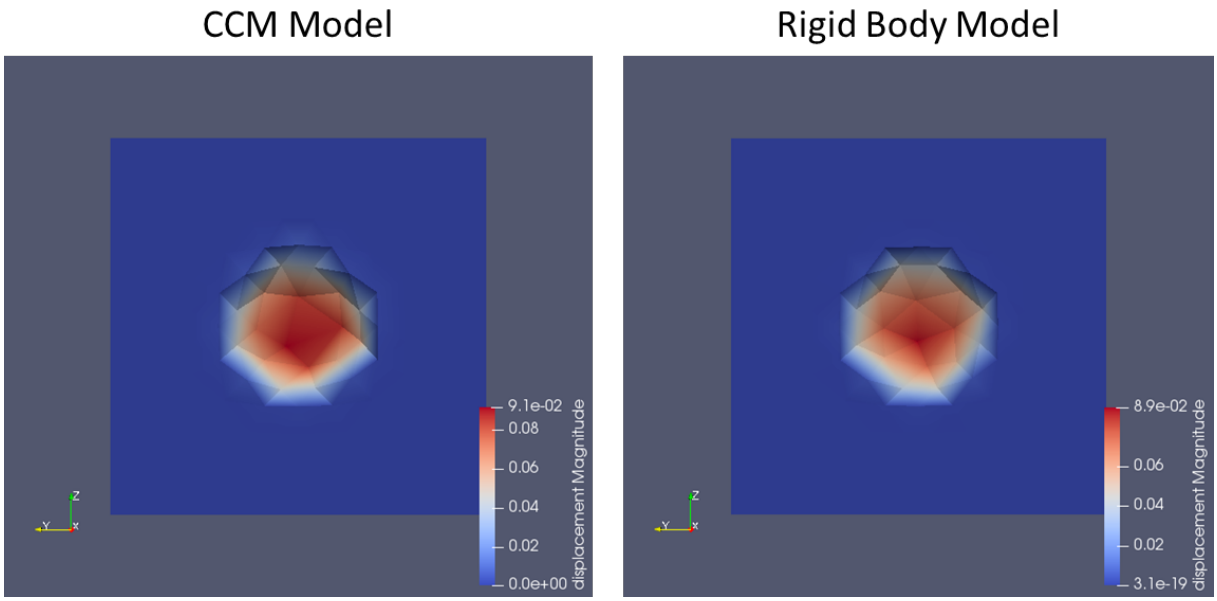


Figure 2-20: Comparison of $ER:\ 10^6$, $RLS:\ 0.5$, $KE: 100\left[J\right]$ Deformation

As expected, for the high ER case, the response of the CCM model and the rigid body model are very similar in magnitude and form, but for the low ER case, the CCM

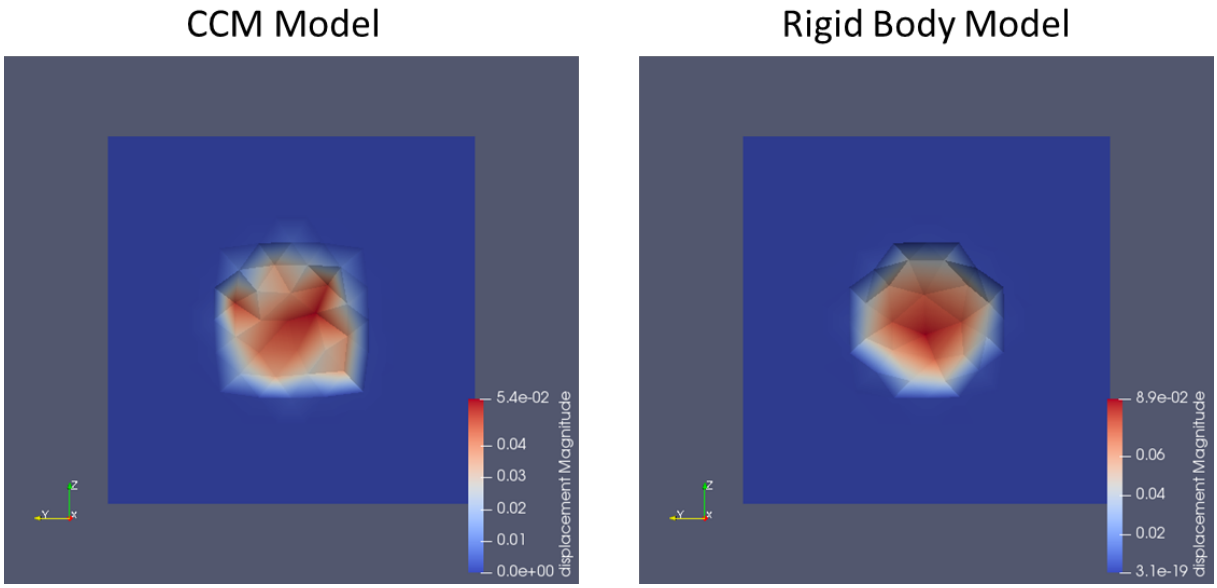


Figure 2-21: Comparison of $piER:\ 10^0$, $RLS:\ 0.5$, $KE: 100\big[J\big]$ Deformation

model predicts 35.2% less average deformation and 39.1% less maximum deformation of the artifact than the rigid body model. This reduced deformation makes physical sense, as the impactor is deforming significantly for low ER, reducing the total amount of deformation of the artifact, while also increasing the area of deformation on the artifact as the impactor "flattens."

Following interrogation of a few corner points to identify the validity of the model trends for both CCM and rigid body, it is necessary to identify the effective accuracy differences between the model on a macroscopic level. To this end, several macroscopic kinematic variables of the artifact object were tracked for both the CCM model and the rigid body model. Namely, the average nodal displacement, the maximum nodal displacement, and the average kinetic energy of the artifact were tracked. To quantify the error between the CCM model results and the rigid body model results, a cumulative root sum of squares (RSS) of the delta between these values was tracked across the simulation and normalized against the average value for the simulation. The normalized error for the average nodal displacement is calculated by

$$\tilde{x}_{avg} = \frac{\Sigma_i^N (x_{c_{avg_i}} - x_{r_{avg_i}})^2}{\frac{\bar{x}_{c_{avg}} + \bar{x}_{r_{avg}}}{2}} \tag{2.34}$$

where $x_{c_{avg_i}}$ is the average nodal displacement for the CCM model at iteration i, $x_{r_{avg_i}}$ is the average nodal displacement for the rigid body model at iteration i, $\bar{x}_{c_{avg}}$ is the average nodal displacement for the CCM model averaged over all iterations, and $\bar{x}_{r_{avg}}$ is the average nodal displacement for the rigid body model averaged over all iterations. The normalized error for the maximum nodal displacement is calculated by

$$\tilde{x}_{max} = \frac{\Sigma_i^N (x_{c_{max_i}} - x_{r_{max_i}})^2}{\frac{\bar{x}_{c_{max}} + \bar{x}_{r_{max}}}{2}} \tag{2.35}$$

where $x_{c_{max_i}}$ is the maximum nodal displacement for the CCM model at iteration i, $x_{r_{max_i}}$ is the maximum nodal displacement for the rigid body model at iteration i, $\bar{x}_{c_{max}}$ is the maximum nodal displacement for the CCM model averaged over all iterations, and $\bar{x}_{r_{max}}$ is the maximum nodal displacement for the rigid body model averaged over all iterations. Finally, the normalized error for the average kinetic energy of the artifact is calculated by

$$\tilde{KE}_{avg} = \frac{\Sigma_i^N (KE_{c_{avg_i}} - KE_{r_{avg_i}})^2}{\frac{\bar{KE}_{c_{max}} + \bar{KE}_{r_{max}}}{2}} \tag{2.36}$$

where $KE_{c_{avg_i}}$ is the average kinetic energy of the artifact for the CCM model at iteration i, $KE_{r_{avg_i}}$ is the average kinetic energy of the artifact for the rigid body model at iteration i, $\bar{KE}_{c_{avg}}$ is the average kinetic energy of the artifact for the CCM model averaged over all iterations, and $\bar{KE}_{r_{avg}}$ is the average kinetic energy of the artifact for the rigid body model averaged over all iterations. These normalized errors are shown in Figures 2-22, 2-23, and 2-24.

From Figure 2-22, for higher RLS and kinetic energies, the normalized average displacement error tends to increase as ER decreases. However, for lower RLS and kinetic energies, this response is inconsistent. This inconsistency is unexpected, as for lower ER in all cases, it would be expected that the rigid body approximation would overpredict the displacement imparted to the artifact, as no internal energy can be retained within the rigid sphere. However, as for smaller RLS, a higher proportion of impactor nodes would not experience displacement, this could artificially reduce the

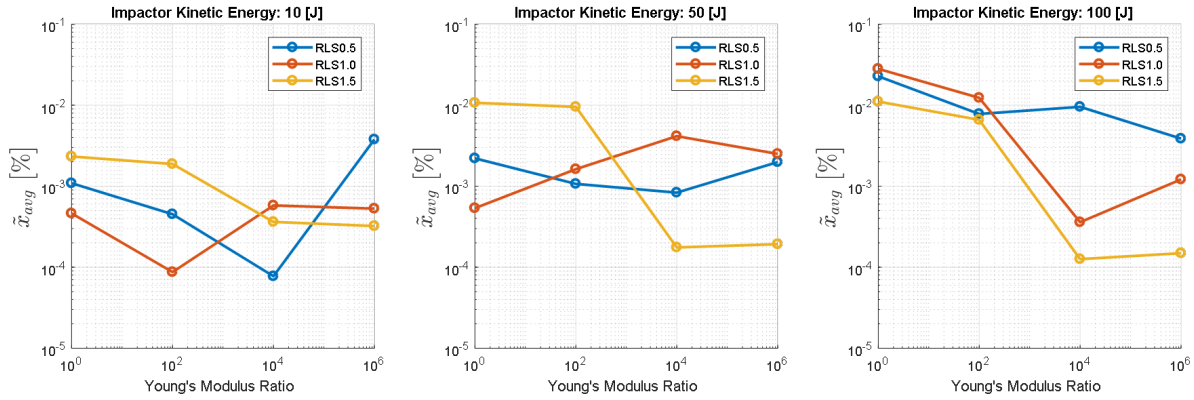


Figure 2-22: Normalized Error in Average Displacement for All Cases

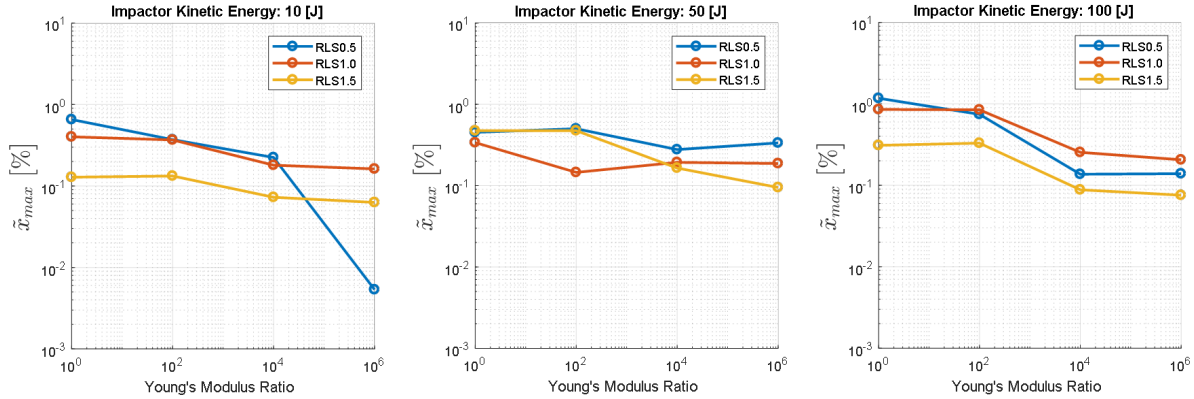


Figure 2-23: Normalized Error in Max Displacement for All Cases

apparent impact of the methods. The maximum displacement of the artifact, however, should not be subject to this artificial reduction. Observation of the trends in Figure 2-23 reflects this expectation. As ER decreases, the normalized error between the CCM method, which can resolve deformations of the impactor sphere, and the rigid body method, which cannot deform the impactor sphere, notably increases across all cases. Furthermore, as the kinetic energy of the impactor sphere increases, the magnitude of the error increases as well. This response is to be expected, as for less rigid impactor spheres, the CCM model predicts greater deformation, and therefore imparts less kinetic energy to the artifact. To verify this, the results shown in Figure 2-24 highlight the normalized error differences in the average kinetic energy of the artifact. While not entirely consistent for all RLS, for decreasing ER, the normalized error tends to increase. More notably, however, is that as the kinetic energy of the

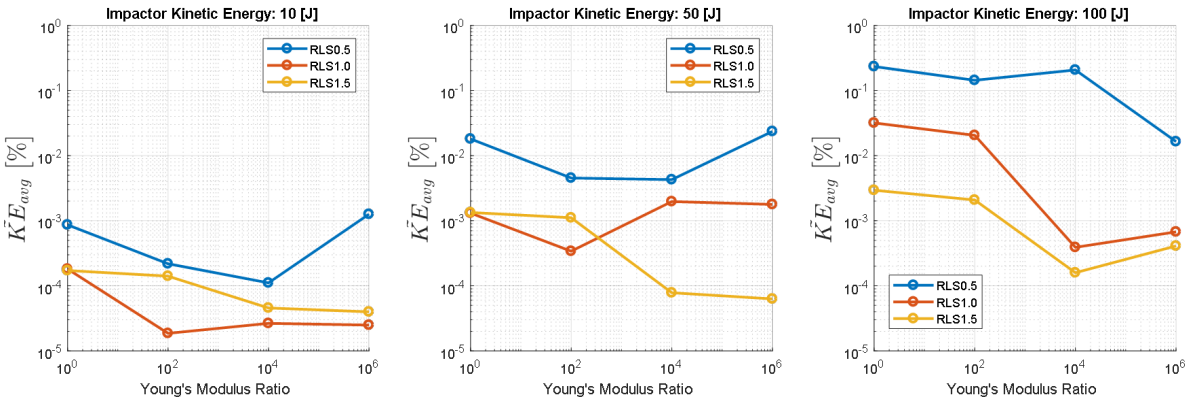


Figure 2-24: Normalized Error in Average Kinetic Energy for All Cases

impactor sphere increases, the normalized error also increases.

While the normalized, cumulative RSS errors may provide an indication of simulation differences across the entire transient, they suggest the differences between the model is quite small even at low ER. However, from the results shown in Figure 2-21, it was seen that a difference on the order of 35% had developed by the end of the simulation. As the state of the artifact at the end of the simulation should highlight the maximum effect of the CCM model versus the rigid body model, three additional normalized error measurements are taken at the end of the simulation, namely

$$\tilde{x}_{avg}(t_{final}) = \frac{|x_{c_{avg_i}}(t_{final}) - x_{r_{avg_i}}(t_{final})|}{max(max(|x_{c_{avg}}|_i), max(|x_{r_{avg}}|_i))}, \tag{2.37}$$

$$\tilde{x}_{max}(t_{final}) = \frac{|x_{c_{max_i}}(t_{final}) - x_{r_{max_i}}(t_{final})|}{max(max(|x_{c_{max}}|_i), max(|x_{r_{max}}|_i))}, \tag{2.38}$$

and

$$\tilde{KE}_{avg}(t_{final}) = \frac{|KE_{c_{avg_i}}(t_{final}) - KE_{r_{avg_i}}(t_{final})|}{max(max(|KE_{c_{avg}}|_i), max(|KE_{r_{avg}}|_i))} \tag{2.39}$$

which represent the absolute error between the CCM model and the rigid body model at the end of the simulation (noted by t_{final}), normalized by the maximum absolute value of average displacement, maximum displacement, and average kinetic energy, respectively. These normalized errors for the final state of the simulation are

shown in Figures 2-25, 2-26, and 2-27.

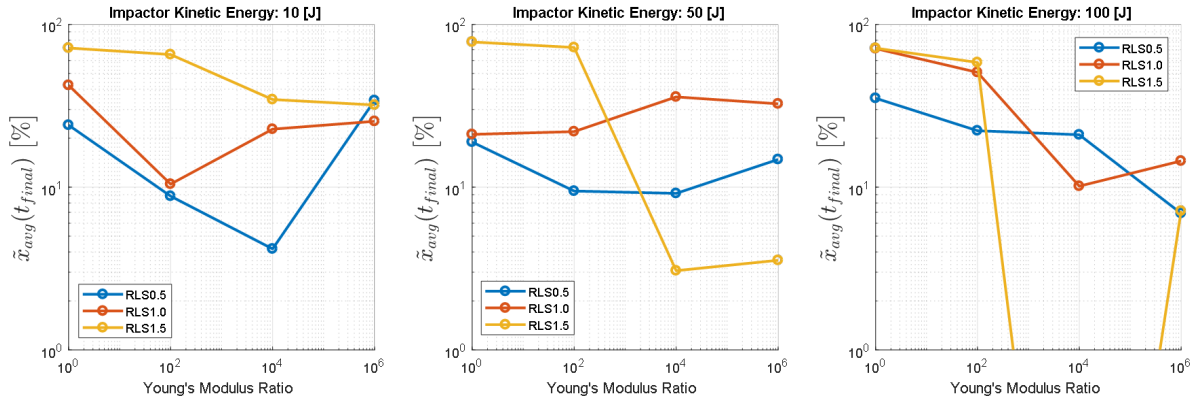


Figure 2-25: Normalized Error in Average Displacement at t_{final} for All Cases

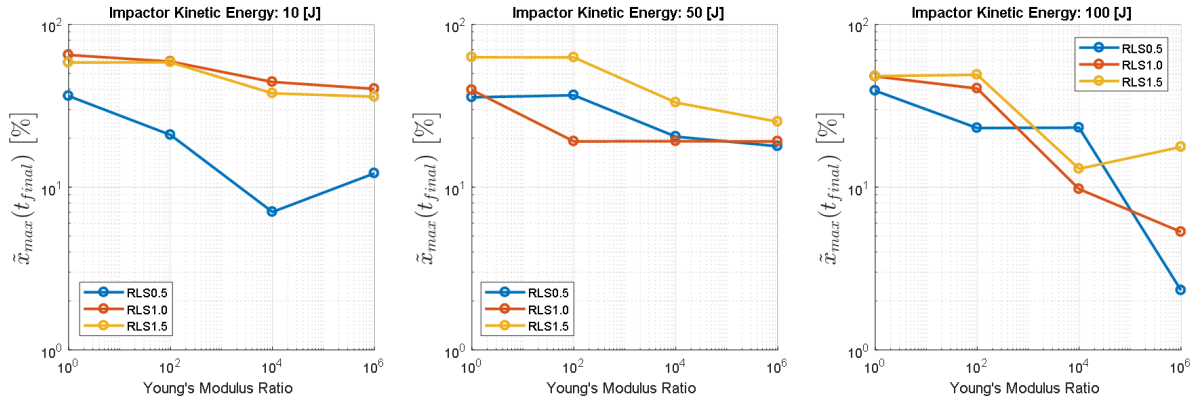


Figure 2-26: Normalized Error in Max Displacement at t_{final} for All Cases

Similar to the normalized, cumulative RSS errors, the average displacement of the artifact at the end of the simulation does not always show a consistent trend across all ER ranges. However, in general, the relative difference in the final average displacement of the artifact between the CCM model and the rigid body model tends to increase with decreasing ER. The maximum displacement of the artifact at the

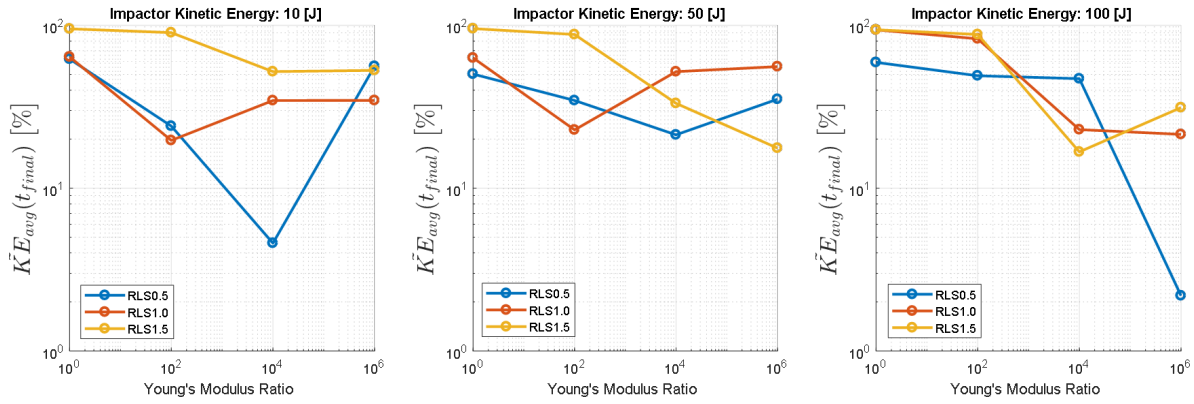


Figure 2-27: Normalized Error in Average Kinetic Energy at t_{final} for All Cases

end of the simulation, however, is noticeably more consistent, with all cases increasing in relative error as ER decreases. Similarly, with some exceptions at low impactor kinetic energies, the average kinetic energy of the impactor at the end of the simulation has higher relative error at lower ER as well. All of these trends corroborate those observed in the normalized, cumulative RSS errors across the entire transient, but an important distinction should be drawn in that the magnitude of the error at the end of the transient is orders of magnitude higher. The difference between the CCM model and the rigid body model is greater than 90% for average kinetic energy at ER of 1.0 and a RLS of 1.5. This apparent difference between the models at low ER agrees with the individual case assessment illustrated in Figures 2-20 and 2-21, where for high ER, the models were nearly equivalent, but for low ER, the differences were on the order of 10^1%.

To aid the end user in determining when to use the CCM model for improved fidelity and to capture deformations associated with the impactor object, the normalized errors and runtime data were collected and collapsed into a set of normalized metrics. First, a means of comparing the runtime expense of the CCM model versus a simplified rigid-body model was developed, referred to as the Runtime Ratio (RTR), given by

$$RTR = \frac{NT_{CCM}}{NT_{Rigid}} \tag{2.40}$$

where

$$NT_{CCM} = \frac{AIR_{CCM}}{ISTS_{CCM}} \tag{2.41}$$

and

$$NT_{Rigid} = \frac{AIR_{Rigid}}{ISTS_{Rigid}} \tag{2.42}$$

Here, AIR refers to the Average Iteration Runtime in seconds and $ISTS$ is the Initial Stable Time Step for a given model, denoted in the subscript where CCM is the CCM model and $Rigid$ is the rigid-body model. The ratio of AIR to the $ISTS$ represents the Normalized Runtime, or NT, for a given model such that the total runtime, TR, can be expressed by

$$TR_x = T_{final} NT_x \tag{2.43}$$

since the average time to compute one iteration, divided by the initial stable timestep provides a rough estimate of the total time required to compute all of the iterations up to T_{final}, the physical end time for a given simulation. Using these Normalized Runtime parameters, the relative cost of the CCM model to the rigid body model is expressed by the Runtime Ratio, RTR. For reference, a comparison of the NT for the CCM and rigid-body models are shown in Figure 2-28.

While Figure 2-28 does not provide immediate data for the end user beyond a general guideline for the runtime associated either the CCM or rigid body models, it is meant to provide a means to interpret error-cost data. Additionally, the nearly identical normalized runtimes in each kinetic energy case is expected and correct; as the initial stable time step can be expressed by $ISTS = f(ER, KE)$ for equal element sizes, normalizing the iteration time by the stable time step will effectively normalize the results by kinetic energy as well.

To provide a single measure by which the relative error to runtime trade can be considered for a given simulation, a new set of trade metrics are developed. Normalized error terms for the error between the CCM model and the rigid body model at

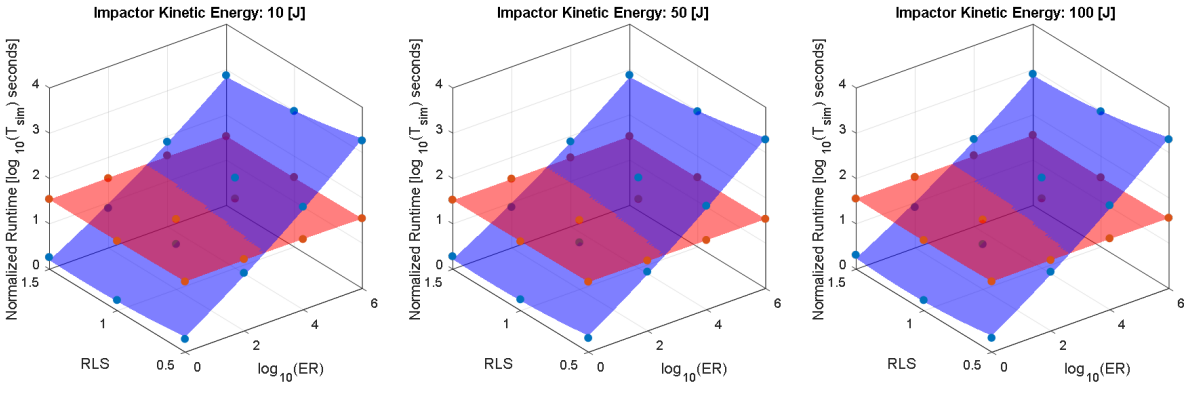


Figure 2-28: Normalized Runtime, NT, for CCM and Rigid Body Models versus ER, RLS, and Impactor KE, with NT_{CCM} surface fit (blue) and data (blue dots), and NT_{Rigid} surface fit (red) and data (red dots)

the end of the simulation were computed by Equations 2.37, 2.38, and 2.39. These normalized errors provided a means to evaluate how much relative error could be expected resulting from the use of the rigid body model over the CCM model. To collapse this with runtime expense, each of these normalized errors were divided by the RTR for that simulation. These metrics form terms that represent a $\frac{relative\ error}{RTR}$, yielding a weighting function that penalizes high relative error and low Runtime Ratio. Figure 2-29 illustrates this error/cost metric for the average kinetic energy error at the end of the simulation, versus the Runtime Ratio between the CCM and rigid body models.

As the weighting function increases in value, it implies that either the relative error has increased, the Runtime Ratio has decreased, or both. Therefore, for high values of the above weighting functions, the CCM model should be used, and for low values, the rigid body model should be used, to maximize the effectiveness of error-to-runtime for the simulation. While the exact threshold beyond which the end user may wish to select the CCM model over a simple rigid body model may differ, these metrics can form a single number by which the CCM model can be recommended over a rigid body approach that the end user can look up before running a simulation. While just a subset of these weighting functions have been provided in Figure 2-29, a full listing of these weighting functions and equations as a function of ER and RLS are

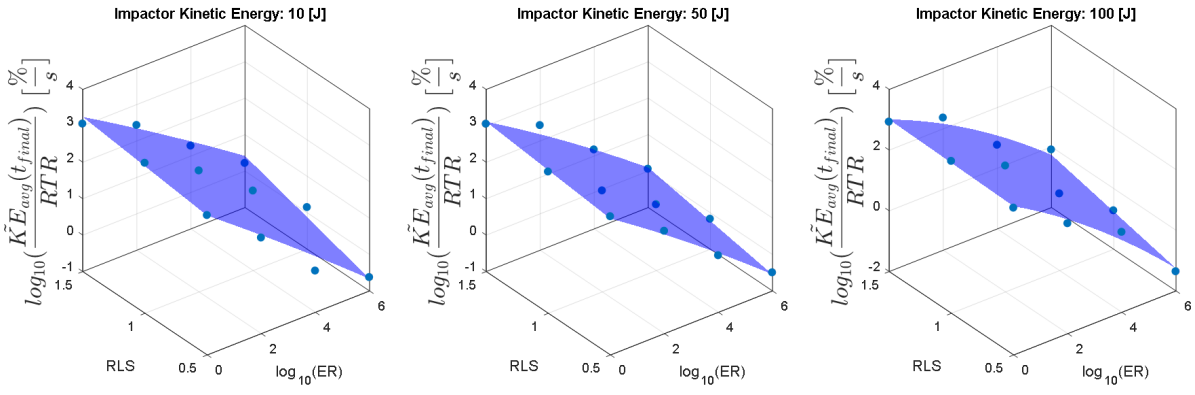


Figure 2-29: Weighted error-to-cost metric: Relative error in average artifact kinetic energy at the end of simulation per Runtime Ratio results, with surface fit (blue surface) and simulation data (blue dots)

provided in Appendix B.

Given the physical results of the models highlighted in Figures 2-20 and 2-21, this reinforces the concept that for simulations involving a deformable impactor and artifact, the CCM model developed here offers improvements in the accuracy of the simulation over a rigid body approximation. However, for cases where one entity is notably stiffer, a rigid body approximation becomes more tenable and comes with a reduced computational cost. For practical implementations, these results provide a framework by which the computational cost versus simulation accuracy trade can be used to identify a model formulation that is most effective for the criticality of that design. Additionally, these results highlight that the trends of the CCM model are correctly aligning with physical expectations. However, proper validation of the CCM model is still necessary, and is addressed in Chapter 3.

Chapter 3

Experimental Validation

The results of the trade study in Section 2.8 highlighted agreement between the CCM model and a rigid-body model when the relative Young's Modulus ratio between impactor and artifact was 10^6 or greater. For these relative stiffnesses, the impactor is effectively a rigid body relative to the artifact and the alignment of the CCM model with a rigid body approximation lends to highlighting the validity of the CCM model to resolve contact events. Additionally, the results suggested that the CCM model was correctly trending in kinetic energy responses for lower Young's Modulus ratios. However, these trends do not provide a means of verifying the capability or accuracy of the CCM model in these lower Young's Modulus ratios.

While contact events between human epidermis and a fiberglass space suit HUT may represent a significantly rigid contact event on the order of 10^{10} Young's Modulus ratio or greater, the interactions between human epidermis and an inflated fabric-based space suit are significantly lower given that the human envelope drives the posture of the suit. Due to the lower rigidity of the pressurized softgood suit, it is necessary to verify the capability and accuracy of the CCM model in capturing contact events in conditions with significantly lower effective stiffness. To this end, an experimental contact study was developed and performed to generate data from which the CCM model could be validated against.

3.1 Overview of Contact Regimes of Interest

While contact events bordering on ballistic impacts could be considered, for this work, the bounds of the contact events should be limited to the domain in which human-suit operation is feasible. Prior EVA mobility studies have indicated a range of velocities for various limbs, such as in wrist and elbow flexion/extension. While human-suit joint kinematics are usually considered in joint or angular space, for contact events, the orthogonal linear velocities between suit and human represent those velocities germane to this study. While not representative of all possible motion profiles, wrist flexion/extension velocities span on the order of 0.15 m/s at the end of the hand, and elbow flexion/extension velocities span on the order of 0.5 m/s in prior kinematic studies of the EMU [15]. Additionally, nominal lunar locomotion velocites are expected to be $\leq$ 1.5 m/s [16]. From these ranges, contact impulse events with impactor velocities spanning the range of 0.25 m/s to 1.0 m/s are of interest for validating the CCM model.

In addition to contact velocities, it is relevant to consider various types of contact events and variable impactor stiffnesses. In terms of contact events, the two primary types considered for this work are those in which the artifact is free to deflect, and those in which the artifact is constrained. For a free artifact, the impactor will primarily drive the displacement of the artifact, while for a fixed artifact, the artifact will primarily drive the displacement of the impactor. For relative stiffnesses, cases in which the impactor is of relatively equivalent stiffness to the artifact and those in which the impactor is of lower should be considered for validation of the CCM model.

Given the noted impact velocities from prior EVA suit studies [15, 16] and the types of contact events to be considered, Table 3.1 represents the regimes of interest for validating the CCM model.

Variable	Domain
Velocity $[m/s]$	$[0.1, 1.0]$
Contact Type	Fixed Artifact, Free Artifact
Impactor Stiffness	Rigid, Flexible

Table 3.1: Contact Regimes of Interest for Experimental Validation of CCM Model

3.2 Experimental Validation Testing

To capture experimental data to validate the CCM model for the regimes noted in Table 3.1, an experimental rig test was developed and executed. While the intent of the CCM model is to accommodate complex, arbitrary systems, but for the sake of validating the model, experimental testing should minimize the degrees of freedom to prevent uncertainty added through unnecessary complexity. One of the simplest forms of contact to resolve is a single-degree of freedom case, where an impactor object contacts an artifact along a fixed trajectory; essentially a drop test. However, the experimental rig needed to have the flexibility to capture a variety of contact types, such as when the impactor is deformable or rigid, or when the artifact can or cannot deflect. With these constraints in mind, a test configuration was proposed as illustrated in Figure 3-1.

An impactor with a prescribed mass M_i and spring constant K_i is driven to contact the artifact at a prescribed velocity V_i. The resultant pressure due to contact, P, is applied to the artifact. The artifact can only rotate in one degree of freedom θ_a, or may be fully constrained. The selection of this architecture provides the flexibility to satisfy all of the contact regimes identified in Section 3.1, with minimal added complexity. For validation, a CAD model of the experimental test rig was developed, discretized and simulated using the CCM model. The primary outputs of V_i, P, and θ_a, were compared on a back-to-back basis with experimental data to provide a means of evaluation the fidelity of the CCM model.

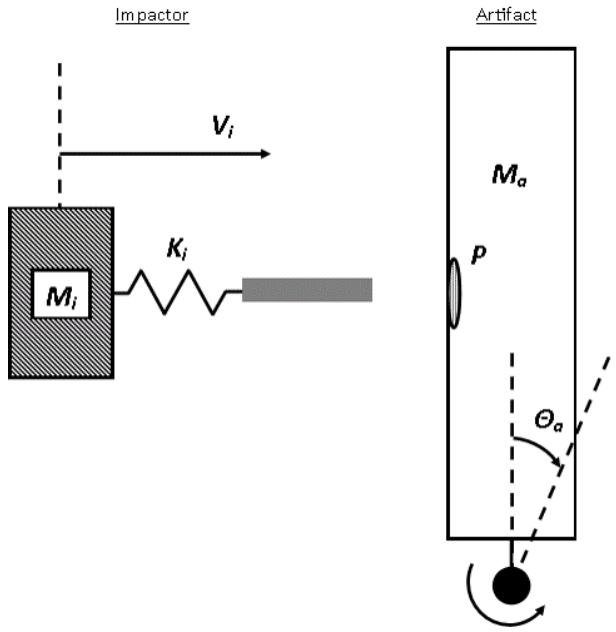


Figure 3-1: Experimental Validation Test Concept

3.2.1 Contact Benchtop Rig Design

The benchtop rig was designed to meet the conditions noted in Section 3.1 and the planned test architecture highlighted in Figure 3-1 is detailed herein. The design of the system incorporated three primary steps, 1) the design of the impactor, 2) the design of the artifact, and 3) the sensor and data collection protocol to capture the results.

The design of the impactor needed to ensure any motion was solely along the intended axis of travel, while minimizing the impact of frictional losses where possible. Additionally, the impactor construction needed to allow for variable K_I to be implemented. The impactor also needed to have a non-intrusive means of applying a constant force for driving the test. Considering the variable K_I and non-intrusive force input, an impactor assembly was constructed as shown in Figure 3-2.

The impactor assembly consists of two wooden blocks at its base, with an aluminum bracket at its front from which an external load can be applied. Above the base, four wooden blocks contain the primary impactor components. At the front is a cylindrical impactor, which extends through a hole at the front of the impactor to

Figure 3-2: Impactor Design for Experimental Benchtop Rig

restrain its motion to the trajectory of motion. Behind the impactor cylinder is where either a compression spring or other material with a known effective spring constant is placed.

To ensure that all motion of the impactor is along a single axis, the impactor assembly is affixed to a track as shown in Figure 3-3.

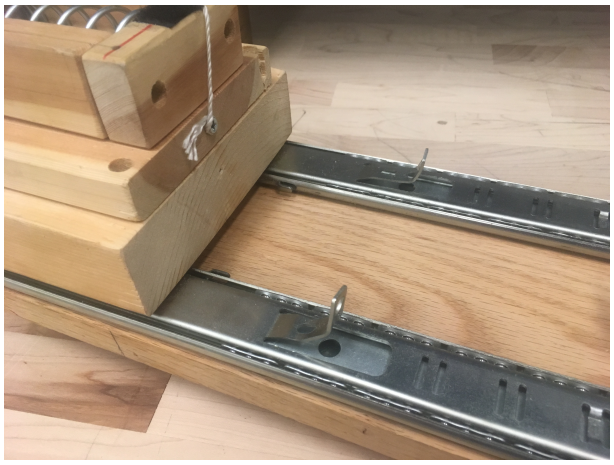


Figure 3-3: Track for Single Axis Motion of Impactor

The track consists of an inner and outer race, with roller bearings between the races to minimize frictional losses during testing.

The design of the artifact, much like the impactor, needs to restrict motion to a single rotational degree of freedom. Additionally, it needs to be able to restrict motion entirely to simulate a fixed artifact condition, and also enable motion with minimal frictional losses for the free artifact condition. With these design goals, an artifact assembly was constructed as shown in Figure 3-4.

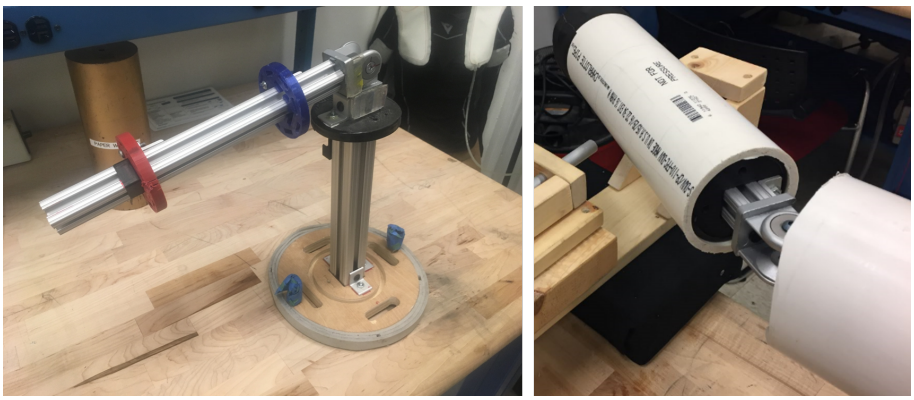


Figure 3-4: Artifact Design for Experimental Benchtop Rig, with internal "skeleton" aluminum joint with stabilizers [left], and with external PVC exterior for contact testing [right]

The artifact consists of two aluminum beams, connected at a pin joint. The beams are enclosed by PVC piping to form the envelope of the artifact that will be impacted. Stabilizer mounts, shown in the left image of Figure 3-4, are used to prevent significant internal bending of the beams when the PVC pipe exterior is impacted. The pin joint used to connect the upper and lower segments of the impactor is responsible for providing the capability to restrict the motion of the leg, or allow free motion with minimal frictional losses. Figure 3-5 highlights this pin joint.

The rotational degree of freedom of the artifact can be tuned by tightening or loosening the screw at the pin joint shown in Figure 3-5. When the screw is fully tightened, the rotational joint is effectively fixed, forming a cantilevered beam configuration. When the screw is loosened, it becomes a pin joint boundary condition, with reduced frictional losses as the screw is loosened.

Figure 3-5: Artifact Pin Joint for Enabling Restricted or Free Rotational Motion

With the impactor and artifact designed to enable the configuration highlighted in Figure 3-1, it remained to identify a sensor set capable of tracking V_I, P, and θ_A during a contact event. First, to track the position, and therefore V_I, of the impactor during a test, the Vicon Motion Capture Nexus System [82] was used. Two Vicon pearls were positioned on the impactor as shown in Figure 3-6.

Figure 3-6: Vicon Pearl Placement for V_I Measurement on Impactor

Next, to sense the pressure associated with a contact event between the impactor and artifact, a Novel Pliance pressure sensor [83] was placed at the point of contact on the artifact, as pictured in Figure 3-7.

Figure 3-7: Novel Pliance Sensor Placement for P on Artifact

Finally, to track the angle of the artifact assembly, an Inertial Measurement Unit (IMU), specifically the APDM Opal [84], was used. The IMU was placed along the internal strut within the artifact assembly to track the relative motion of the artifact. The IMU was supplemented by three Vicon pearls, one at the center of rotation and two along the centerline of the artifact, to verify spatial location, as illustrated in Figure 3-8.

Figure 3-8: Vicon Pearl Placement for θ_A Measurement on Artifact

With this sensor set, the primary degrees of freedom for these experiments could be tracked transiently. To validate the CCM model within the contact regimes of interest, a test plan was developed to capture data germane to the contact regimes of Section 3.1.

3.2.2 Experimental Test Plan and Setup

To capture data representative of those noted in Section 3.1, a series of test points were proposed that provided a discrete grid spanning the bounds of Table 3.1. The full grid of test points are noted in Table 3.2.

Case	**Velocity $[m/s]$**	**Contact Type**	**Impactor Rigidity**
1	0.25	Fixed	Rigid
2	0.50	Fixed	Rigid
3	1.00	Fixed	Rigid
4	0.25	Fixed	Flexible
5	0.50	Fixed	Flexible
6	1.00	Fixed	Flexible
7	0.25	Free	Rigid
8	0.50	Free	Rigid
9	1.00	Free	Rigid
10	0.25	Free	Flexible
11	0.50	Free	Flexible
12	1.00	Free	Flexible

Table 3.2: Experimental Benchtop Test Plan

To accelerate the impactor to the velocities noted in Table 3.2, the impactor assembly was attached to a weight of 0.5 kg that was draped across a rod to allow the weight to pull in the direction of gravity, illustrated in Figure 3-9.

To attain the impact velocities noted in Table 3.2, the impactor distance offset from the artifact was tuned for each target velocity case. To set up the artifact for "Fixed" or "Free" contact types, the frictional screw illustrated in Figure 3-5 was used. To restrict the degrees of freedom of the artifact, the frictional screw was sufficiently tightened until the artifact was unable to appreciably deflect. To set up the artifact for free rotation, the frictional screw was loosened up to the point beyond which the artifact would deflect out-of-plane to minimize frictional losses, while allowing the artifact to deflect.

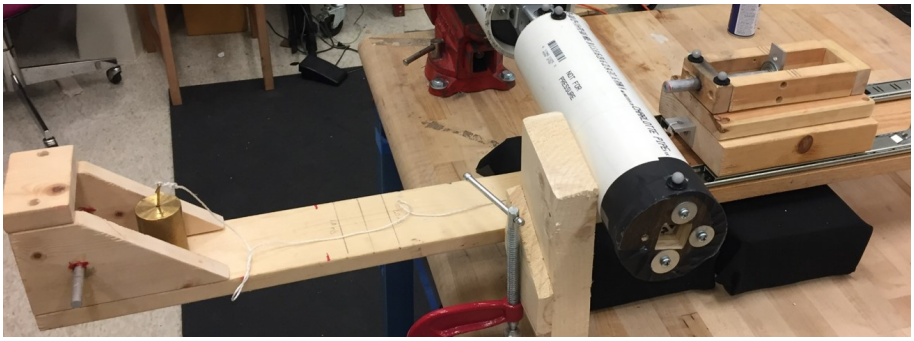


Figure 3-9: Experimental Benchtop Rig with weight attached to Impactor Assembly

Finally, to modify the rigidity of the impactor, the object behind the impactor rod was either constrained by a bracket with a wooden block, or with a compression spring. For reference, the spring constant of the compression spring was characterized and found to be $K_s = 1895\ N/m$.

Using these modifications to the benchtop assembly, the test points of Table 3.2 were executed. Each test case was run three times and the data synchronized between the sensors using a Matlab program interface to the Novel Pliance system, the IMU triggers and the Vicon motion capture system. Finally, the IMU data was corrected for drift during testing by linearly shifting the data to realign with zero at the starting posture before each test case.

The full set of experimental results collected are provided in Appendix C. Using the experimental data, simulations using the CCM model can be run using the boundary conditions from the test procedure. The simulation results for V_i, θ_a, and P obtained from these cases can be compared against the experimental data to identify the capability and accuracy of the CCM model.

3.3 Contact Model Validation Study

To perform back-to-back simulations using the CCM model of the experimental test cases noted in Section 3.2, a computational representation of the benchtop system had to be constructed, relevant boundary conditions and constraints provided, and a consistent post-processing method used to extract simulation data consistent with

that collected in the experimental testing.

3.3.1 Model Representation of Experimental Rig

To simulate the experimental test cases using the CCM model, a CAD representation of the benchtop system was constructed using SolidWorks. First, the full artifact assembly was constructed, as illustrated in Figure 3-10.

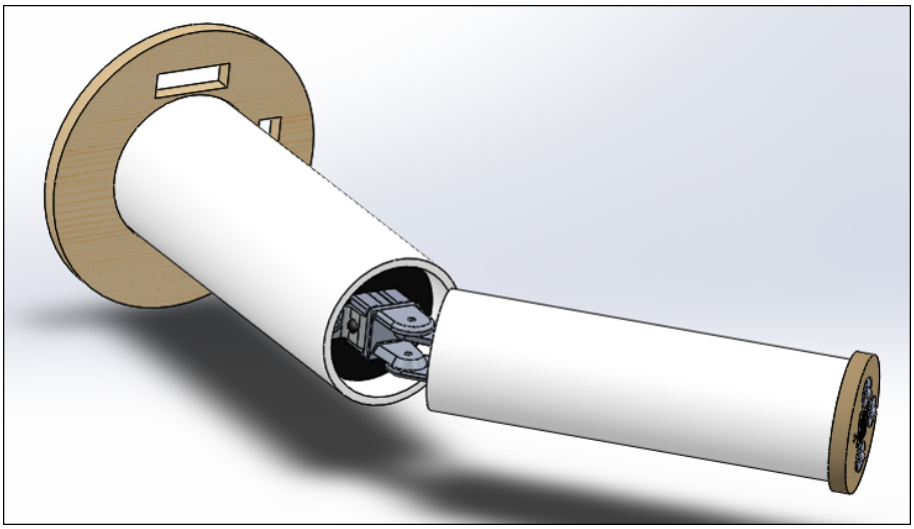


Figure 3-10: Full Artifact Assembly CAD Model

However, as the nature of the experimental testing was to capture only motion relevant to rotational component of the artifact, the full CAD was not used in the simulations. Instead, the CAD was reduced to only include the rotational component. Furthermore, as the internal construction within the artifact contained a collection of brackets, bolts, and beams with small radii fillets which are not germane to the contact events occuring on the outer shell of the artifact, a dynamically similar, simplified CAD was developed. The external envelope size of the artifact was preserved, but the internal beams and stabilizers simplified to cylinders and rectangular beams to eliminate unnecessary complexity. To ensure that the dynamic response of the artifact would match that of the full, high fidelity model, the material properties of the internal

components were tuned until the center of mass and all moments of inertia about the center of rotation matched the full CAD model. This simplified but dynamically similar CAD representation of the artifact is illustrated on the right in Figure 3-11, while the full CAD is shown on the left for reference.

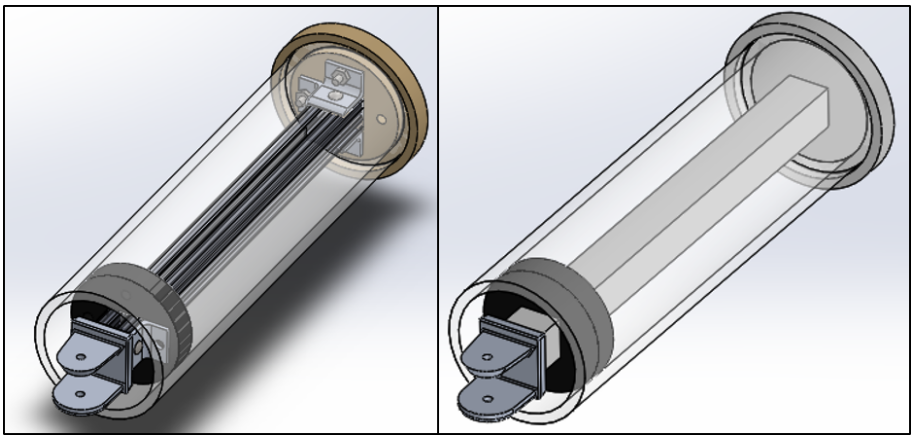


Figure 3-11: Rotational-Only Artifact CAD - High Fidelity (left), Simplified (right)

The impactor assembly required two configurations to be represented in the CAD model; one for the "Rigid" impactor and another for the "Flexible" impactor. The Rigid impactor CAD is illustrated in Figure 3-12.

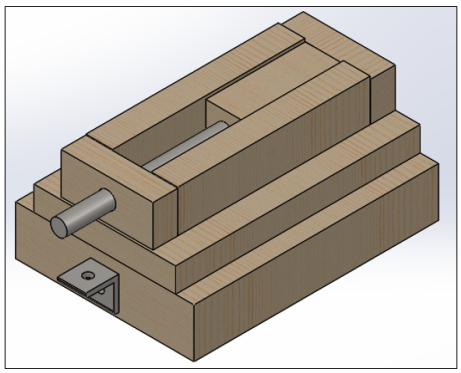


Figure 3-12: Impactor CAD Model - Rigid Configuration

As with the artifact simplification, for the Flexible configuration of the impactor the compression spring was represented via a simple cylinder to reduce the complexity of downstream meshing. To ensure the cylindrical representation of the spring would capture both the spring constant and mass of the full fidelity spring, the Young's Modulus of the spring cylinder was set by

$$E_s = \frac{K_s L}{A} \tag{3.1}$$

Furthermore, to ensure that the cylindrical representation captured the spring constant $K_s = 1895\ N/m$, a static uniaxial compression test was run using SolidWorks Simulation with a 1.0 N compressive force, and the deflection of the cylinder verified to meet this requirement. The simplified Flexible Impactor CAD and the uniaxial compression test results are illustrated in Figure 3-13.

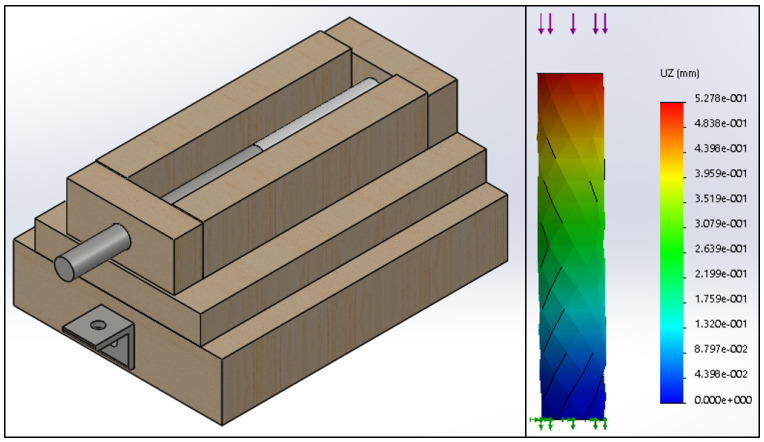


Figure 3-13: Impactor CAD Model - Flexible Configuration (left), K_s check (right)

The artifact and impactor CAD models were arranged in space to ensure the starting posture of each test case could be set in a consistent manner. The primary inputs to the combined CAD model representation of the test cases is illustrated in Figure 3-14.

Initial geometric configurations for each test case were generated in SolidWorks,

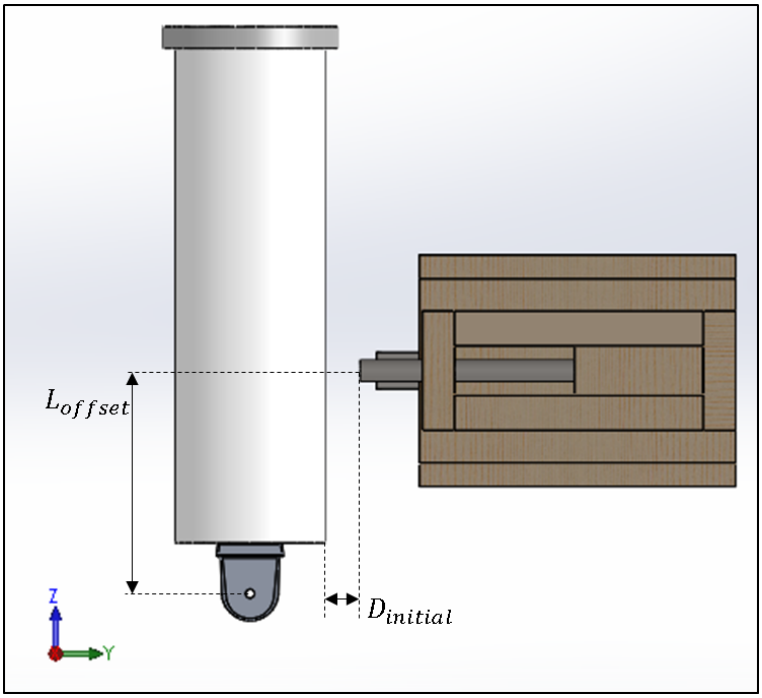


Figure 3-14: Combined CAD Model with Initial Geometry

and provided as the input to the SolidWorks Simulation environment for meshing and identification of boundary conditions for interaction with Summit.

3.3.2 Model Meshing and Boundary Condition Setup

The SolidWorks Simulation environment was selected for meshing the combined CAD models for each test case configuration. While any meshing environment is viable for meshing the domain, a special conversion toolset was developed over the course of this work that enabled ease of porting information between SolidWorks and SUMMIT through the Universal I-DEAS file format. Further information on this conversion toolset can be found in Appendix D. The combined CAD models were meshed using the SolidWorks standard mesh format, and with the restriction to only generate linear elements to enable interaction with the SUMMIT contact methods. The resultant meshes have element counts on the order of 5×10^3, with an example of the Rigid, Fixed configuration illustrated in Figure 3-15.

To drive the simulation, two primary boundary conditions were provided to SUM-

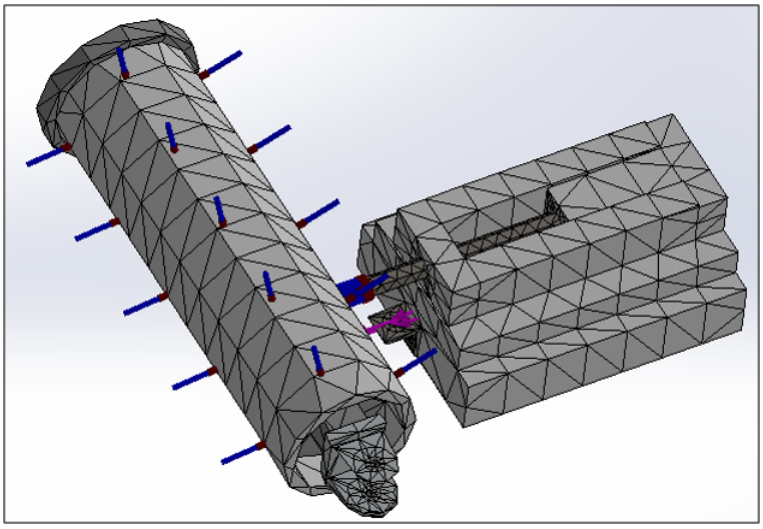


Figure 3-15: Rigid Impactor, Fixed Artifact Simplicial Mesh

MIT from SolidWorks; the input force used to drive the impactor and the selection of surfaces on which to enforce contact. The input force was selected to act along the interior of the bracket on the front of the impactor, to represent the force of the weight pulling the impactor during the simulation. The surface on which the input force acts is illustrated in Figure 3-16.

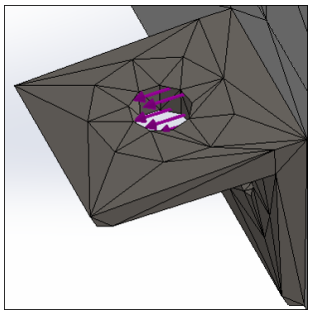


Figure 3-16: Impactor Input Force Boundary Condition Location

While only an optional input, to reduce computational expense, surfaces in the CAD on which to enforce contact in SUMMIT were selected in SolidWorks. The surfaces selected to form SUMMIT Contact Surface objects are highlighted in Figure 3-17.

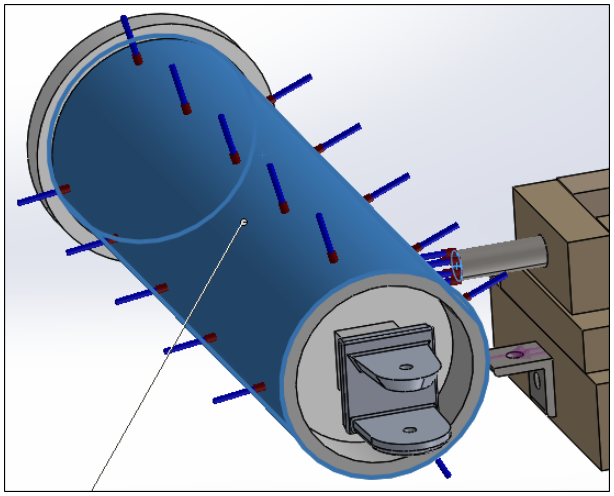


Figure 3-17: Selected Surfaces for Contact Enforcement

Observation of the input force location noted in Figure 3-16 highlights that the force used to pull the impactor is not along the axis of the impactor center of mass. In the experimental setup, the track which the impactor traversed, coupled with gravity, ensured that all motion was along a single axis and the impactor would not rotate. In the CCM model simulation, it is undesirable to model the effects of gravity or the frictional losses of the track due both the added computational expense. Instead, Dirichlet constraints were applied to the top and bottom of the impactor which restrict motion in the out-of-plane direction. Additionally, while contact surfaces could be used to constrain the out-of-plane motion of the impactor rod, resolving those contact interactions is not germane to the simulation. To maintain a single degree-of-freedom motion of the impactor rod, Dirichlet constraints were applied to the impactor rod as well to restrict out-of-plane motion. The constraints acting on the impactor assembly are illustrated in Figure 3-18.

For the Fixed and Free artifact configurations, two different sets of Dirichlet-type constraints were used. For the Fixed configuration, all nodes along the interior surfaces of the artifact center of rotation were constrained in all degrees of freedom, effectively forming a cantilevered beam. For the Free configuration, a rotational boundary condition formulation was developed with a simple Coulomb friction model and applied to the same nodes along the interior surfaces of the artifact center of ro-

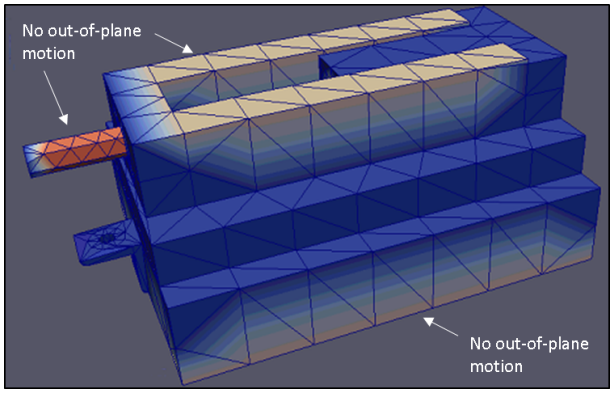


Figure 3-18: Dirichlet Constraints Applied to Impactor

tation. The governing equations of motion for the Free artifact rotational boundary condition are enforced as an additional "Corrector" step along the lines of the correction step noted in Section 2.5 after both structural mechanics and contact mechanics have completed. First, the normal velocity of each node is computed by

$$v_{n_{i+1}} = max(\frac{r_{i+1}^{'} - r_i}{\Delta t}, 0.0) \tag{3.2}$$

where $r_{i+1}^{'}$ is the predicted radius of a given node after structural and contact mechanics have been enforced, and r_i is the radius of that node from the prior time step. Next, the idealized tangential velocity is found by

$$v_{t_{ideal}} = r_i(\frac{\theta_{i+1}^{'} - \theta_i}{\Delta t}) \tag{3.3}$$

where $\theta_{i+1}^{'}$ is the predicted angle of a given node after enforcing structural and contact mechanics, and θ_i is the prior nodal angle. Finally, to enforce a Coulomb friction law on the rotation, the following nonlinear equation is used to compute the corrected tangential velocity

$$v_{t_{i+1}} = \begin{cases} 0.0 & \text{if } |v_{t_{ideal}}| \leq \mu_s |v_{n_{i+1}}| \\ v_{t_{ideal}} - \mu_k(\frac{|v_{n_{i+1}}|}{|v_{t_{ideal}}|} v_{t_{ideal}}) & \text{if } |v_{t_{ideal}}| > \mu_s |v_{n_{i+1}}| \end{cases} \tag{3.4}$$

where μ_k is the specified kinetic friction coefficient and μ_s is the specified static friction coefficient. Finally, to translate the corrected tangential velocity and enforce

the rotational boundary condition, the following constraints are applied:

$$r_{i+1} = r_i \tag{3.5}$$

which ensures no radial displacement of the nodes and

$$\theta_{i+1} = \frac{v_{t_{i+1}}}{r_{i+1}} \tag{3.6}$$

which drives the rotation of the nodes about the center of rotation based on the updated tangential velocity $v_{t_{i+1}}$. While such a friction model is relatively simple, it provides a means by which the rotational motion of the Free artifact can be tuned to account for frictional losses not germane to the CCM model. Additionally, it should be stressed that while this friction model is intended to replicate the effect of a Coulomb model with μ_s and μ_k, the physical relevance of μ_s without applying a constant normal force to the nodes is weakened, and acts as more of a tuning parameter for capturing static frictional effects. For both the Fixed and Free artifact configurations, the nodes within the artifact mesh that were selected to enforce these laws and exact constraints are illustrated in Figure 3-19.

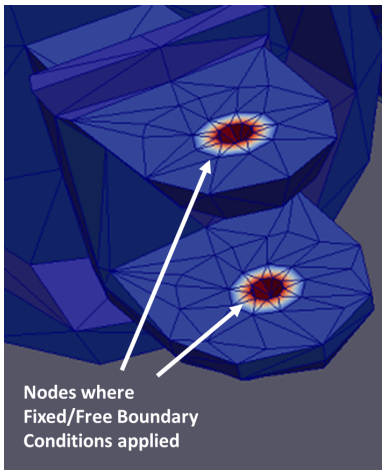


Figure 3-19: Nodal locations for Dirichlet constraints applied to artifact to enforce Fixed/Free configurations

Finally, the manner in which the input force was driven into the system should be discussed. While frictional losses cannot be discounted, the transient response to the system up until the contact event is of less significance, so long that the kinetic energy and posture of the system aligns well with the experimental data. To avoid the added complexity of a frictional model, the input force provided to the impactor in the simulations, up until the contact event, was calculated through differentiating the tracked impactor velocity, V_i, over time and computing the input force through the known mass of the impactor. In this fashion, for all times leading up to the contact event, the input force is effectively an open-loop driver used to ensure the impactor tracks the motion prescribed by the simulation. However, this methodology cannot apply once the contact event occurs. As a result, whenever the experimental pressure, P, was non-zero and the differentiated input force was negative, the input force was instead forced to zero. During these events, if the differentiated input force was positive, this value was provided to the simulation. This approach was adopted to match forces feasibly introduced by the weight, while allowing the CCM model response to drive any momentum changes in the negative direction due to the contact event.

While this open-loop methodology simplified the boundary conditions applied to the impactor at all times leading up to the contact event, it comes at some expense. Namely that the boundary conditions applied to the impactor during and after the contact event are unknown following the contact event, as attempting to match the open-loop velocity profile would yield a trivial result; the model would replicate the contact event exactly because it was forced to do so, rather than using the CCM model to enforce contact. Additionally, as the open-loop method offers the benefit of not requiring tuning of frictional coefficients, it also provides the detriment of not knowing the frictional coefficients that act as hysteresis and damping on the impactor. Finally, during the contact event the weight used to drive the impactor forward, illustrated previously in Figure 3-9, would rebound upwards due to the contact impulse. This rebound would result in a short time period in which the force of the weight was not applied to the impactor, an effect not explicitly captured by this open-loop method.

In future studies, it may be worthwhile to apply a load cell to the input driver of the impactor, to allow the transient force applied to be tracked and applied directly to the simulation. This would require up-front tuning of a friction model for the impactor, but once matched coefficients are obtained, the uncertainty for post-impact events would be reduced. However, for this study, the open-loop method described above was implemented for speed and the specific focus on resolving the contact event.

3.3.3 Model Post-Processing Procedure

Before reviewing the results of the CCM model simulations of the experimental data, the means by which the simulation data was post-processed should be discussed. To reiterate, the three macroscopic metrics that are to be tracked on a back-to-back basis are the impactor velocity, V_i, the impact pressure, P, and the artifact angle, θ_A. To post-process the CCM model results from SUMMIT, the data analysis and visualization software Paraview [85] was used. This section details how each of these metrics were extracted and post-processed from the CCM model simulations.

The impactor velocity, V_i, was relatively straightforward to obtain from the model. As Figure 3-6 highlighted, two Vicon pearls were placed on the top of the impactor to track its velocity over time. In a similar fashion, two nodal points on the impactor were selected and tracked over time. The selected nodes are illustrated in Figure 3-20.

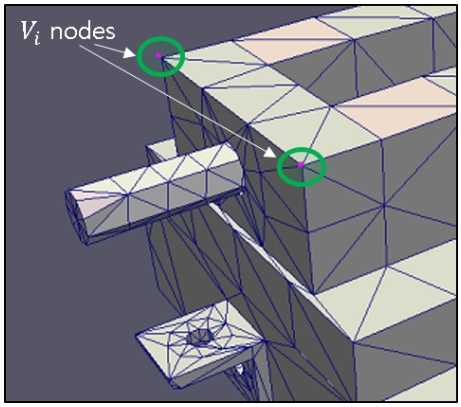


Figure 3-20: Nodes used for V_i tracking in CCM Model Simulation Results

The artifact angle, θ_a was similarly straightforward to track in the simulation. As the center of rotation of the artifact was either fully constrained in the Fixed configuration, or permitted rotational-only motion in the Free configuration, the angle relative to the center of rotation could be tracked through the use of one node in a similar fashion to the Vicon pearls shown in Figure 3-8. The node from the CCM model simulation used for calculating θ_a is illustrated in Figure 3-21.

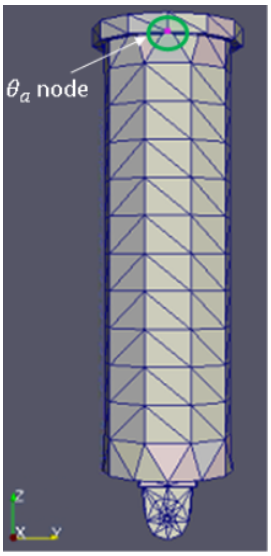


Figure 3-21: Node used for θ_a tracking in CCM Model Simulation Results

The artifact angle, θ_a was then calculated at each time step by

$$\theta_a(t) = \frac{180^\circ}{\pi} \times \tan^{-1}(\frac{u_y(t)}{x_z(t)}) \tag{3.7}$$

where $u_y(t)$ is the displacement of the node in the direction of the impactor motion, and $x_z(t)$ is the distance of the node from the artifact center of rotation.

Finally, the pressure, P, of the contact event was tracked through a SUMMIT Contact Surface internal method during runtime, rather than in Paraview. As the forces introduced for enforcing the contact constraint via the CCM model are not true Neumann type conditions, but instead are a nodal displacement and velocity update,

the effective contact forces acting on the artifact had to be calculated by tracking the nodes within the contact surface and continuously checking for updated velocity components. Whenever the DCR method updated the nodal velocities, reflecting a node had been pushed back to enforce contact, this velocity was differentiated across the model time step and multiplied by that node's lumped nodal mass. This total contact force vector was then projected along the axis of motion of the impactor. These forces were then collected and reported at a given time step as the total set of contact forces acting at that time, noted as $F_c(t)$ over the surface of the sensor, Γ_s. To compute the pressure acting on the artifact in a manner consistent with that of the Novel Pliance sensor illustrated in Figure 3-7, the total surface area of the Novel Pliance sensor was used as the area term. Any contact force acting within the bounds of this surface area was then divided by the area of the sensor, A_p, by

$$P_{raw}(t) = \frac{|F_c(t)|_{\Gamma_s}}{A_p} \tag{3.8}$$

While this provides the raw pressure acting on the artifact during the simulation, the Novel Pliance sensor has certain characteristics which require further processing of the simulation data to properly align with the experimental data. The Novel Pliance sensor was observed to have a deadband at low pressure; this is to say that the pressure sensor would report a value of zero for pressure below a specific threshold. In addition to this deadband, the Novel Pliance sensor was also sampled at a frequency of 128 Hz, and as a result, effectively aliases the sensed pressure beyond half of this frequency, or the Nyquist frequency. These digital effects are significant for aligning pressure data, as a large pressure spike over a short time scale may not be realized by the pressure sensor, while the explicit CCM model resolves every high frequency pressure oscillation within its significantly smaller time step size. To allow the CCM results to align with the sampled pressure data, a 1st-order butterworth lowpass filter was applied to the raw simulation pressure data, P_{raw}, with a cutoff frequency set to 64 Hz, or the Nyquist of the Novel Pliance sensor based on its sampling rate. A normalized Bode diagram of the 1st-order lowpass filter applied to the simulation

pressure data is illustrated in Figure 3-22.

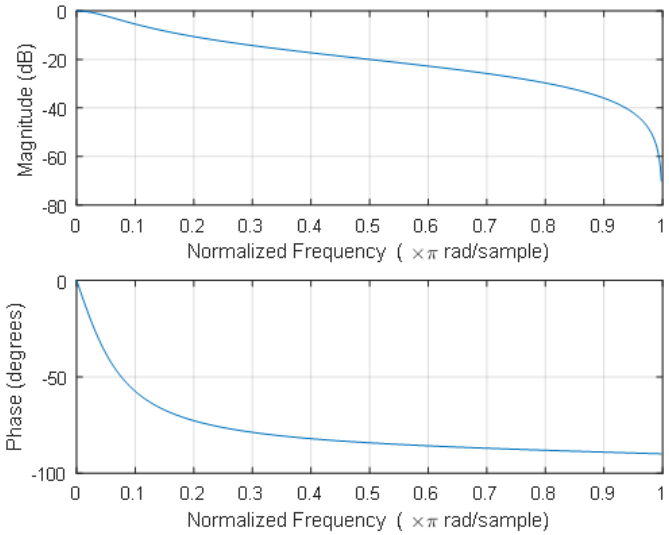


Figure 3-22: Normalized Bode Diagram of Digital 1st-Order Lowpass Filter applied to CCM model simulation pressure results for alignment with experimental pressure data sampling rate

This filtered simulation data was then used to align with the experimental data presented in Section 3.4.

3.4 Comparison of Model to Experimental Results

The CCM model was used to simulate the test cases noted in Table 3.2, using the methods discussed in Sections 3.3.1, 3.3.2, and 3.3.3. Input boundary and initial conditions were derived from the experimental data. This section details the back-to-back results of the experimental data to the CCM model results for all four configurations:

1) Rigid Impactor, Fixed Artifact (RX)

2) Rigid Impactor, Free Artifact (RF)

3) Flexible Impactor, Fixed Artifact (FX)

4) Flexible Impactor, Free Artifact (FF)

It should be noted that only results for test cases involving an impactor velocity of 0.25 m/s are presented here. This reduced selection is because the interaction of the minimum stable time step between the structural mechanics domain was unable to resolve the higher kinetic energy cases without heuristically slowing the simulation time scale. As a result of the stable time step interaction issue, higher velocity cases became unstable at the time of the contact event. This stability issue and potential mitigation strategies are discussed in greater detail in Section 4.2. However, the results for the 0.25 m/s impactor velocity still provide a means of comparing the experimental data against the CCM model. The capability of the model to match experimental results in V_i, P, and θ_a for each 0.25 m/s configuration are detailed herein.

Additionally, a set of terminology will be used throughout the discussion of these results, namely "effective rigidity." This term is a means of referring to the relative rigidity of one object to another, such as that given by the Young's Modulus ratio ER considered in Section 2.8. This definition is pertinent as the relative rigidity of two objects involved in a contact event dictates the resultant response of the two objects. If the artifact has higher effective rigidity in simulation than in experimental data, the artifact will deflect in a shorter time scale, with a greater sensed pressure over a shorter period of time, than if it had less effective rigidity. Additionally, for the Fixed configuration (which acts like a cantilevered beam), the oscillatory frequency of the artifact would be higher with higher effective rigidities, as the natural frequency would scale with its springlike qualities by $\omega_n = \sqrt{\frac{k}{J}}$ where k is the effective rigidity expressed as a spring constant, and J is the moment of inertia of the artifact about its center of rotation.

3.4.1 Rigid Impactor, Fixed Artifact Configuration Results

The second impact test from the experimental data for the Rigid Impactor, Fixed Artifact, 0.25 m/s target velocity case (RX025) was selected as the trial to simulate

with the CCM model. The CCM simulation was run in parallel using 16 processors, with dynamic RCB partitioning for the contact domain. The experimental and CCM model results for the impactor velocity, V_i, are illustrated in Figure 3-23.

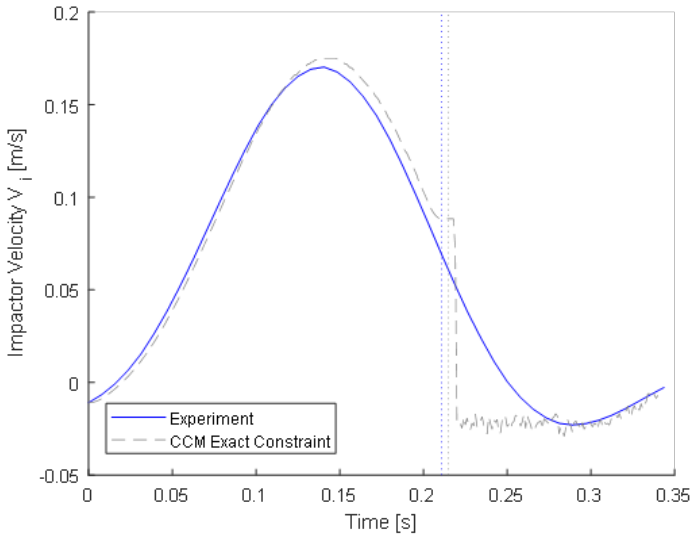


Figure 3-23: RX025 V_i Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dotted), CCM Contact Time (grey dotted)

Observation of the back-to-back comparison of V_i between the experimental data and the CCM model results highlights that the CCM model predicted a contact event with significantly higher rigidity than the experimental data reported. Furthermore, the CCM model also predicts the contact event occurred approximately 4 ms later than in the experimental data. However, it should be noted that the sampling rate of the experimental data was at 7.8125 ms, thus the accuracy of the CCM model impact event prediction is within one experimental frame of accuracy. The primary differentiation between the CCM model and the experimental data is the significantly shorter time scale associated with the contact event in the CCM model results, relative to the experimental data. This sharp acceleration can be attributed to two possible causes. First, is due to the nature by which the input force is provided in

the simulation. As noted at the end of Section 3.3.2, the input force provided to the CCM model simulation was based on the differentiation of the velocity data, and known mass of the impactor to define an input force open-loop that would account for frictional losses and other un-modeled dynamics by virtue of tracking being derived from the trajectory of the impactor. However, once the contact event was detected (noted as any time in which the sensed pressure was non-zero), the input force was modified to ensure only positive forces could be attributed to the system to allow the CCM model to resolve the change in momentum of the system, while only allowing forces driven by the weight to modify the input force. However, this formulation does not account for static or kinetic frictional losses associated with the impactor track when the impactor is being pushed back by the fixed artifact. Without this damping force, the response of the rigid impactor to the fixed artifact will appear purely as a function of the rigidity of the system and experience no frictional damping that would be present in the physical system. Additionally, the frictional screw used to constrain the artifact into the Fixed configuration cannot perfectly restrict all displacement, while the CCM model simulation can enforce an exact restriction on all degrees of freedom of the artifact center of rotation. The exact constraints of the simulation would therefore cause the effective rigidity of the artifact to be greater than in the physical system. If the simulation artifact does have greater effective rigidity, the artifact will experience less overall deflection than the physical system, and for a shorter overall length of time. Figure 3-24 illustrates the transient θ_a results from the experimental data and the CCM model.

As suggested in the V_i results, the effective rigidity of the artifact is higher for the CCM model simulation than in the physical system. This increased effective rigidity is apparent in that the maximum deflection of the experimental system peak at approximately 0.14 degrees, while the CCM model, with a perfectly constrained center of rotation, only deflected approximately 0.03 degrees, and proceeded to oscillate at a higher frequency as well. This would suggest that the primary driver of the difference in the response of the simulation versus the experimental data is due to the model constraints more effectively preventing deflection than the physical system. To

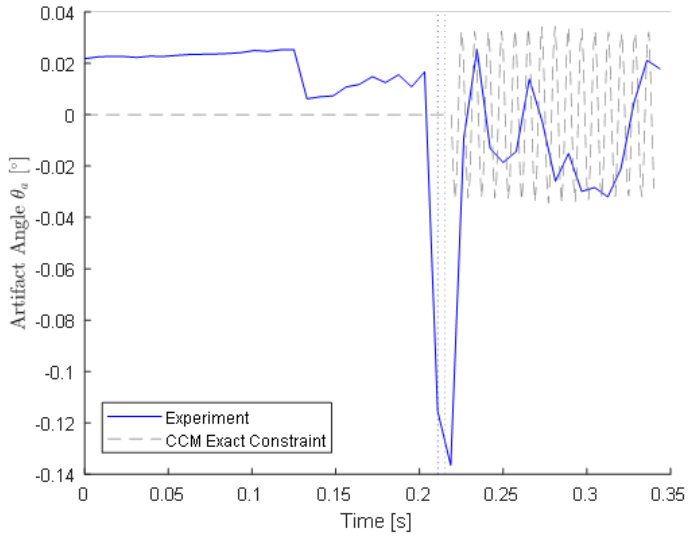


Figure 3-24: RX025 θ_a Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time (grey dotted)

corroborate that the constraints have restricted deflection, the transient pressure, P, results are shown in Figure 3-25.

The transient pressure responses shown in Figure 3-25 corroborate the higher effective rigidity of the artifact in the CCM model simulation than in the physical system. The greater pressure magnitude over a shorter time scale highlights that the contact event transferred more kinetic energy in a shorter span of time than the physical system, whereas in the physical system, some of that energy was used in the greater displacement of the artifact, resulting in a lower sensed pressure. Additionally, as the artifact displaced for a greater length of time in the physical system, the impactor remained in contact with the artifact for a longer duration, spreading out the period of sensed pressure. The CCM model simulation, by comparison with the V_i results in Figure 3-23, indicate that the artifact pushed the impactor back in a shorter period of time, yielding a span of time in which the impactor was not in contact with the artifact, prior to the impactor rolling forward and re-impacting the artifact.

To verify that the exact constraint is the root cause for the higher effective rigidity of the CCM model result versus the experimental data, the friction model described in Equations 3.2 through 3.6 was employed to constrain the artifact, and prescribed

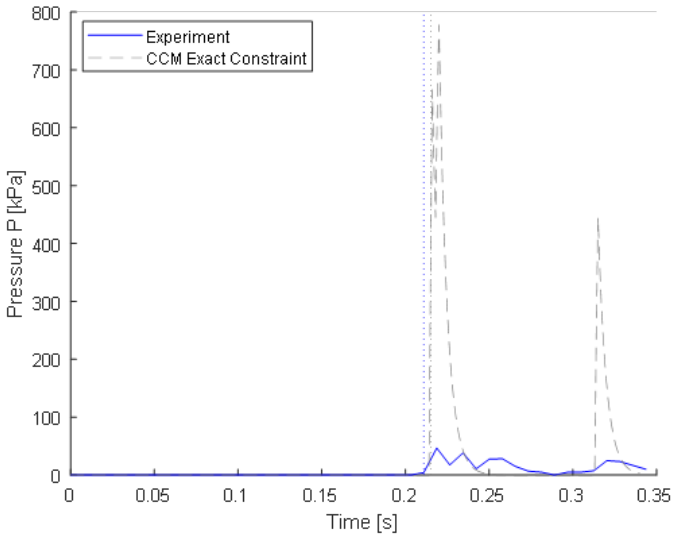


Figure 3-25: RX025 P Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time (grey dotted)

an artificially high static coefficient of friction, μ_s, of 50.0 and kinetic coefficient of friction, μ_k, of 0.99. This methodology, while expected to approach the exactly fixed constraint, permits some local nodal displacement if the tangential momentum of a given node is sufficiently high and as such, expected to capture the more compliant constraint of the experimental setup. Figures 3-26, 3-27, and 3-25 represent the simulation results using this frictional constraint approach against the experimental data and the exact constraint simulation results for V_i, θ_a, and P, respectively.

By representing the Fixed configuration with the frictional model, the overall trend in the impactor velocity is not significantly modified. The end velocity of the impactor is slightly higher than that represented by the exact constraint; given that the exact constraint permits no deflection at the center of rotation, this corroborates that the exact constraint method results in an artifact with higher effective rigidity than that of the frictional model.

The angle of the artifact, θ_a, captures this reduction in kinetic imparted back into the impactor, as the overall deflection of the artifact is greater with the friction model than with the exact constraint. As some localized deformations are permitted, albeit heavily penalized by the large μ_k and μ_s, the friction model causes the artifact

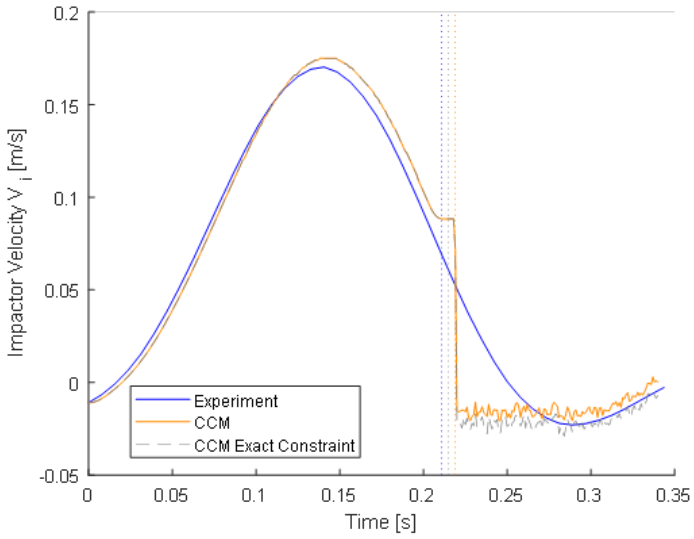


Figure 3-26: RX025 V_i Transient Results, Experimental (blue), CCM Model with Frictional Constraint (orange), CCM Model with Exact Constraint (grey dashed), Experimental Contact Time (blue dotted), CCM with Frictional Constraint Contact Time (orange dotted), CCM with Exact Constraint Contact Time (grey dotted)

to absorb more kinetic energy from the impactor, causing it to deflect more as a result of the contact event. Furthermore, the reduction in the effective rigidity of the artifact is more apparent when comparing between the CCM model results using the friction model and the exact constraint. The exact constraint oscillates with a higher frequency than that of the friction model, indicating that as the moment of inertia, J, of the artifact is constant, the effective rigidity of the artifact is lower for the friction model approach.

Finally, the back-to-back results of the sensed pressure, P, aligns with the trends of reduced effective rigidity of the artifact when using the friction model. As the impactor deflects further than with the exact constraint, the pressure of the impact is delayed until the frictional forces hold the artifact in place. At this point, some kinetic energy from the impactor has already been imparted to deflecting the artifact, and the resultant pressure is less than in the case of the exact constraint, which has higher effective rigidity.

Attention should be called to the velocity of the impactor post-impact for both the experimental data and the CCM model simulation in Figure 3-26. While the

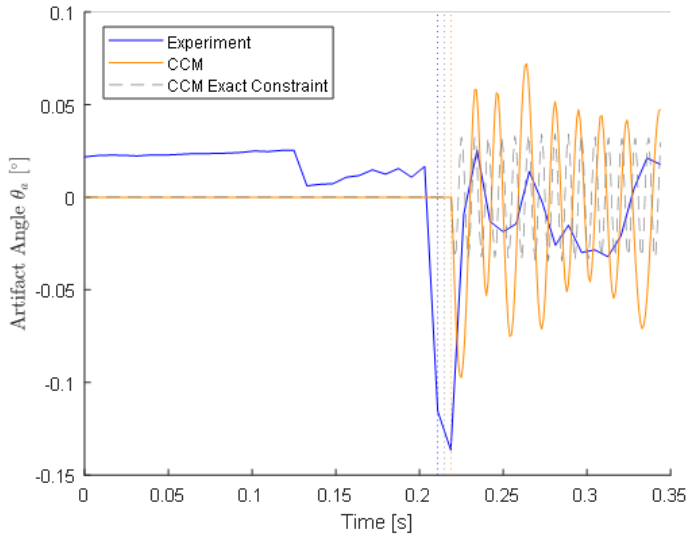


Figure 3-27: RX025 θ_a Transient Results, Experimental (blue), CCM Model with Frictional Constraint (orange), CCM Model with Exact Constraint (grey dashed), Experimental Contact Time (blue dotted), CCM with Frictional Constraint Contact Time (orange dotted), CCM with Exact Constraint Contact Time (grey dotted)

CCM model results reached the post-impact velocity of approximately -0.02 m/s more quickly, both the CCM model simulation and experimental data reached equivalent velocities post-impact. This alignment indicates the overall change in momentum of the impactor predicted by the CCM model matches that of the physical simulation. As a result, the CCM model is effective at capturing the overall kinetic energy changes associated with the physical system for the Rigid Impactor, Fixed Artifact test case. While the updated results incorporating the friction model highlight some improvements in matching aspects of the data, one primary boundary condition that is not captured in the CCM simulation results are the frictional losses associated with the impactor. The effects of friction leading up to the impact event were captured through driving the simulation in an open-loop fashion, but after the contact event, this method was not reasonable to apply. As a result, the impactor velocity during and post-impact is now a function of the un-modeled frictional losses. If a friction model were applied to the impactor, it is expected that the pressure time scale would increase, as the artifact must overcome friction to push the impactor back. This highlights the need for careful tuning of friction models and boundary conditions

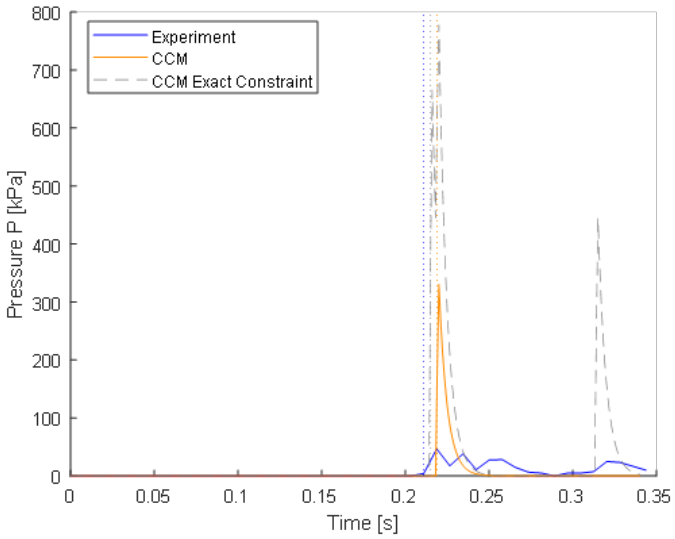


Figure 3-28: RX025 P Transient Results, Experimental (blue), CCM Model with Frictional Constraint (orange), CCM Model with Exact Constraint (grey dashed), Experimental Contact Time (blue dotted), CCM with Frictional Constraint Contact Time (orange dotted), CCM with Exact Constraint Contact Time (grey dotted)

when attempting to data match the CCM model, but the conservation of kinetic energy highlighted in Figure 3-26 reinforces that, for the contact event itself, the CCM model is capable of capturing experimental contact events for a Rigid Impactor, Fixed Artifact configuration.

3.4.2 Rigid Impactor, Free Artifact Configuration Results

The second impact test from the experimental data for the Rigid Impactor, Free Artifact, 0.25 m/s target velocity case (RF025) was selected as the trial to simulate with the CCM model. The CCM simulation was run in parallel with 16 processors, using dynamic RCB partitioning. The impactor velocity, V_i, back-to-back results of the experimental data versus the CCM model, with and without friction, are shown in Figure 3-29. Note that the coefficients of friction selected for the CCM model results with artifact friction included are $\mu_s = 0.2$ and $\mu_k = 0.1$.

In both cases for the CCM model (with and without friction), the impactor is pushed back significantly further than in the experimental data. This increased change in velocity would suggest that, regardless of if the artifact can rotate with

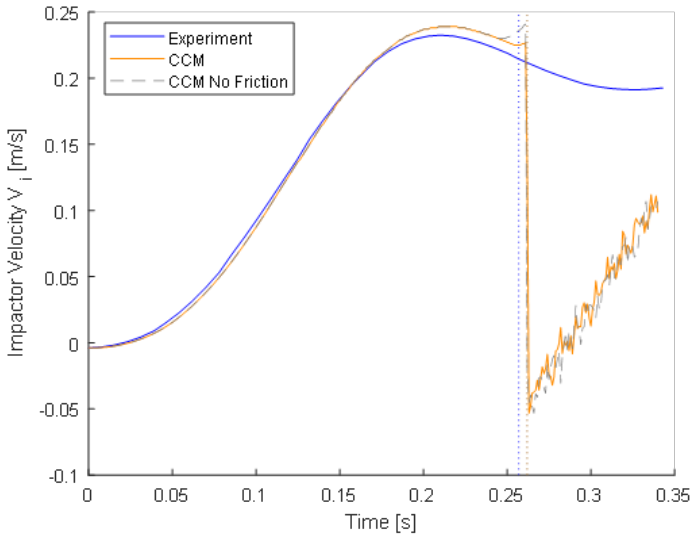


Figure 3-29: RF025 V_i Transient Results, Experimental (blue), CCM Model with friction (orange), CCM Model with no friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time with no artifact friction (grey dotted)

or without friction, the change in momentum applied to the impactor is significantly higher than the experimental results. As noted in the results for the Rigid Impactor, Fixed Artifact configuration, the impactor is driven with an open-loop force input to match experimental data leading up to the contact event; during and after the event, a constant input force of the applied weight is provided, but no frictional losses are modeled. It is expected that this lack of frictional losses results in the impactor being pushed back too easily, as in the physical system, the friction of the impactor race would resist and damp the change in momentum of the impactor. This distinction is of greater importance for configurations where the artifact is Free rather than Fixed, as the effective rigidity of the artifact for the Fixed configuration is essentially constant; for the Free condition, the initial deformation of the material may appear rigid, but the allowance for rotation makes the artifact highly compliant. So for the Fixed artifact configuration, the higher effective rigidity of the artifact results in an equivalent change in momentum between the CCM model and the experimental data, but for the Free artifact configuration, the impactor still experiences the short time scale of the contact event due to the rigid artifact material and deflects backwards in

a similar fashion to the Rigid Impactor, Fixed Artifact case. Despite the simulation not capturing the frictional effects acting on the impactor post-impact, it is expected that the artifact angle should match relatively well with the experimental data, as the imparted kinetic energy should be conserved based on the Rigid Impactor, Fixed Artifact results. Figure 3-30 illustrates the θ_a transient response of the CCM model, with and without friction, on a back-to-back basis with the experimental data.

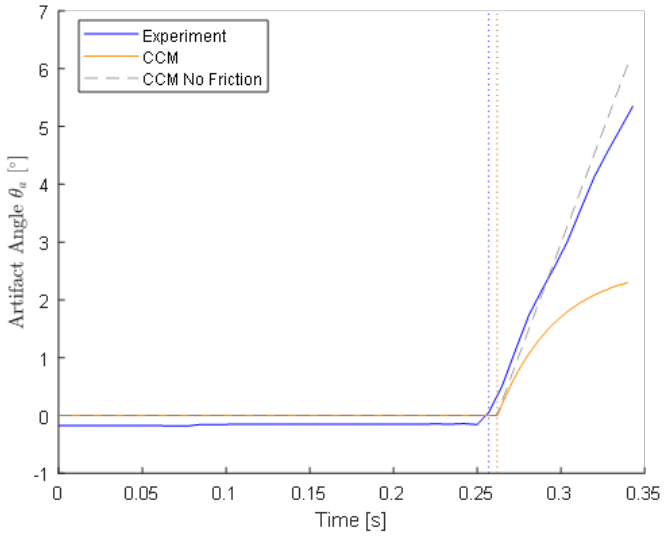


Figure 3-30: RF025 θ_a Transient Results, Experimental (blue), CCM Model with artifact friction (orange), CCM Model with no artifact friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time with no artifact friction (grey dotted)

The results highlighted in Figure 3-30 suggest that the frictional losses at the artifact center of rotation are closer to negligible than for the results with $\mu_k = 0.2$ and $\mu_s = 0.1$. However, the CCM results with no frictional losses due continue with no change in angular velocity while the experimental data does slightly decay. As the experimental results are bounded by the friction and frictionless CCM results, this indicates that the simulation frictional coefficients could be tuned to match experimental data. Additionally, as the CCM model results without friction align closely

with the experimental results up until 0.3 seconds, this highlights that the kinetic energy transfer to the artifact agrees well between the CCM model and the physical system. Given the results for V_i and θ_a, this indicates that the CCM model is correctly capturing the contact event, but due to the lack of friction model acting on the impactor, cannot correctly resolve the post-impact response of the impactor. Additionally, given that the impactor velocity, V_i, was changed significantly further and with greater magnitude than the experimental data, it is expected that the sensed pressure, P, of the contact event will experience a similar higher magnitude, shorter time scale response. Figure 3-31 illustrates the back-to-back sensed pressure results between the CCM model and experimental data.

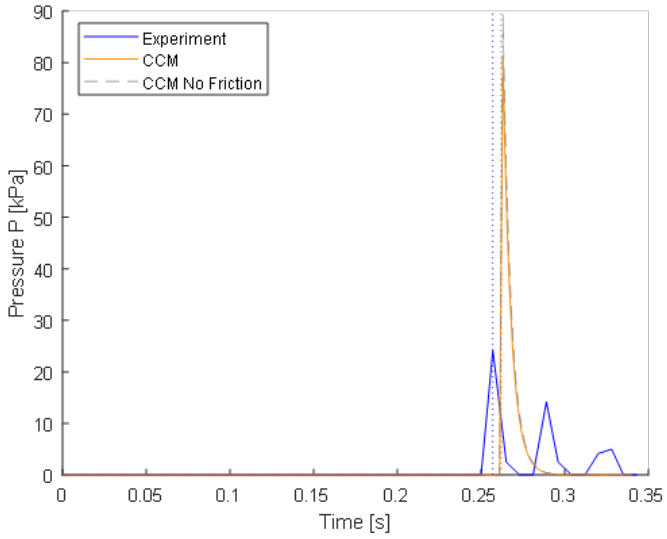


Figure 3-31: RF025 P Transient Results, Experimental (blue), CCM Model with artifact friction (orange), CCM Model without artifact friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time without artifact friction (grey dotted)

As expected, in both instances the CCM model predicts a notably higher pressure magnitude over a shorter time scale than the experimental data, with or without artifact friction. This result is expected as the impactor velocity, V_i, is drastically

reduced relative to the experimental data as a result of the impact. This shorter time scale contact event in the CCM results yields results in an equivalent artifact deflection as seen in Figure 3-30, but without the hysteresis or damping provided by friction acting on the impactor, the impactor experiences a greater change in momentum than the physical system. If the impactor were pushed back at a rate mimicking that of the experimental data, it would be expected to yield smaller, repeated contact events as seen in the experimental results as it continues to impact the artifact due to its continued forward momentum. These results stress the need for a proper friction model to be applied to the impactor to guide the post-impact response of the impactor, but the θ_a response highlights that the kinetic energy imparted to the artifact matches well with the experimental data. This suggests that, similar to the Rigid Impactor, Fixed Artifact case, the CCM model is correctly resolving the contact event itself and that the post-impact differences between the simulation and experiment are a function of the boundary conditions (or lack thereof) applied to the impactor and artifact.

3.4.3 Flexible Impactor, Fixed Artifact Configuration Results

The second impact test from the experimental data for the Flexible Impactor, Fixed Artifact, 0.25 m/s target velocity case (FX025) was selected as the trial to simulate with the CCM model. The CCM simulation was run in parallel with 16 processors, using dynamic RCB partitioning, and the artifact is constrained exactly using the method described in Section 3.3.2. The experimental and CCM model results for the impactor velocity, V_i, are illustrated in Figure 3-32.

The CCM model predicts the contact event occurs approximately 6 ms before the experimental data, primarily due to the slightly greater velocity prior to contact as a result of the open-loop input force applied to the impactor, coupled with the experimental velocity data sampling at 100 Hz yielding a $\pm 5ms$ uncertainty band due to aliasing in the experimental data. The CCM model predicts a shorter timescale contact event than the experimental data; resolving approximately 0.5 seconds sooner before the experimental data shows an increase in velocity after the contact event.

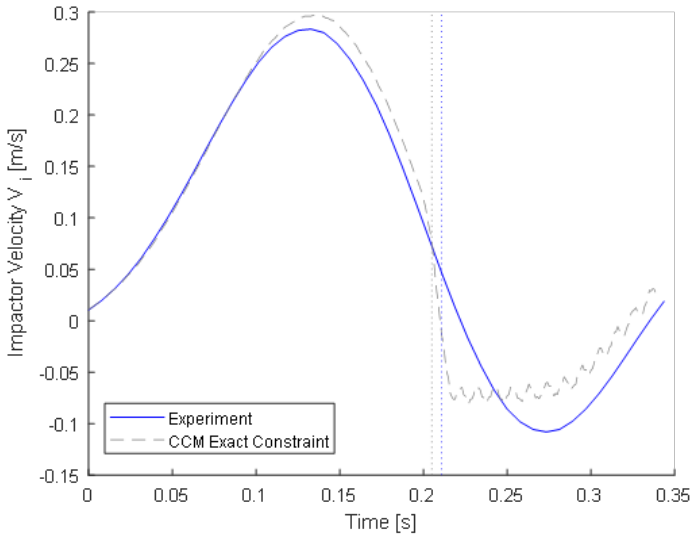


Figure 3-32: FX025 V_i Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time (grey dotted)

This result would suggest the CCM model should yield a faster artifact deflection, as the length of time of the contact event was reduced. To verify this, the artifact angle, θ_a, is shown in Figure 3-33.

As expected, the CCM model predicts a shorter period of time in which the artifact is displaced before returning to oscillate about zero. This higher frequency response is also expected, as the exactly fixed constraint at the center of rotation of the artifact ensures the CCM model of the artifact acts as a cantilevered beam. As there is no damping within the simulated artifact, the oscillations persist after the contact event. In the physical system, however, the frictional pin used to constrain the artifact cannot perfectly restrict all motion. As a result, some deflection occurs over a longer time scale, and extends the length of time in which the impactor is displacing the artifact, prior to the artifact pushing the impactor back. Futhermore, frictional losses in the physical pin joint, and within the artifact assembly itself, serve to dampen the response over time, resulting in less effective rigidity in the experiment than the simulation, as evidenced by the difference in the oscillatory periods post-impact in Figure 3-33. Based on this shorter time scale displacement response in the CCM model, it is expected that the resultant impact pressure on the artifact will

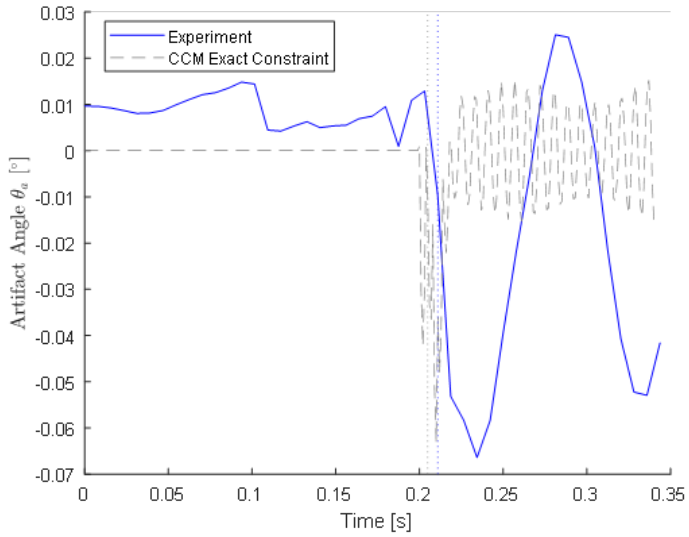


Figure 3-33: FX025 θ_a Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time (grey dotted)

be greater in magnitude and shorter in time. Figure 3-34 highlights the transient pressure response for the FX configuration results.

The CCM model highlights a response that is consistent with an effectively more rigid artifact configuration than the experimental results, as the resultant pressure acting on the artifact is both greater in magnitude and shorter in time scale. As noted previously, given the simulation perfectly constraining all degrees of freedom of the artifact's center of rotation, the apparent rigidity of the artifact is increased relative to the physical system which permits greater deflection. Additionally, the CCM model predicts two pressure spikes, indicating the simulation detected two contact events, while the experimental results indicate a single, prolonged contact event over the same length of time. However, relative to the Rigid Impactor (RX/RF) cases, the pressure magnitude aligns more closely with the experimental data, likely due to the flexibility of the impactor spring dominating the error introduced by the lack of an impactor friction model. While these results highlight differences between the CCM model results and the experimental data, the differences can likely be attributed to the physical system permitting some deflection in the artifact, while the CCM model exactly constrains the artifact center of rotation. Despite the need to tune the

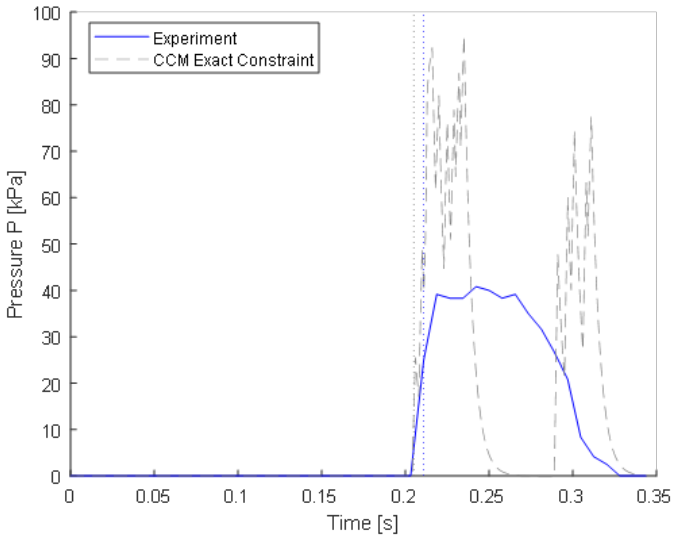


Figure 3-34: FX025 P Transient Results, Experimental (blue), CCM Model (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time (grey dotted)

boundary constraints acting on the artifact, the nearly identical maximum deflection of the artifact shown in Figure 3-33, combined with similar post-impact velocities in Figure 3-32 highlight that the CCM model resolves the contact event momentum change in a manner consistent with experimental data. Similar to the conclusions noted for the Rigid Impactor, Fixed Artifact configuration, the boundary conditions and constraints applied in the simulation should be tuned to improve the match to data, but the CCM model is capable of correctly resolving the kinetic energy response of the contact event for the Flexible Impactor, Fixed Artifact configuration.

3.4.4 Flexible Impactor, Free Artifact Configuration Results

The second impact test from the experimental data for the Flexible Impactor, Free Artifact, 0.25 m/s target velocity case (FF025) was selected as the trial to simulate with the CCM model. The CCM simulation was run in parallel with 16 processors, using dynamic RCB partitioning. The CCM model was run using the friction model described in Section 3.3.2 with $\mu_s = 0.4$ and $\mu_k = 0.7$, and was also run without friction applied to the artifact as well. The experimental and CCM model results for the impactor velocity, V_i, are illustrated in Figure 3-35.

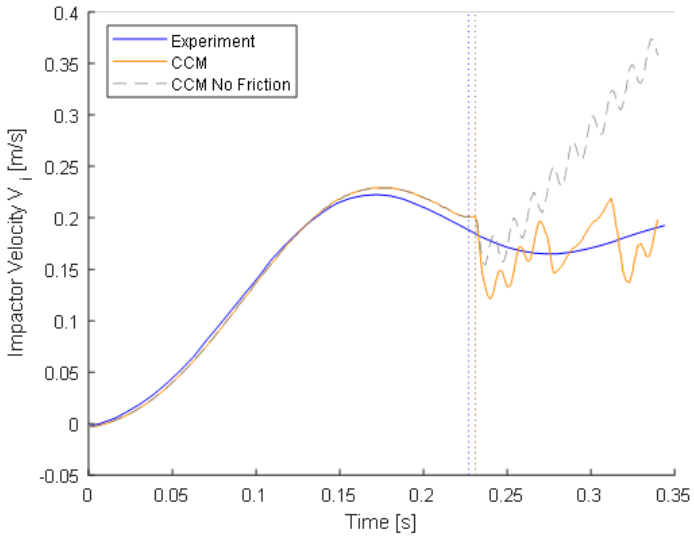


Figure 3-35: FF025 V_i Transient Results, Experimental (blue), CCM Model with artifact friction (orange), CCM Model without artifact friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time without artifact friction (grey dotted)

Relative to the other configurations, the CCM model with artifact friction coefficients of $\mu_s = 0.4$ and $\mu_k = 0.7$ yields much better match to the experimental data for V_i after the contact event. This improved alignment is likely due to the flexible impactor rod being the dominant hysteresis in the impactor system for the contact event (where for the rigid impactor, the unmodeled impactor race friction would be the only hysteresis or damping), coupled with the friction model enabling an improved match to the effective rigidity of the physical system. This is to say, that because both the impactor and artifact are compliant, the error introduced in the other simulations due to the lack of a friction model acting on the impactor is significantly reduced. This improvement is evident when comparing the post-impact velocity of the impactor for the CCM model with and without friction; with friction on the artifact, the impactor appears to continuously lightly impact the artifact after the first event. Without friction, it is expected the artifact deflected significantly after the first contact event, which allows the impactor to accelerate beyond the experimental results (as the artifact is out of the way). To corroborate this, Figure 3-36 illustrates the transient response of θ_a.

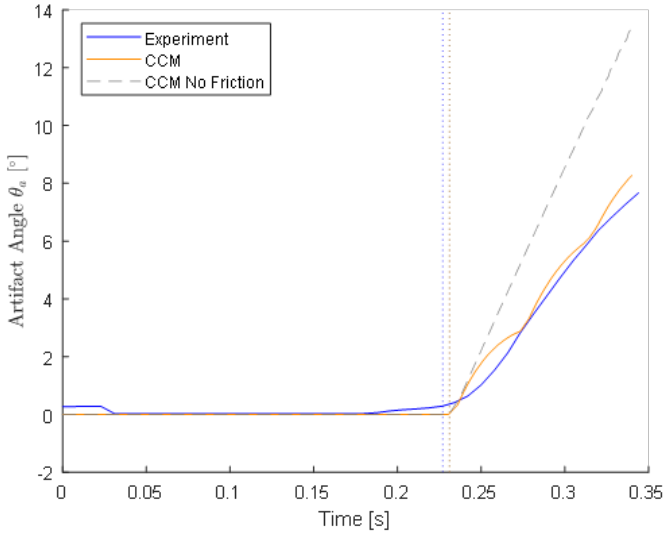


Figure 3-36: FF025 θ_a Transient Results, Experimental (blue), CCM Model with artifact friction (orange), CCM Model without artifact friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time without artifact friction (grey dotted)

The transient response of θ_a matches well between the CCM model with friction and the experimental data. The artifact does initially deflect more than the experimental result, which can be improved with tuning, but as additional smaller contact events ensue, the result approaches the experimental data. Additionally, as was expected, without friction applied the artifact deflects and moves out of the way of the impactor, enabling the unrestricted acceleration of the impactor in the frictionless case. Despite the relatively good agreement between the CCM model with friction and the experimental results, the CCM model appears to resolve a series of discrete contact events during the transient, while the experimental data appears to maintain one prolonged contact until approximately 0.32 seconds when the θ_a slope begins to decay. This would suggest that the sensed pressure, P, of the CCM model would experience a series of small pulses, while the experimental data will have one prolonged pressure event. Figure 3-37 illustrates the transient pressure response of the CCM model versus the experimental data.

The pressure response of the CCM model and experimental data corroborate the response observed in θ_a, with the experimental data experiencing a single, prolonged

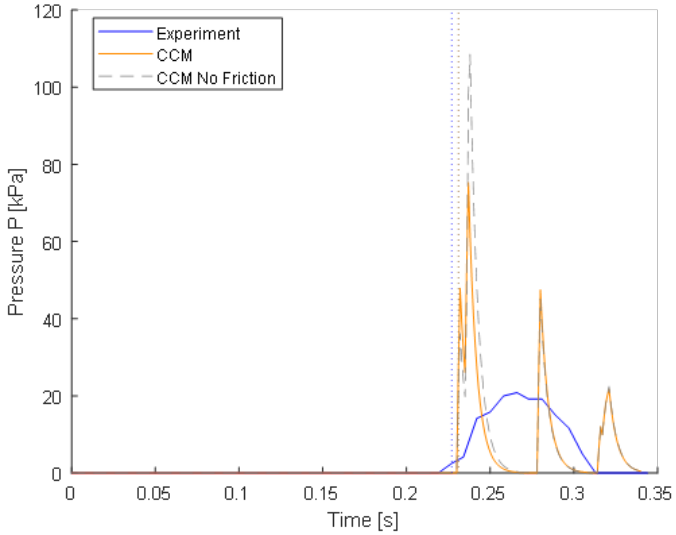


Figure 3-37: FF025 P Transient Results, Experimental (blue), CCM Model with artifact friction (orange), CCM Model without artifact friction (grey dashed), Experimental Contact Time (blue dash), CCM Contact Time with artifact friction (orange dotted), CCM Contact Time without artifact friction (grey dotted)

pressure event while the CCM model with friction experiences a series of discrete pressure events. Note that the CCM model without friction experiences only the first pressure event, albeit with slightly greater magnitude, and does not experience further pressure events, as the impactor and artifact do not come into contact again. Similar to the Flexible Impactor, Fixed Artifact (FX) test case, the pressure predicted by the CCM model exceeds that of the experimental data, but does align better than the Rigid Impactor (RX/RF) cases. This is likely attributed to the flexible spring in the impactor capturing a portion of the hysteresis present in the physical impactor. In both cases, the frictional losses acting on the impactor are not present, but the flexible spring is likely the dominant pole of the impactor system, and as such, provides an improved response. However, as with the other configurations, it should be stressed that the reduced impact of the unmodeled impactor friction is an effect of model tuning to match the physical system, just as the effect of incorporating friction on the artifact significantly improved the match to the experimental data. As V_i after the impact event tracks the experimental data even despite the lack of friction acting on the impactor, the CCM model does appear to correctly enforce contact in agreement

with the physical system for the Flexible Impactor, Free Artifact configuration as well.

3.5 Summary of Results

Four impactor-artifact contact events captured in experimental testing were used as the basis for simulations using the CCM model. The CCM model was run open-loop to match the experimental boundary conditions until the time of the first contact event. Cases involving a Rigid Impactor (RX/RF) experienced a shorter timescale, greater magnitude contact event than observed in the experimental data. Cases involving a Flexible Impactor (FX/FF) experienced similar shorter timescale, greater magnitude contact events in simulation than in the experimental data, but aligned more closely than the Rigid Impactor cases.

Despite the deviations in the timescale of the contact event, the position of the impacted artifact, θ_a, aligned in a manner consistent with the experimental data in terms of maximum post-impact amplitude. Tuning the response of the artifact using a friction model yielded significantly improved alignment, highlighting that the CCM model is resolving the contact response in a manner consistent with the experiment, but the overall simulation accuracy is contingent on the tuning of exogenous boundary conditions such as friction. Additionally, shorter timescale contact response in the CCM simulations is likely attributed to the lack of a friction model acting on the impactor, while the physical system experienced hysteresis and damping due to the impactor race. Given the improved alignment between the CCM model and experiment for cases with the Flexible Impactor over a Rigid Impactor, it is expected that the implementation of a friction model on the impactor would allow improved matching between the CCM simulations and experimental data.

These results primarily indicate that the CCM model is able to capture the effects of the contact events observed in experimental data for all four contact configurations, with improved alignment to Flexible Impactor configurations. However, to better realize the effects present in the physical simulation, friction models for the artifact

and impactor should be implemented and tuned, and such methods are recommended for detailed design implementations using the CCM model.

Chapter 4

Conclusions and Future Work

This chapter focuses on providing a summary of the capabilities of the Coupled Contact-Mechanics (CCM) model developed over the course of this work. The computational cost, scaling, and trade study versus a simple rigid-body approach are reviewed. Furthermore, the results of the validation study against experimental data are summarized, and conclusions regarding the capability, accuracy, and tuning of the CCM model are drawn. Finally, areas of work associated with the CCM model and the overall human-suit model are identified for future development and improvements.

4.1 Coupled Contact-Mechanics Model Capability

Two primary studies were performed to evaluate the capability, accuracy, and computational effectivity of the CCM model. First, a trade study against a simple rigid-body contact model was performed to benchmark the computational expense associated with using the CCM model versus a simpler method, and trading this computational cost against the relative accuracy of using the CCM model. Second, a validation study using experimental benchtop rig data provided a means by which to evaluate the overall accuracy and fidelity of the model against physical contact problem data, and also highlighted tuning parameters necessary to calibrate the model to a given physical system. This section provides an overview of the results of these studies, and draws conclusions regarding the CCM model from these results.

The trade study of the CCM model against a simple rigid-body contact model indicated that, for problems involving an impactor with a significantly higher rigidity than the artifact (on the order of 10^6 more rigid) the rigid-body contact model was equivalent to the CCM model within approximately 10% across the total simulation time, while boasting a 10^3 improvement in total runtime over the CCM model. Given that as the effective rigidity of the impactor increases, the rigid-body assumption approaches reality, and thus should be expected to yield an reasonably accurate result. Additionally, as the rigid body model represents the impactor as a geometric entity with no finite element discretization, it does not suffer from the need to reduce its stable time step for this highly stiff system as the CCM model must do to maintain simulation stability. However, this trend reverses as the relative rigidity of the impactor and artifact approach one another, with the CCM model yielding equivalent runtime with identical impactor and artifact rigidities, while boasting an improvement in normalized RSS error across the entire simulation on the order of 1%, with up to a 90% relative accuracy improvement at in the predicted state of the artifact at the end of the simulation. While the CCM model results are not necessarily "truth" to consider this a true accuracy measurement, the trends observed in the simulation results, such as those illustrated in Figure 2-21, highlight that the CCM model follows correct physical trends associated with a deformable impactor, while the rigid body model does not. Additionally, a set of weighting functions were developed that collapsed the simulation parameters of relative Young's Modulus ratio, ER, relative impactor length scale, RLS, and impactor kinetic energy, KE, into their effects on relative error to computational cost functions. These weighting function provide a single value that can be used to determine when the CCM model should be used over a simple rigid body model in terms of both error and total runtime. The end user can use such weighting functions to set a threshold for selecting a model based on the criticality of their current problem, and the full set of these results have been collected in Appendix B.

The back-to-back validation analyses between the CCM model and the benchtop data highlighted a few points.

- The CCM model predicted overall momentum changes in both the artifact and the impactor consistent with those observed in the experimental data arising from the contact event.

- The lack of damping or friction acting on the impactor boundary conditions resulted in a reduced time scale associated with the contact impulse in the CCM model results versus that of the experimental data.

- The use of a simple friction model on the artifact center of rotation enabled improved alignment between the CCM model results and experimental data, indicating a pathway to a data match with careful tuning of friction coefficients.

- The CCM model aligned best with data for the Flexible Impactor, Free Artifact (FF) configuration, and differed the most with data for the Rigid Impactor, Free Artifact (RF) configuration.

- While the CCM model is effective at resolving the contact event itself, the variation in results as a function of boundary conditions and constraints stresses the need for rigorous system identification when applying this model to physical systems. A recommended design methodology has been developed from the learnings of this study in Appendix E.

Leveraging the results of the trade study from Section 2.8, it is relevant to consider if the CCM model was appropriate for the contact domain selected for this validation study, or if a simple rigid body model would have been ideal. Furthermore, it is relevant to consider if the CCM model is appropriate for the domain of contact between the human and suit in general. To determine this, Figure 4-1 highlights where the validation test cases exist, along with a rough approximation of the domain of human-suit contact interactions could reside.

While the kinetic energy of the contact event may vary beyond or below 10 Joules, as currently illustrated in Figure 4-1, highlights that the CCM model is strongly recommended for resolving the RX/RF configurations, noted by the green dot, due to the low ER of those cases. The FX/FF cases, noted by the yellow dot, however, are

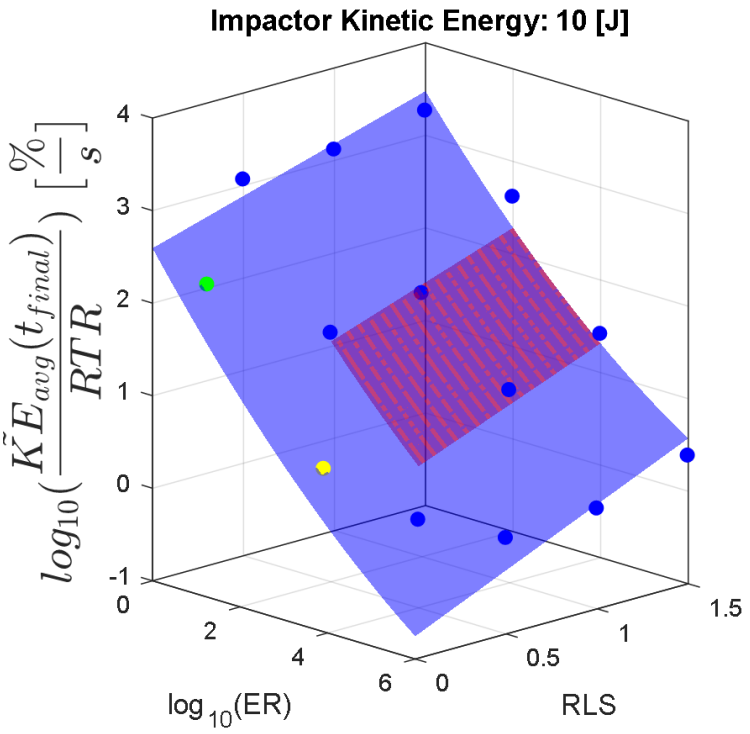


Figure 4-1: Overlay of RX025/RF025 contact domain (green dot), FX025/FF025 contact domain (yellow dot), and approximated human-suit interaction domain (purple surface), on the weighting function of average artifact kinetic energy error per runtime ratio (blue surface), with trade study simulation data (blue dots)

less recommended by the weighting function, yielding a sub 10 $\frac{\%}{s}$ error-to-cost weight. While this does not indicate that the CCM model is inappropriate for capturing such a case where the artifact had a higher Young's modulus than the flexible impactor, it does suggest that a simple rigid body model may have yield relatively equivalent results with reduced runtime. The purple surface on Figure 4-1 was approximated based on the relative Young's Moduli of human epidermis and S-Glass material yielding an ER around the order of 10^{10}. It should be noted that this type of human-suit contact involves human interaction with the hardgood material region of a suit, and is expected to have a higher ER than those interactions between a human and pressurized softgood suit material. With this caveat in mind, the weighting function suggests that for small, localized regions of contact between the human and suit, a rigid body method may be viable, but as the contact region increases in size, or the relative ER decreases, the CCM model becomes increasingly more effective. Given the higher

effectivity of the CCM model for these rigid suit, deformable human interactions, it follows that for deformable softgood suit, deformable human interactions, which would have lower ER, the CCM model is significantly more effective than a simple rigid body model.

While the CCM model has demonstrated capability to capture physical contact events with reasonable accuracy, and has been compared against a simple rigid body model as a benchmark, significant work beyond that presented in this thesis remains, as is discussed in Section 4.2.

4.2 Future Work

Over the course of this work, three specific issues were noted that merit further investigation and development in the future to enable the full, integrated human-suit model architecture. This section details these issues, and provides recommendations for future work in each of these domains accordingly.

4.2.1 CCM-Musculoskeletal Model Pipeline Development

While not necessarily an issue germane to the development or improvement of the CCM model itself, one aspect of the overall integrated-human suit model that was not developed extensively over the course of this work was the integration between the CCM model and OpenSim. This is not to say that a connectivity between the SUMMIT-based CCM model and OpenSim does not exist. At the onset of this work, previous efforts had established a means of coupling the results of a CCM simulation, which computed exogenous torques, with a dynamic OpenSim simulation using SimBody. Furthermore, the software developed over the course of this work has consistently aimed to maintain the ability to compile with and link to libraries germane to the OpenSim environment.

However, the above capabilities represent the limits reached over the course of this work. To fully demonstrate the entire integrated human-suit model, more detailed test cases should be performed involving a human-artifact or simplified human-suit

transient contact problem, and the results of these CCM model simulations provided to OpenSim via the developed coupling methods to exercise the model and identify any limitations or issues. Furthermore, the high-level aim of this work is to satisfy that noted in Chapter 1: Development of a coupled finite element and musculoskeletal model of the integrated human-suit system; demonstrating the capability of the entire model is germane to satisfying this claim. Human-suit testing with in-suit pressure measurements should be used to further develop and calibrate the model.

This leads to a tangential discussion regarding the modeling of EVA suits such as the Mark III or Demonstrator suit. In Chapter 2, it was noted that the nonlinear Neohookean material model was employed as the constitutive law during this work. While the Neohookean model is able to resolve large deformations without generating physically unrealistic deformations that would arise from a linear elastic model, special attention should be called to the selection of a constitutive model for the soft-good EVA suit materials. Material characterization testing should be performed on individual sections of the suits, and the resulting stress-strain relationships used to determine if an alternative constitutive model should be selected, such as a superelastic or orientation-dependent model. This selection and validation of a representative material model falls under the component calibrations highlighted in Figure 1-4, and is a necessary step in tuning the model for a given human-suit application.

4.2.2 Integrated Minimum Stable Time Step

The most pressing issue identified during the course of this work was identified during the CCM model validation testing discussed in Section 3.4. In the current CCM model PEC algorithm architecture noted in Section 2.5, the minimum stable time step used for the explicit integration of the model is set by the structural mechanics solver. For problems involving high internal energy, or sufficiently small minimum element sizes (the two primary drivers of the CFL number for structural mechanics), the stable time step of the system is sufficiently small as to resolve the introduced contact impulses driven by the nodal "push back" effects of the DCR method. However, for coarse meshes involving problems with high kinetic energy contact events, the minimum

stable time step of the system was found to be insufficient to maintain stability of the explicit simulation. For all cases noted in Section 3.1 involving impactor velocities of 0.50 m/s or greater, the moment the impactor contacted the artifact, the simulation became unstable and experienced segmentation faults, resulting in the simulation failing.

Initially, extensive debugging was performed to attempt to identify improper memory allocations between the DCR and SUMMIT domains through the Contact Surface object, but memory testing performed via Valgrind [71] did not highlight any such software issues. Additionally, the partitioning method was toggled between Equal Elements and RCB, and the number of processors varied from serial to up to 26 to verify these stability issues were not the result of these functions. The root cause being the minimum stable time step was finally isolated when the stable time step of the overall simulation was heuristically slowed, until one of the higher velocity test cases was able to converge. However, as the time step had been slowed heuristically, and was not computed at runtime based on the current state of the simulation, the computational expense of the CCM model for these cases was exceptionally high.

To address this, a proper means of communicating the minimum stable time step between the SUMMIT structural and DCR contact domains is recommended. To this end, some potential strategies are proposed:

1) Provide the minimum structural element size to the Contact Surface object to compute a worst-case time step in the DCR domain

The basis for this method is that, by providing the pacing element length from structural mechanics, the Contact Surface can compute a worst-case expectation for the stable time step by assuming that, as a worst-case assumption, this minimum size element may deform by the full amount of the maximum velocity magnitude within the Contact Surface multiplied by the current time step, or

$$C_{pacing} = \frac{\dot{\boldsymbol{u}}\Delta t}{L_{min}} \tag{4.1}$$

where $\dot{\boldsymbol{u}}$ is the maximum velocity vector in the Contact Surface, Δt is the current time step size, L_{min} is the minimum structural element size, and C_{pacing} is the resulting celerity that paces the CFL number for the simulation. From this minimum celerity, the minimum stable time step assuming a purely elastic collision can be computed. This is an extremely conservative approach, as the worst-case deformation is unlikely to occur along the maximum velocity vector of a Contact Surface node on the minimum size element at every given time step. However, this provides a possible starting point for enabling a means for the Contact Surface to inform the PEC algorithm of potential energy introduction due to a worst-case contact impulse and enable a stable solution. Furthermore, this method could be adapted to activate or linearly scale in as a function of the distance between nodes in the contact search, essentially only slowing the solution as the time step refinement is necessary.

2) Run the Contact Surface twice for a feedforward time step computation

The terminology of "feedforward" is roughly adapted from control theory, where a feedback controller is provided an open-loop expectation of a future time step to improve transient performance. In this instance, the Contact Surface search and enforcement steps would be run at time i and i+1, with the feedforward expectation of i+1 assuming no structural mechanics update. Using this feedforward expectation of the Contact Surface state, the structural mechanics system can use the solution from time i and i+1 to compute a contact force delta, through differentiation of the Contact Surface velocities and masses. This contact force based on the feedforward expectation can then be used to compute an expected minimum stable time step for the structural mechanics simulation, slowing it as the Contact Surface projects increased deformations into the future. This method provides a means by which to give the structural mechanics time step the ability to "prepare" for a sudden contact impulse introducing energy to the simulation, and does so only when contact is imminent within two time steps. However, unlike the first proposed approach, there is no guarantee that the computed stable time step will be sufficient. If the deformations predicted by structural mechanics at time i+1 would yield an increase in

the energy introduced to the system through a contact impulse, then the feedforward expectation would be insufficient to capture this effect.

The two concepts noted above have not yet been implemented in the SUMMIT Contact Surface object methods, and they are not intended to represent the only viable means by which the minimum stable time step can be effectively calculated and communicated between the structural and contact domains. However, it should be stressed that a viable solution to the stability issue is necessary to practicably scale the CCM model into more complex problems without the need for heuristic tuning of the time step scaling.

4.2.3 Contact Surface Parallelization Improvements

In Section 2.7.3, it was found that the parallelization scheme used for the Contact Surface was suboptimal, with the benefit of additional processors falling off and reversing beyond approximately 10 processors. The root cause of this reversal in effectiveness is due to the need to gather the full, global Contact Surface entity at each time step from every processor prior to communication from SUMMIT to DCR. The cost of gathering, updating, and communicating the global surface quickly paces the simulation, as the advantage of partitioning fades as the global system must be reconstructed on each processor. However, caution must be exercised when attempting to correct for this weakness - the SUMMIT structural mechanics and DCR contact mechanics domains have different partitions intentionally, to balance the workload evenly for both problems, and without the need to dedicate processors to only structural mechanics, or only contact enforcement. As a result, each partition must have the necessary information for its domain available at each iteration, but the structural mechanics partition on a given processor may have no commonality with the contact mechanics partition.

To this end, a solution is proposed: the development of a memory map between the SUMMIT structural mechanics partition and the DCR contact mechanics partition on each processor. While this does result in the need to generate a new data structure on each processor, this memory map would scale as a function of the total

element count in a given partition divided by the number of processors, ensuring scaling with processor count. This memory map would need to contain information that indicates which contact domain node corresponds with which structural mechanics domain node, and furthermore, which processor each of these nodes resides on. This will result in the creation of an abstract "processor boundary" for each processor, and the number of processors along a given boundary may be uneven. However, by informing each processor of which processors contain the structural or contact domain data relevant to its current domain, the full contact surface does not need to be reconstructed on each processor in each time step. For a static, Equal Elements partitioning scheme of both structural mechanics and contact mechanics domains, this is process can be performed once, and is theoretically straightforward to construct. However, the problem becomes more complex when dynamic load balancing, such as the dynamic RCB partitioning method, is used as the partition will update as the simulation progresses. This would require the memory map be updated transiently as well to maintain consistent mapping between domains. However, if such a memory map can be successfully constructed, the parallel scaling of the Contact Surface methods should be improved.

4.2.4 Large Deformation Contact Problems

Classical FEM methods perform well for problems involving relatively small deformations of the initial mesh entities, but begin to break down as individual elements begin to significantly distort; this reduces mesh quality and risks both simulation accuracy and stability as the element Jacobians warp. The class of problems considered in this work involved primarily small deformations, but for highly deformable softgood EVA suit designs, large deformation problems will likely dominate the problem domain.

To alleviate these expected issues, a variety of domain remeshing or reformulations can be applied to correct for element distortion, including continuous adaptive remeshing [86], or the conversion to a particle-based formulation [47,87]. Continuous adaptive remeshing is the process of pausing the simulation after a set time scale, or potentially an external trigger such as mesh quality dropping below a threshold, and

remeshing the global problem domain before proceeding. While the remeshing process directly addresses the mesh quality problem, it is expensive and not well-suited for parallel computation, as the remeshing process must occur in serial to re-establish the discretization of the global domain.

Converting to a particle-based formulation, particularly the class of formulation known as peridynamics [87], is an alternative method that should be investigated for application to the CCM model. In the peridynamic formulation, the finite elements are eliminated, and replaced by a cloud of nodes with continuous range-limited interaction laws applied between the nodes. As the elimination of the finite elements removes the low quality elements entirely, it is robust to resolving large deformation problems. This formulation has proven particularly attractive for problems involving discontinuities and fracture [47], and could provide a similar advantage to large deformation problems addressed by the CCM model. Furthermore, while the CCM model does enforce contact through the concept of triangular surface elements, it is already well-aligned to communicate with an arbitrary structural mechanics formulation, represented by finite elements or peridynamics, as beyond surface detection, it behaves similarly to a particle-based method in contact enforcement.

4.3 Summary of Contributions

While additional work remains to further improve the CCM model and enable the complete integrated human-suit modeling system, significant capabilities have been enabled, improved and validated over the course of this work. To briefly summarize these contributions to deformable human-suit modeling:

- Established a pipeline for the integrated human-suit modeling system, trading architectures from high-level requirements to select a connectivity capable of supporting EVA suit design problems.

- Restored legacy code for the implementation of the DCR method; this involved correcting memory leaks, updating to a new build system to enable external

linking and successful compilation, and code optimization.

- Developed a standardized interface Contact Surface class to enable interaction between SUMMIT and the DCR method, capable of both serial and parallel computation schemes across the SUMMIT and DCR domains using MPI, METIS and Zoltan, and implemented automated unit testing for these methods.

- Created conversion methods between Universal I-DEAS and SUMMIT files to enable the CAD-SUMMIT connectivity of the human-suit modeling pipeline.

- Successfully implemented a Predict-Evaluate-Correct Newmark algorithm to couple structural mechanics with DCR contact enforcement to form the Coupled Contact-Mechanics (CCM) model capable of both external contact and self-contact enforcement.

- Performed scalability studies on the CCM model using static and dynamic partitioning schemes, identifying the current scalability of the CCM model in parallel.

- Identified error versus computational cost trades between the explicit DCR method and a simplified rigid contact penalty method, which both corroborated the CCM model for rigid body contact problems, and provided a means to aid the end user in selection of an appropriate modeling method for non-rigid problems, across a range of normalized problem parameters.

- Verified the accuracy and capability of the CCM model against experimental benchtop testing for a variety of contact types, which demonstrated the capability of the CCM model to enforce contact consistently with the experimental data, while stressing the significance of tuning exogenous boundary conditions of the model.

Appendix A

SUMMIT::ContactSurface Detailed Code Description and Documentation

summit::ContactSurface Class Reference

#include <contact_surface.h>

Public Member Functions

	ContactSurface (const **summit::Mesh** &**mesh**, const **summit::DynamicsSystem** &sys)
virtual	**~ContactSurface** ()
void	**Build** (const **summit::Boundary** &boundObj, **summit::MPISummit** &mpiSummit)
void	**UpdateMech** (const **summit::ExplicitContactIntegrator** &solv, const **summit::DynamicsSystem** &sys, const **summit::CommunicationManager** &commMan, const **summit::MPISummit** &mpiSummit)
void	**UpdateContactForce** (**summit::NodalField**< **real** > &dContactForces, double timeStep, **summit::CommunicationManager** &commMan)
void	**UpdateContactDisp** (**summit::NodalField**< **real** > &contactDisp, **summit::NodalField**< **real** > &contactVelo)
void	**RegisterContact** (double tolerance, double friction, double restitution, std::string &orientation)
void	**RegisterPredictor** ()
void	**SetTimeStep** (double dt)
void	**EnforceContact** ()
void	**BuildCSME** ()
int	**getGlobalNodeID** (int i)
int	**getLocalNodeID** (int i)
int	**getProcNum** (int i)
void	**writeMesh** (const std::string &filename)
int	**nodes** ()
int	**faces** ()
double *	**coords** ()
void	**Partition** (**summit::MPISummit** &mpiSummit)
void	**RegenGlobalContact** (**summit::MPISummit** &mpiSummit)

Public Attributes

int	**nNodes**
int	**nFaces**
double *	**Coordinates**
int *	**Connectivities**
double *	**Displacement**
double *	**Velocity**
double *	**Residual**
double *	**Mass**

Private Member Functions

void	**GetBoundData** (std::vector< **MeshEntity** *> const &elementContainer, std::vector< std::vector< double > > &uniqueVertices, std::vector< std::vector< int > > &arrayConnectivity)
int	**mapLocalNodeID** (const int globalID, **summit::OuterBoundary** &boundary, std::vector< std::vector< **real** > > &vertices)
int	**findLocalNodeID** (const int globalID, std::vector< std::vector< **real** > > &localNodes, std::vector< std::vector< **real** > > &globalNodes)
int	**findGlobalNodeID** (const int localID, std::vector< std::vector< **real** > > &localNodes, std::vector< std::vector< **real** > > &globalNodes)

Private Attributes

const **summit::Mesh** &	**_mesh**
const **summit::DynamicsSystem** &	**_sys**
std::vector< std::vector< **real** > >	**_vertices**
std::vector< std::vector< **real** > >	**_lVertices**
std::vector< std::vector< **real** > >	**_gVertices**
std::vector< int >	**_globalNodalIDMap**
std::vector< int >	**_localNodalIDMap**
contact::Contact *	**_con**
int *	**_procNum**

Detailed Description

Class designed as the interface and communicator between Summit solid mechanics and DCR explicit contact systems in serial and parallel

Definition at line **34** of file **contact_surface.h**.

Constructor & Destructor Documentation

◆ ContactSurface()

summit::ContactSurface::ContactSurface (const **summit::Mesh** & mesh,
const **summit::DynamicsSystem** & sys
)

Constructor

Parameters

[in] **mesh** the partitioned Summit mesh object

[in] **sys** the **DynamicsSystem** object used for transient solid mechanics

Definition at line **45** of file **contact_surface.cc**.

References **_procNum**, **Connectivities**, **Coordinates**, **Displacement**, **Mass**, **nFaces**, **nNodes**, **Residual**, **Velocity**, and **~ContactSurface()**.

◆ ~ContactSurface()

summit::ContactSurface::~ContactSurface ( ) virtual

Destructor

Definition at line **67** of file **contact_surface.cc**.

References **_procNum**, **Connectivities**, **Coordinates**, **Displacement**, **mapLocalNodeID()**, **Mass**, **Residual**, and **Velocity**.

Referenced by **ContactSurface()**.

Member Function Documentation

◆ Build()

void summit::ContactSurface::Build (const **summit::Boundary** & boundObj,
summit::MPISummit & mpiSummit
)

Function designed to construct the contact surface from the local partition's discretized function space and instantiate a DCR contact object.

Parameters

[in] **boundObj** the full (not partitioned) boundary object to build the contact surface from

[in] **mpiSummit** the mpi communicator used by the Summit system

Function to complete the contact surface given the information provided in the system object only. This method uses the **DiscretizedBoundary** object to populate the CS, as opposed to the BuildCSME method, which uses the **Mesh** object itself.

Definition at line **214** of file **contact_surface.cc**.

References **_globalNodalIDMap**, **_gVertices**, **_localNodalIDMap**, **_lVertices**, **_mesh**, **_procNum**, **_sys**, **summit::Boundary::begin()**, **Connectivities**, **Coordinates**, **summit::FunctionSpace::coordinates()**, **summit::FunctionSpace::dim()**, **summit::FunctionSpace::DiscrNodesForFace()**, **Displacement**, **summit::Boundary::end()**, **findGlobalNodeID()**, **findLocalNodeID()**, **summit::System::function_space()**, **summit::OuterBoundary::GenerateBoundary()**, **summit::NodalField< T >::getArray()**, **GetBoundData()**, **Mass**, **nFaces**, **nNodes**, **summit::NodalField< T >::nodes()**, **Residual**, **summit::Point< dim >::set()**, **summit::Boundary::size()**, **UpdateMech()**, and **Velocity**.

Referenced by **findGlobalNodeID()**, and **TEST()**.

◆ BuildCSME()

void summit::ContactSurface::BuildCSME ( )

Builder function using **Mesh** object NOTE: This function is deprecated - only **Build()** should presently be used to construct the contact surface. This method will be eliminated in a future revision.

Function to complete the contact surface given the information provided in the mesh and system objects.

Definition at line **625** of file **contact_surface.cc**.

References **_mesh**, **_vertices**, **summit::Boundary::boundaryElement()**, **Connectivities**, **Coordinates**, **summit::MeshEntityIterator::end()**, **summit::OuterBoundary::GenerateBoundary()**, **mapLocalNodeID()**, **nFaces**, **nNodes**, **summit::Boundary::size()**, and **writeMesh()**.

Referenced by **SetTimeStep()**.

◆ coords()

double* summit::ContactSurface::coords ( ) inline

Definition at line **174** of file **contact_surface.h**.

References **summit::vector()**.

◆ EnforceContact()

void summit::ContactSurface::EnforceContact ( )

Routine which removes collisions found along the contact surface using the DCR method

Function to enforce the geometric impenetrability constraint on the contact surface object

Definition at line **582** of file **contact_surface.cc**.

References **_con**, **Displacement**, **nNodes**, **Residual**, **SetTimeStep()**, and **Velocity**.

Referenced by **RegisterPredictor()**, and **TEST()**.

◆ faces()

int summit::ContactSurface::faces ( ) inline

Definition at line **173** of file **contact_surface.h**.

◆ findGlobalNodeID()

int summit::ContactSurface::findGlobalNodeID (const int localID,
std::vector< std::vector< **real** > > & localNodes,
std::vector< std::vector< **real** > > & globalNodes
) private

Definition at line **175** of file **contact_surface.cc**.

References **Build()**.

Referenced by **Build()**, and **findLocalNodeID()**.

◆ findLocalNodeID()

```
int summit::ContactSurface::findLocalNodeID ( const int                                globalID,
                                               std::vector< std::vector< real > > &    localNodes,
                                               std::vector< std::vector< real > > &    globalNodes
                                              )                                                    private
```

Definition at line **141** of file **contact_surface.cc**.

References **findGlobalNodeID()**.

Referenced by **Build()**, and **mapLocalNodeID()**.

◆ GetBoundData()

```
void summit::ContactSurface::GetBoundData ( std::vector< MeshEntity *> const &    elementContainer,
                                             std::vector< std::vector< double > > &  uniqueVertices,
                                             std::vector< std::vector< int > > &     arrayConnectivity
                                            )                                                    private
```

Extracts the unique vertices and connectivites from a given **summit::Boundary** object

Parameters

[in]	**elementContainer**	the filtered boundary object that contains only elements within the contact surface boundary
[in,out]	**uniqueVertices**	a 2D vector which is populated with the vertices in the contact surface boundary
[in,out]	**arrayConnectivity**	a 2D vector which is populated with the connectivity of the nodes within the contact surface boundary

Definition at line **735** of file **contact_surface.cc**.

References **_sys**, **summit::FunctionSpace::coordinates()**, **summit::FunctionSpace::dim()**, **summit::FunctionSpace::DiscrNodesForEdge()**, **summit::FunctionSpace::DiscrNodesForFace()**, **summit::System::function_space()**, **summit::NodalField< T >::getArray()**, and **RegenGlobalContact()**.

Referenced by **Build()**, and **writeMesh()**.

◆ getGlobalNodeID()

int summit::ContactSurface::getGlobalNodeID (int i) inline

Helper function that returns a global (solid mechanics) nodal ID for a given contact surface local nodal ID

Parameters

[in] **i** the contact surface nodal ID number return the corresponding solid mechanics system nodal ID number, -1 if not found

Definition at line **147** of file **contact_surface.h**.

Referenced by **TEST()**.

◆ getLocalNodeID()

int summit::ContactSurface::getLocalNodeID (int i) inline

Helper function that returns a contact surface nodal ID from a solid mechanics nodal ID, -1 if the corresponding solid mechanics node is not found

Parameters

[in] **i** the solid mechanics system nodal ID number return the corresponding contact surface nodal ID number, -1 if not found

Definition at line **155** of file **contact_surface.h**.

Referenced by **TEST()**.

◆ getProcNum()

int summit::ContactSurface::getProcNum (int i) inline

Helper function that returns the partition number which is responsible for the solid mechanics of the given contact surface nodal ID

Parameters

[in] **i** the contact surface nodal ID return the corresponding processor number that contains the contact surface node in its solid mechanics partition

Definition at line **163** of file **contact_surface.h**.

◆ mapLocalNodeID()

int summit::ContactSurface::mapLocalNodeID (const int globalID,
summit::OuterBoundary & boundary,
std::vector< std::vector< **real** > > & vertices
) private

Function to generate the mapping from the full mesh nodal IDs to the contact surface nodal IDs

Definition at line **86** of file **contact_surface.cc**.

References **summit::Boundary::boundaryElement()**, **summit::Mesh::dim()**, **summit::MeshEntityIterator::end()**, **findLocalNodeID()**, **summit::Boundary::mesh()**, and **summit::Boundary::size()**.

Referenced by **BuildCSME()**, and **~ContactSurface()**.

◆ nodes()

int summit::ContactSurface::nodes ( ) inline

Definition at line **172** of file **contact_surface.h**.

◆ Partition()

void summit::ContactSurface::Partition (**summit::MPISummit** & mpiSummit)

Definition at line **553** of file **contact_surface.cc**.

References **_con**, **summit::MPISummit::mpiComm()**, and **RegisterPredictor()**.

Referenced by **RegisterContact()**, and **TEST()**.

◆ RegenGlobalContact()

void summit::ContactSurface::RegenGlobalContact (**summit::MPISummit** & mpiSummit)

Function to regenerate the global boundary for the contact surface from the partitions when running in parallel

Definition at line **802** of file **contact_surface.cc**.

References **_globalNodalIDMap**, **_localNodalIDMap**, **_procNum**, **_sys**, **Connectivities**, **Coordinates**, **summit::FunctionSpace::coordinates()**, **summit::FunctionSpace::dim()**, **summit::FunctionSpace::DiscrNodesForFace()**, **Displacement**, **summit::System::function_space()**, **summit::NodalField< T >::getArray()**, **Mass**, **summit::MPISummit::mpiComm()**, **nFaces**, **nNodes**, **summit::NodalField< T >::nodes()**, **summit::MPISummit::rank()**, **Residual**, **summit::MPISummit::size()**, and **Velocity**.

Referenced by **GetBoundData()**, and **TEST()**.

◆ RegisterContact()

void summit::ContactSurface::RegisterContact (double tolerance,
double friction,
double restitution,
std::string & orientation
)

Routine which registers a DCR contact mesh object with the contact methods from a summit contact surface

Parameters

[in]	**tolerance**	the tolerance for enforcing contact about the contact surface
[in]	**friction**	sets frictional loss coefficients to use during contact
[in]	**restitution**	toggle which dictates purely elastic contact [2] or plastic contact [1]
[in]	**orientation**	tag which indicates the direction from which to enforce normal direction from the contact surface

Function to register a new contact surface mesh with the contact methods

Definition at line **543** of file **contact_surface.cc**.

References **_con**, **Connectivities**, **Coordinates**, **Mass**, **nFaces**, **nNodes**, and **Partition()**.

Referenced by **TEST()**, and **UpdateContactDisp()**.

◆ RegisterPredictor()

void summit::ContactSurface::RegisterPredictor ( )

Routine which registers the displacements and velocities from the predictor step with the DCR contact method

Function to register the displacements and velocities from the predictor step with the contact methods

Definition at line **569** of file **contact_surface.cc**.

References **_con**, **Displacement**, **EnforceContact()**, **nNodes**, and **Velocity**.

Referenced by **Partition()**.

◆ SetTimeStep()

void summit::ContactSurface::SetTimeStep (double dt)

Function to set the time step of the contact resolver method

Definition at line **614** of file **contact_surface.cc**.

References **_con**, and **BuildCSME()**.

Referenced by **EnforceContact()**, and **TEST()**.

◆ UpdateContactDisp()

void summit::ContactSurface::UpdateContactDisp (**summit::NodalField**< **real** > & contactDisp,
summit::NodalField< **real** > & contactVelo
)

Routine which takes in the nodal fields for displacement and velocity from the solid mechanics system and updates the fields for nodes in the contact surface. Should only be called after **EnforceContact()**.

Parameters

[in,out]	**contactDisp**	nodal field that is updated with modified nodal displacements from the contact resolver solution.
[in,out]	**contactVelo**	nodal field that is updated with modified nodal velocities from the contact resolver solution.

Function to update the solid mechanics system neumann boundary values (displacements), and return a nodal field containing updated nodal displacements to be applied to the solid mechanics system boundary conditions on the next update.

Definition at line **513** of file **contact_surface.cc**.

References **_localNodalIDMap**, **summit::NodalField< T >::dim()**, **Displacement**, **summit::NodalField< T >::nodes()**, **RegisterContact()**, and **Velocity**.

Referenced by **TEST()**, and **UpdateContactForce()**.

◆ UpdateContactForce()

void summit::ContactSurface::UpdateContactForce (**summit::NodalField**< **real** > & dContactForces,
double timeStep,
summit::CommunicationManager & commMan
)

Routine which returns updated delta dirichlet-type boundary conditions for the solid mechanics system with information from the contact resolver. Should only be called after **EnforceContact()**. NOTE: This formulation is currently deprecated in favor of UpdateContactDisp, and may be eliminated in a future revision.

Parameters

[in,out]	**dContactForces**	nodal field that is updated with modified nodal forces from the contact resolver.
[in]	**timeStep**	the current solid mechanics system time step
[in]	**commMan**	the Summit communication manager, used for accessing nodal fields

Function to update the solid mechanics system dirichlet boundary values (forces), and return a nodal field containing updated nodal forces to be applied to the solid mechanics system boundary conditions on the next update.

Definition at line **474** of file **contact_surface.cc**.

References **_globalNodalIDMap**, **_sys**, **summit::FunctionSpace::dim()**, **summit::System::function_space()**, **summit::DynamicsSystem::GetNodalField()**, **Mass**, **summit::FunctionSpace::nodes()**, **UpdateContactDisp()**, and **Velocity**.

Referenced by **UpdateMech()**.

◆ UpdateMech()

```
void summit::ContactSurface::UpdateMech ( const summit::ExplicitContactIntegrator & solv,
                                          const summit::DynamicsSystem &            sys,
                                          const summit::CommunicationManager &      commMan,
                                          const summit::MPISummit &                 mpiSummit
                                        )
```

Routine which updates the contact surface with information from the solid mechanics solver

Parameters

[in]	**solv**	the Summit solid mechanics solver object, used to access the residual
[in]	**sys**	the Summit solid mechanics system object, used to access nodal fields
[in]	**commMan**	the Summit communication manager, used to access nodal fields
[in]	**mpiSummit**	the Summit mpi communicator, used for parallel explicit contact

Function to initialize/update the contact surface nodal quantities of displacement, velocity, residual and mass. Should only be called after the solver has been initialized and boundary conditions applied so that the nodal quantities will have been updated.

Definition at line **375** of file **contact_surface.cc**.

References **_globalNodalIDMap**, **_procNum**, **_sys**, **summit::FunctionSpace::dim()**, **Displacement**, **summit::System::function_space()**, **summit::NodalField< T >::getArray()**, **summit::DynamicsSystem::GetNodalField()**, **Mass**, **summit::MPISummit::mpiComm()**, **nNodes**, **Residual**, **summit::Solver::residual()**, **summit::MPISummit::size()**, **UpdateContactForce()**, and **Velocity**.

Referenced by **Build()**, and **TEST()**.

◆ writeMesh()

```
void summit::ContactSurface::writeMesh ( const std::string &  filename )
```

Writes a summit mesh file that represents the contact surface as a linear shell

Parameters

[in]	**filename**	the name of the mesh file to write the mesh to

Function to generate a mesh file representing the contact surface object

Definition at line **701** of file **contact_surface.cc**.

References **_mesh**, **Connectivities**, **Coordinates**, **summit::Mesh::dim()**, **GetBoundData()**, **nFaces**, and **nNodes**.

Referenced by **BuildCSME()**.

Member Data Documentation

◆ _con

contact::Contact* summit::ContactSurface::_con

Definition at line **55** of file **contact_surface.h**.

Referenced by **EnforceContact()**, **Partition()**, **RegisterContact()**, **RegisterPredictor()**, and **SetTi**

◆ _globalNodalIDMap

std::vector<int> summit::ContactSurface::_globalNodalIDMap private

Definition at line **53** of file **contact_surface.h**.

Referenced by **Build()**, **RegenGlobalContact()**, **UpdateContactForce()**, and **UpdateMech()**.

◆ _gVertices

std::vector<std::vector<**real**> > summit::ContactSurface::_gVertices private

Definition at line **52** of file **contact_surface.h**.

Referenced by **Build()**.

◆ _localNodalIDMap

std::vector<int> summit::ContactSurface::_localNodalIDMap private

Definition at line **54** of file **contact_surface.h**.

Referenced by **Build()**, **RegenGlobalContact()**, and **UpdateContactDisp()**.

◆ _lVertices

std::vector<std::vector<**real**> > summit::ContactSurface::_lVertices private

Definition at line **51** of file **contact_surface.h**.

Referenced by **Build()**.

◆ _mesh

const **summit::Mesh**& summit::ContactSurface::_mesh private

Definition at line **48** of file **contact_surface.h**.

Referenced by **Build()**, **BuildCSME()**, and **writeMesh()**.

◆ _procNum

int* summit::ContactSurface::_procNum private

Definition at line **56** of file **contact_surface.h**.

Referenced by **Build()**, **ContactSurface()**, **RegenGlobalContact()**, **UpdateMech()**, and **~ContactSurface()**.

◆ _sys

const **summit::DynamicsSystem**& summit::ContactSurface::_sys private

Definition at line **49** of file **contact_surface.h**.

Referenced by **Build()**, **GetBoundData()**, **RegenGlobalContact()**, **UpdateContactForce()**, and **UpdateMech()**.

◆ _vertices

std::vector<std::vector<**real**> > summit::ContactSurface::_vertices private

Definition at line **50** of file **contact_surface.h**.

Referenced by **BuildCSME()**.

◆ Connectivities

int* summit::ContactSurface::Connectivities

Definition at line **39** of file **contact_surface.h**.

Referenced by **Build()**, **BuildCSME()**, **ContactSurface()**, **RegenGlobalContact()**, **RegisterContact()**, **TEST()**, **writeMesh()**, and **~ContactSurface()**.

◆ Coordinates

double* summit::ContactSurface::Coordinates

Definition at line **38** of file **contact_surface.h**.

Referenced by **Build()**, **BuildCSME()**, **ContactSurface()**, **RegenGlobalContact()**, **RegisterContact()**, **TEST()**, **writeMesh()**, and **~ContactSurface()**.

◆ Displacement

double* summit::ContactSurface::Displacement

Definition at line **41** of file **contact_surface.h**.

Referenced by **Build()**, **ContactSurface()**, **EnforceContact()**, **RegenGlobalContact()**, **RegisterPredictor()**, **TEST()**, **UpdateContactDisp()**, **UpdateMech()**, and **~ContactSurface()**.

◆ Mass

double* summit::ContactSurface::Mass

Definition at line **45** of file **contact_surface.h**.

Referenced by **Build()**, **ContactSurface()**, **RegenGlobalContact()**, **RegisterContact()**, **TEST()**, **UpdateContactForce()**, **UpdateMech()**, and **~ContactSurface()**.

◆ nFaces

int summit::ContactSurface::nFaces

Definition at line **37** of file **contact_surface.h**.

Referenced by **Build()**, **BuildCSME()**, **ContactSurface()**, **RegenGlobalContact()**, **RegisterContact()**, **TEST()**, and **writeMesh()**.

◆ nNodes

int summit::ContactSurface::nNodes

Definition at line **36** of file **contact_surface.h**.

Referenced by **Build()**, **BuildCSME()**, **ContactSurface()**, **EnforceContact()**, **RegenGlobalContact()**, **RegisterContact()**, **RegisterPredictor()**, **TEST()**, **UpdateMech()**, and **writeMesh()**.

◆ Residual

double* summit::ContactSurface::Residual

Definition at line **43** of file **contact_surface.h**.

Referenced by **Build()**, **ContactSurface()**, **EnforceContact()**, **RegenGlobalContact()**, **TEST()**, **UpdateMech()**, and **~ContactSurface()**.

◆ Velocity

double* summit::ContactSurface::Velocity

Definition at line **42** of file **contact_surface.h**.

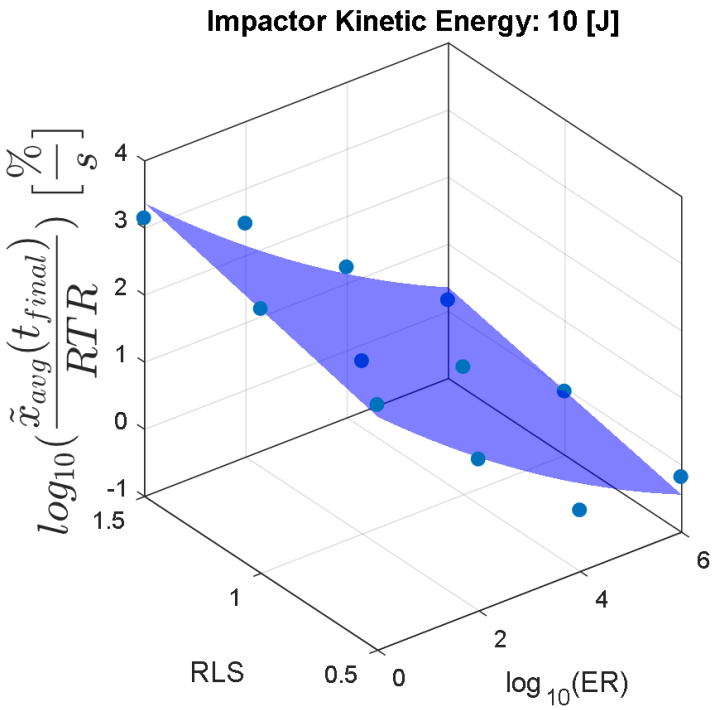
Referenced by **Build()**, **ContactSurface()**, **EnforceContact()**, **RegenGlobalContact()**, **RegisterPredictor()**, **TEST()**, **UpdateContactDisp()**, **UpdateContactForce()**, **UpdateMech()**, and **~ContactSurface()**.

The documentation for this class was generated from the following files:

- /home/cdking2/dv/summit/master/src/contact/sfc/**contact_surface.h**
- /home/cdking2/dv/summit/master/src/contact/sfc/**contact_surface.cc**

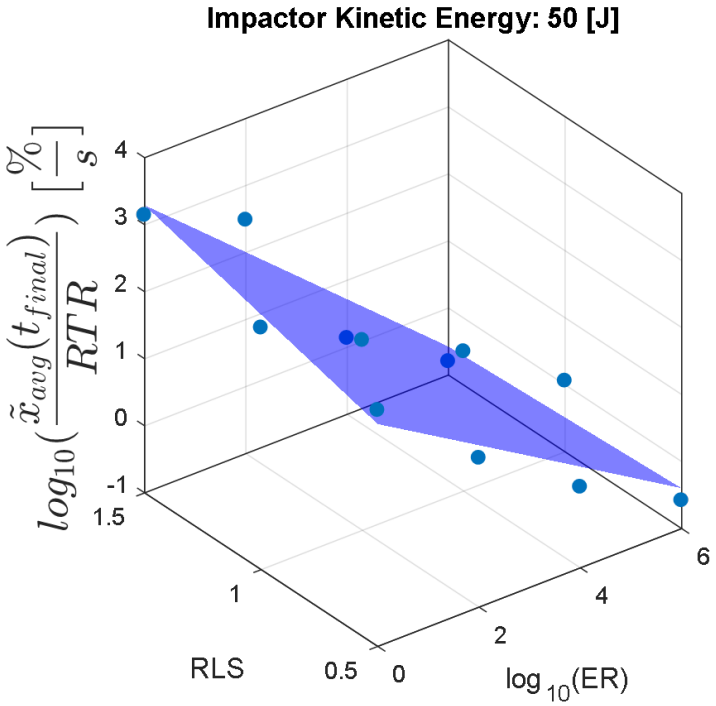
Appendix B

Trade Study Relative Error to Runtime Cost Results



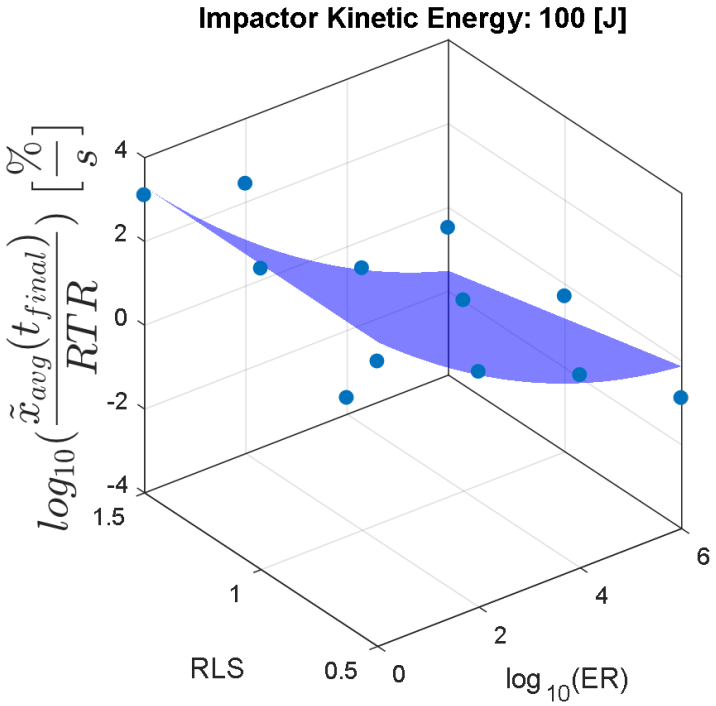
$$\log_{10}\left(\frac{\tilde{x}_{avg}(t_{final})}{RTR}\right) = 2.026 - 0.6419\log_{10}(ER) + 0.8965RLS - 0.02735\log_{10}(E\,R)^{\,2} - 0.01636\log_{10}(ER)RLS$$

Figure B-1: Error-Cost Weighting Function: Relative error in average displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 10 J kinetic energy



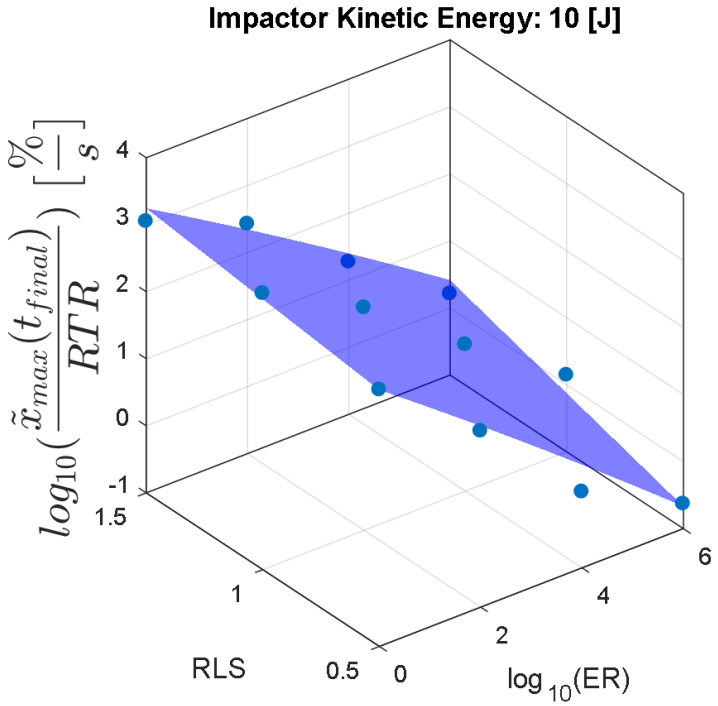
$$\log_{10}\left(\frac{\tilde{x}_{avg}(t_{final})}{RTR}\right) = 1.827 - 0.3539\log_{10}(ER) + 0.9747RLS - 0.0004026\log_{10}(E\,R)^{\,2} - 0.1919\log_{10}(ER)RLS$$

Figure B-2: Error-Cost Weighting Function: Relative error in average displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 50 J kinetic energy



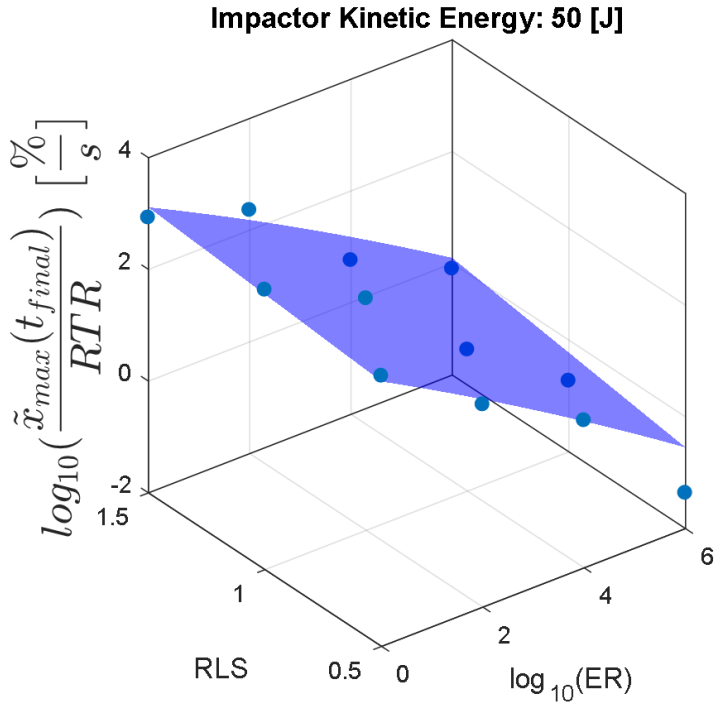
$$\log_{10}\left(\frac{\tilde{x}_{avg}(t_{final})}{RTR}\right) = 3.238 - 0.8925\log_{10}(ER) + 0.042RLS + 0.07438\log_{10}(E\,R)^{\,2} - 0.2373\log_{10}(ER)RLS$$

Figure B-3: Error-Cost Weighting Function: Relative error in average displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 100 J kinetic energy



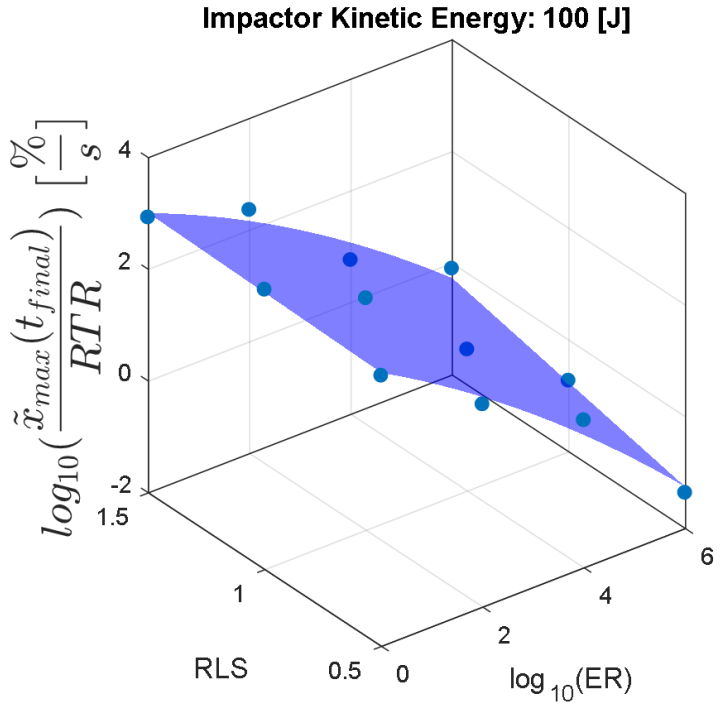
$$\log_{10}\left(\frac{\tilde{x}_{max}(t_{final})}{RTR}\right) = 2.613 - 0.6063\log_{10}(ER) + 0.4205RLS - 0.005128\log_{10}(E\,R)^{\,2} + 0.1104\log_{10}(ER)RLS$$

Figure B-4: Error-Cost Weighting Function: Relative error in maximum displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 10 J kinetic energy



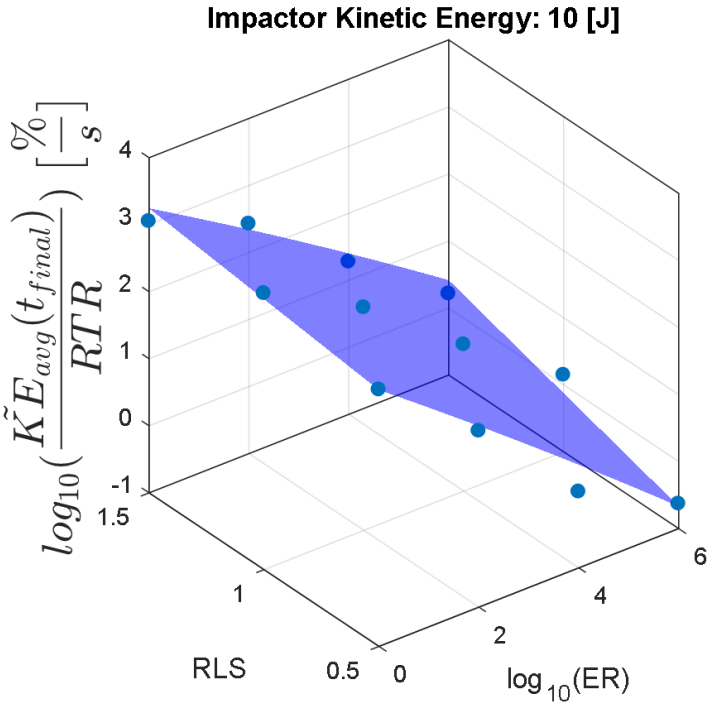
$$\log_{10}\left(\frac{\tilde{x}_{max}(t_{final})}{RTR}\right) = 2.554 - 0.5156\log_{10}(ER) + 0.3708RLS - 0.009201\log_{10}(E\,R)^{\,2} + 0.04567\log_{10}(ER)RLS$$

Figure B-5: Error-Cost Weighting Function: Relative error in maximum displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 50 J kinetic energy



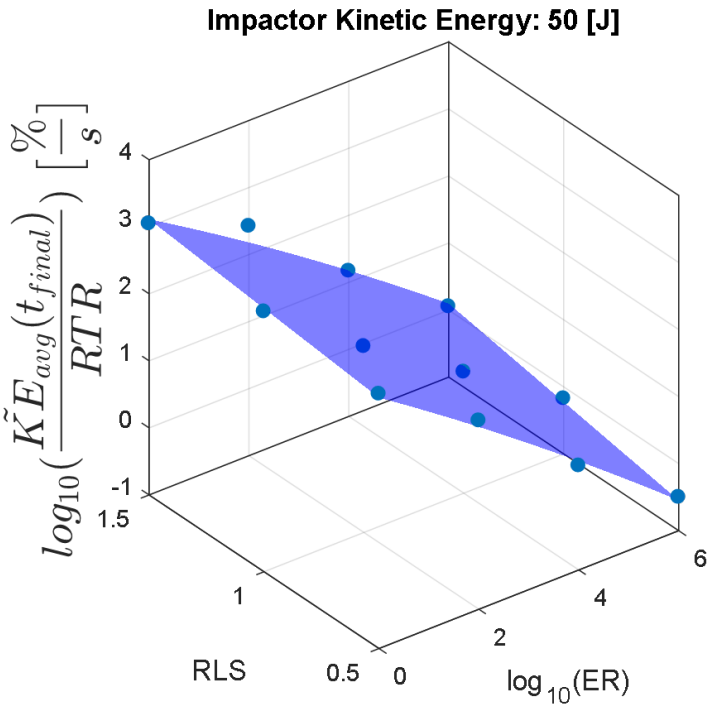
$$\log_{10}\left(\frac{\tilde{x}_{max}(t_{final})}{RTR}\right) = 2.838 - 0.58\log_{10}(ER) + 0.112RLS - 0.03081\log_{10}(E\,R)^{\,2} + 0.1464\log_{10}(ER)RLS$$

Figure B-6: Error-Cost Weighting Function: Relative error in maximum displacement at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 100 J kinetic energy



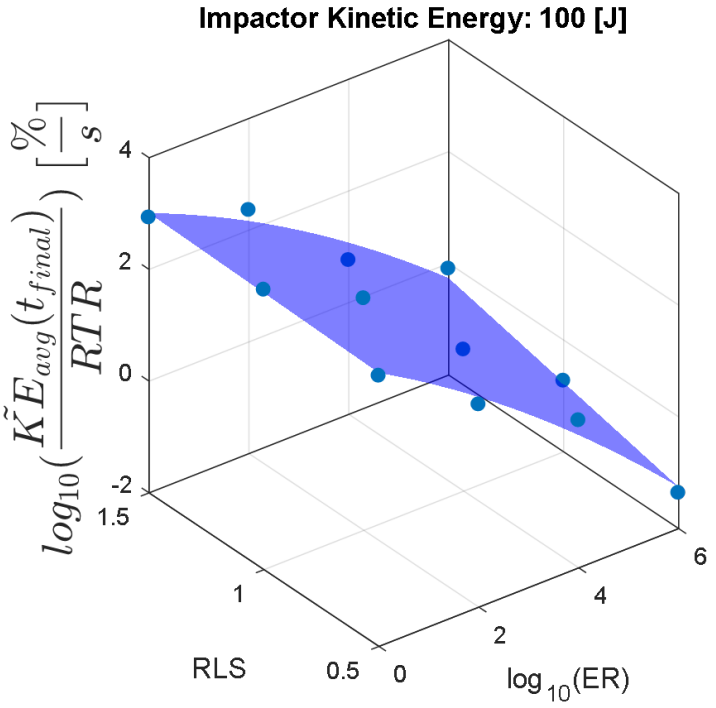
$$\log_{10}\left(\frac{\widetilde{KE}_{avg}(t_{final})}{RTR}\right) = 2.592 - 0.7261\log_{10}(ER) + 0.5894RLS + 0.02805\log_{10}(E\,R)^{\,2} + 0.04883\log_{10}(ER)RLS$$

Figure B-7: Error-Cost Weighting Function: Relative error in average artifact kinetic energy at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 10 J kinetic energy



$$\log_{10}\left(\frac{\widetilde{KE}_{avg}(t_{final})}{RTR}\right) = 2.54 - 0.4871\log_{10}(ER) + 0.5518RLS + 0.001013\log_{10}(E\,R)^{\,2} + 0.0364\log_{10}(ER)RLS$$

Figure B-8: Error-Cost Weighting Function: Relative error in average artifact kinetic energy at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 50 J kinetic energy



$$\log_{10}\left(\frac{\widetilde{KE}_{avg}(t_{final})}{RTR}\right) = 3.126 - 0.5856\log_{10}(ER) + 0.08523RLS - 0.03272\log_{10}(E\,R)^{\,2} + 0.1649\log_{10}(ER)RLS$$

Figure B-9: Error-Cost Weighting Function: Relative error in average artifact kinetic energy at simulation end per Runtime Ratio expense, with surface fit (blue) and simulation data (blue dots) and surface equation expressed below for an impactor with 100 J kinetic energy

Appendix C

Experimental Benchtop Rig Test Case Results

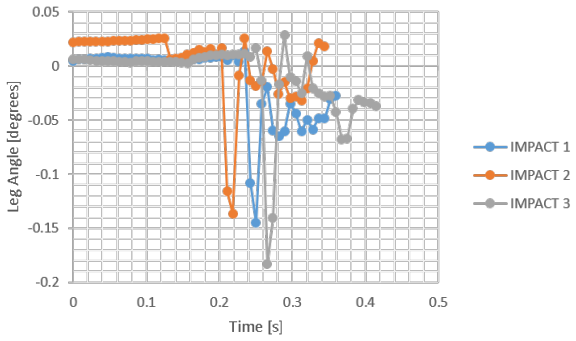


Figure C-1: Rigid Impactor, Fixed Artifact, 0.25 m/s - Artifact Angle

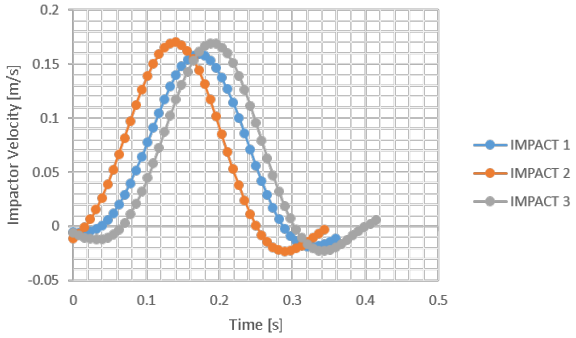


Figure C-2: Rigid Impactor, Fixed Artifact, 0.25 m/s - Impactor Velocity

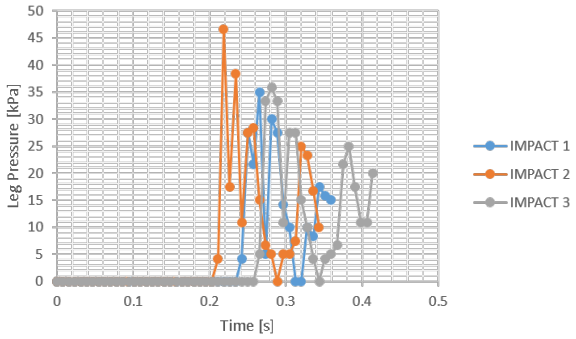


Figure C-3: Rigid Impactor, Fixed Artifact, 0.25 m/s - Impact Pressure

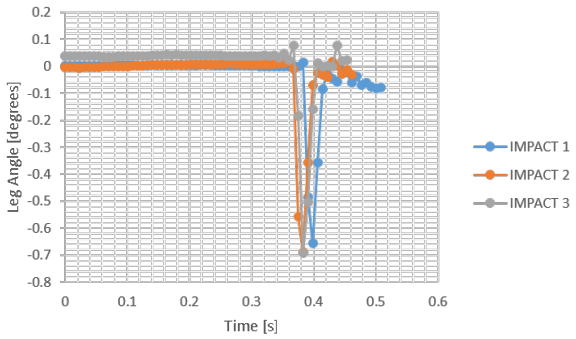


Figure C-4: Rigid Impactor, Fixed Artifact, 0.50 m/s - Artifact Angle

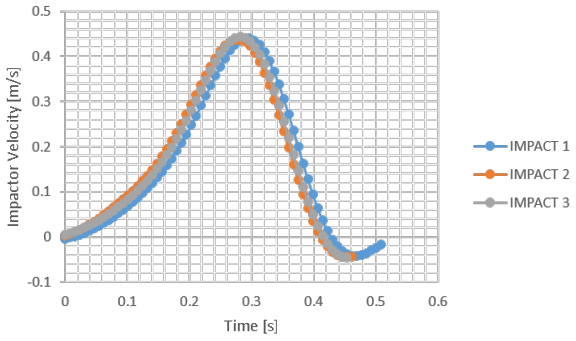


Figure C-5: Rigid Impactor, Fixed Artifact, 0.50 m/s - Impactor Velocity

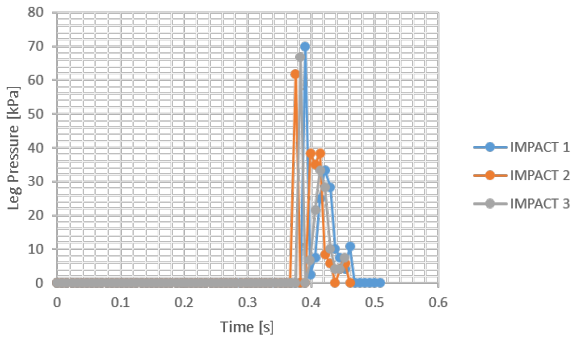


Figure C-6: Rigid Impactor, Fixed Artifact, 0.50 m/s - Impact Pressure

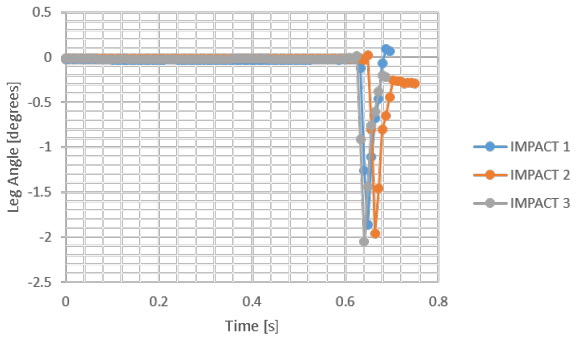


Figure C-7: Rigid Impactor, Fixed Artifact, 1.00 m/s - Artifact Angle

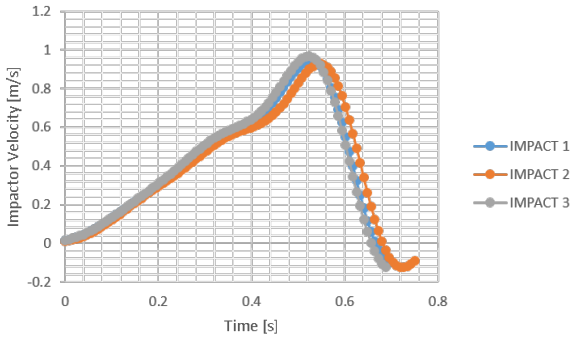


Figure C-8: Rigid Impactor, Fixed Artifact, 1.00 m/s - Impactor Velocity

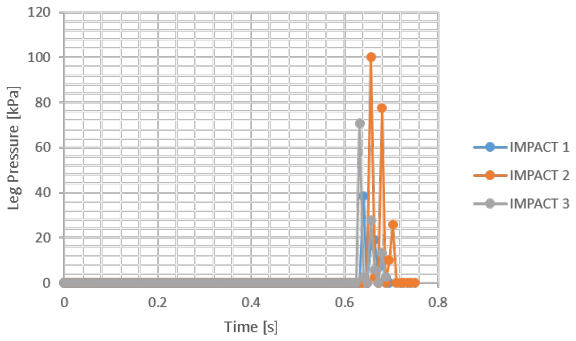


Figure C-9: Rigid Impactor, Fixed Artifact, 1.00 m/s - Impact Pressure

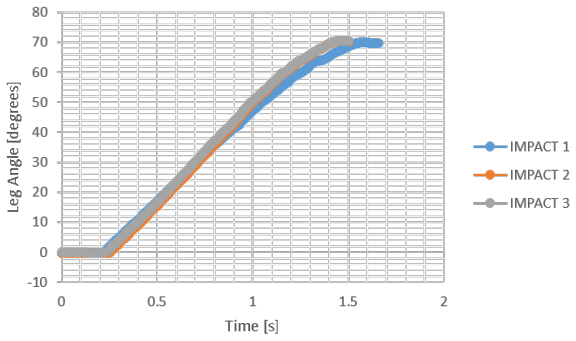


Figure C-10: Rigid Impactor, Free Artifact, 0.25 m/s - Artifact Angle

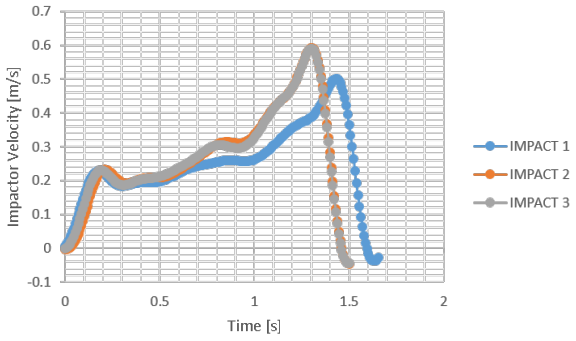


Figure C-11: Rigid Impactor, Free Artifact, 0.25 m/s - Impactor Velocity

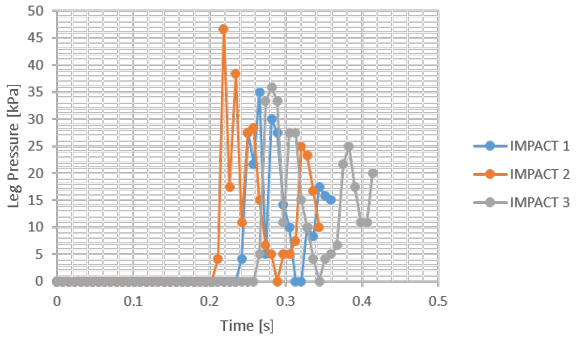


Figure C-12: Rigid Impactor, Free Artifact, 0.25 m/s - Impact Pressure

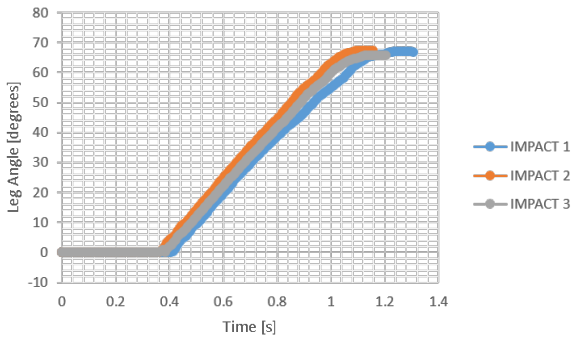


Figure C-13: Rigid Impactor, Free Artifact, 0.50 m/s - Artifact Angle

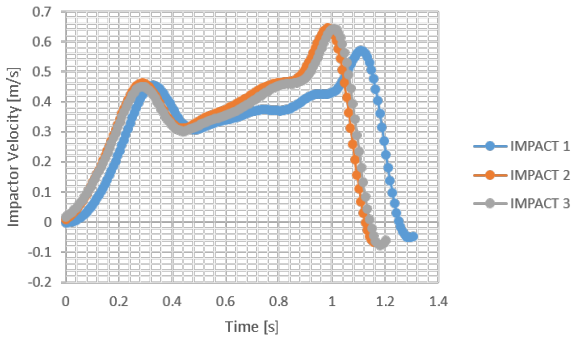


Figure C-14: Rigid Impactor, Free Artifact, 0.50 m/s - Impactor Velocity

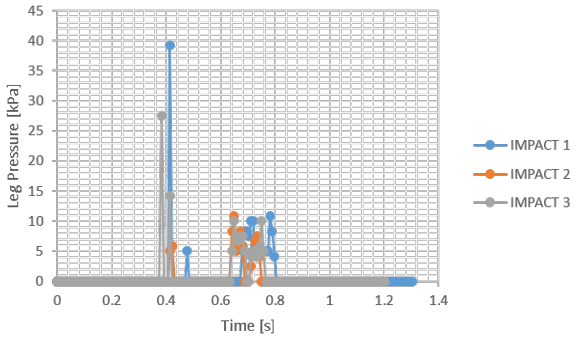


Figure C-15: Rigid Impactor, Free Artifact, 0.50 m/s - Impact Pressure

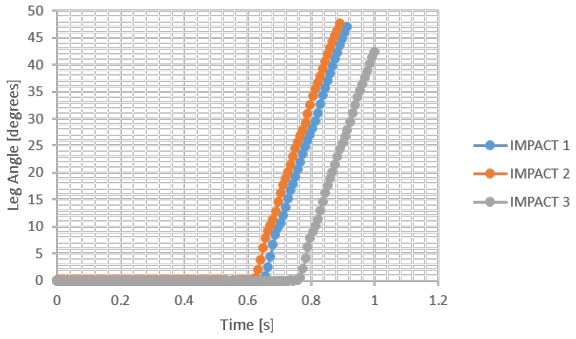


Figure C-16: Rigid Impactor, Free Artifact, 1.00 m/s - Artifact Angle

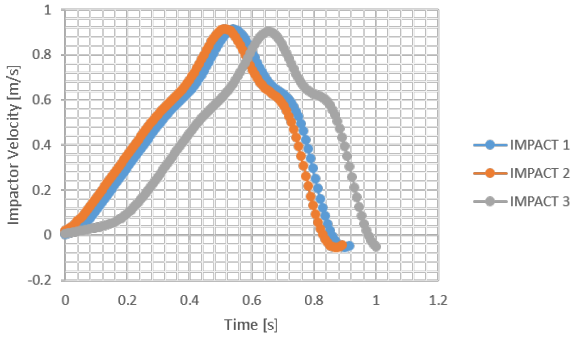


Figure C-17: Rigid Impactor, Free Artifact, 1.00 m/s - Impactor Velocity

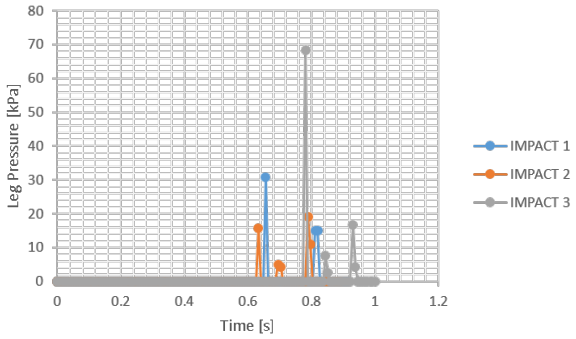


Figure C-18: Rigid Impactor, Free Artifact, 1.00 m/s - Impact Pressure

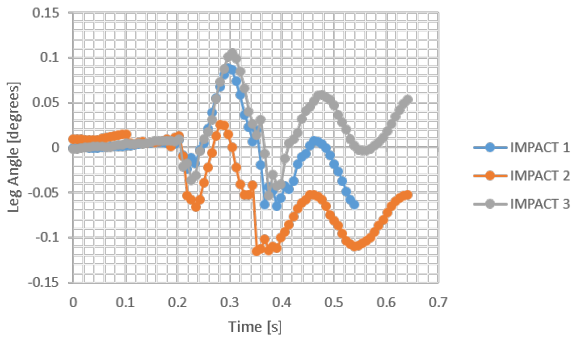


Figure C-19: Flexible Impactor, Fixed Artifact, 0.25 m/s - Artifact Angle

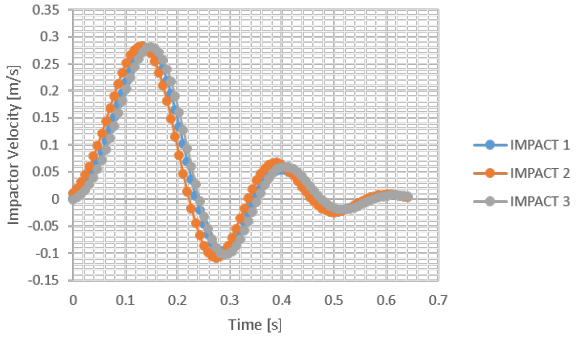


Figure C-20: Flexible Impactor, Fixed Artifact, 0.25 m/s - Impactor Velocity

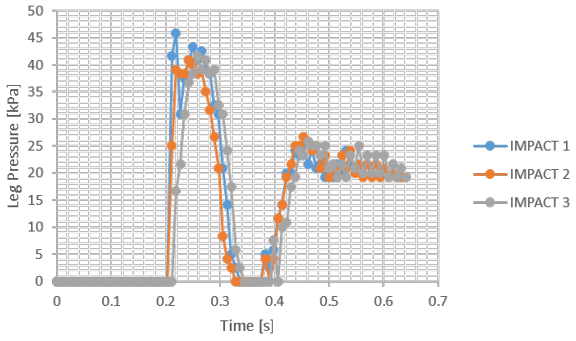


Figure C-21: Flexible Impactor, Fixed Artifact, 0.25 m/s - Impact Pressure

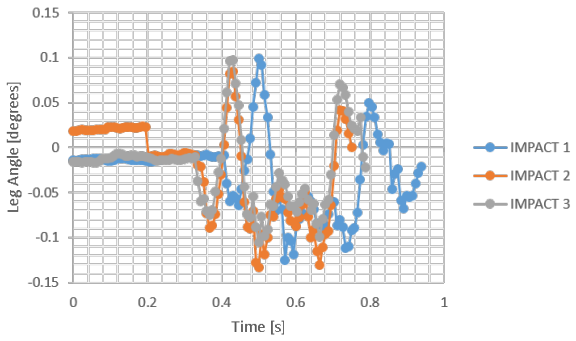


Figure C-22: Flexible Impactor, Fixed Artifact, 0.50 m/s - Artifact Angle

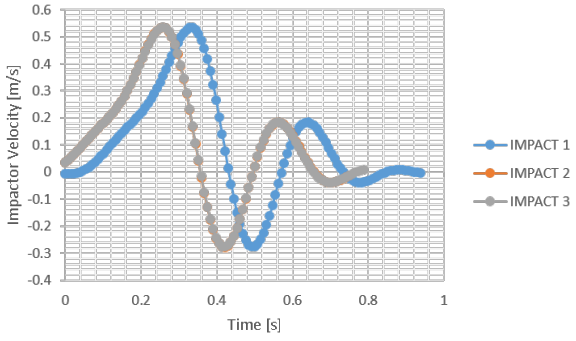


Figure C-23: Flexible Impactor, Fixed Artifact, 0.50 m/s - Impactor Velocity

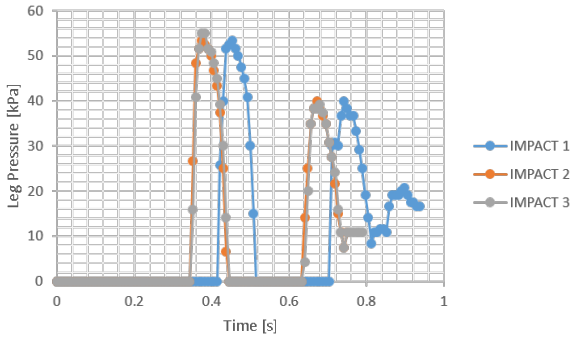


Figure C-24: Flexible Impactor, Fixed Artifact, 0.50 m/s - Impact Pressure

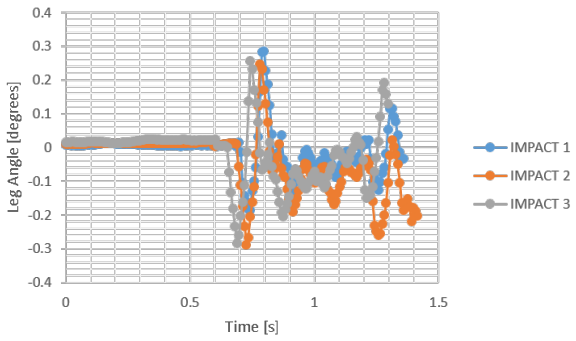


Figure C-25: Flexible Impactor, Fixed Artifact, 1.00 m/s - Artifact Angle

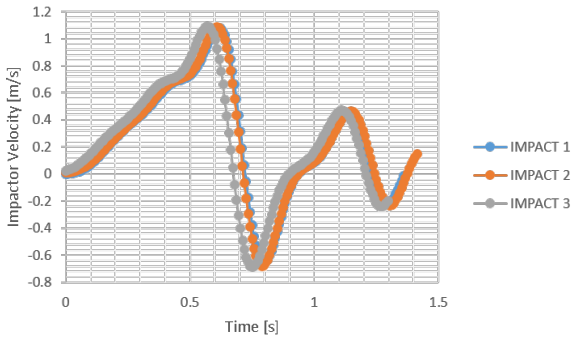


Figure C-26: Flexible Impactor, Fixed Artifact, 1.00 m/s - Impactor Velocity

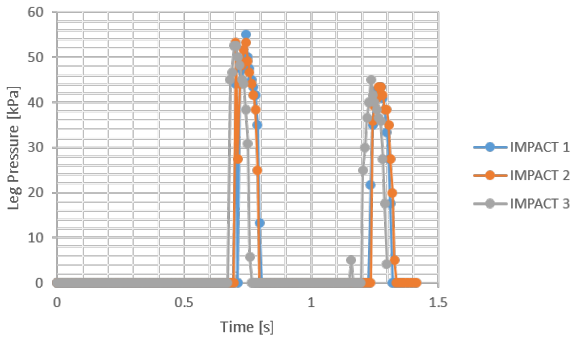


Figure C-27: Flexible Impactor, Fixed Artifact, 1.00 m/s - Impact Pressure

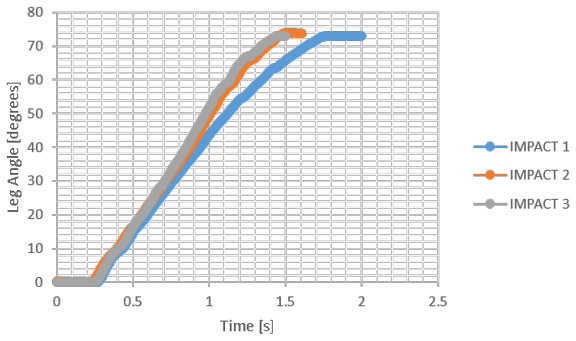


Figure C-28: Flexible Impactor, Free Artifact, 0.25 m/s - Artifact Angle

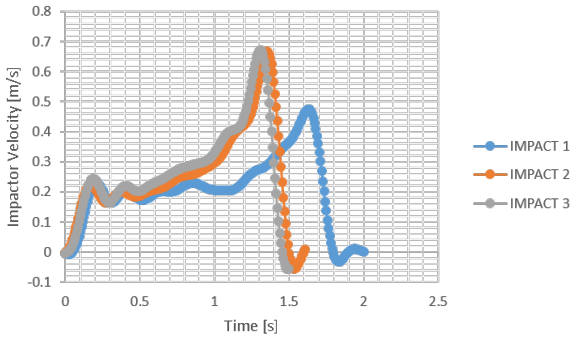


Figure C-29: Flexible Impactor, Free Artifact, 0.25 m/s - Impactor Velocity

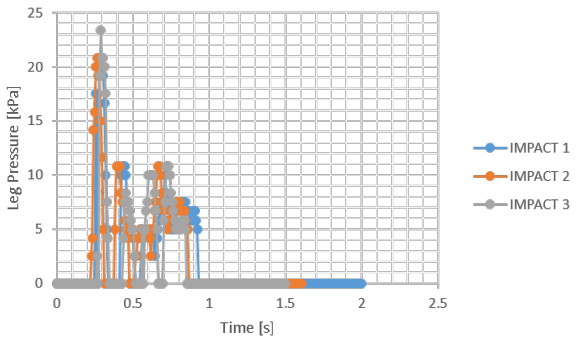


Figure C-30: Flexible Impactor, Free Artifact, 0.25 m/s - Impact Pressure

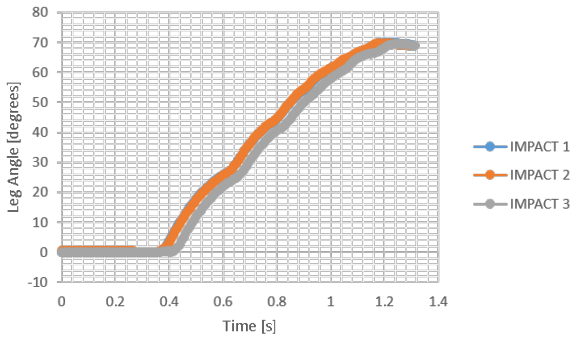


Figure C-31: Flexible Impactor, Free Artifact, 0.50 m/s - Artifact Angle

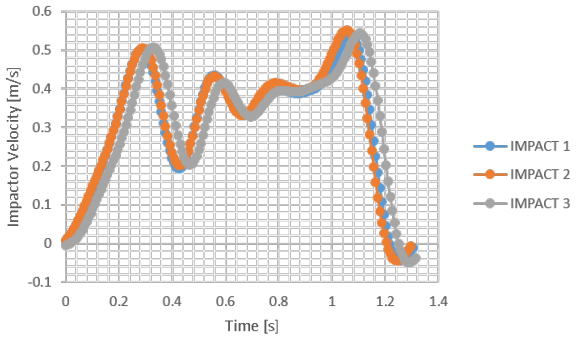


Figure C-32: Flexible Impactor, Free Artifact, 0.50 m/s - Impactor Velocity

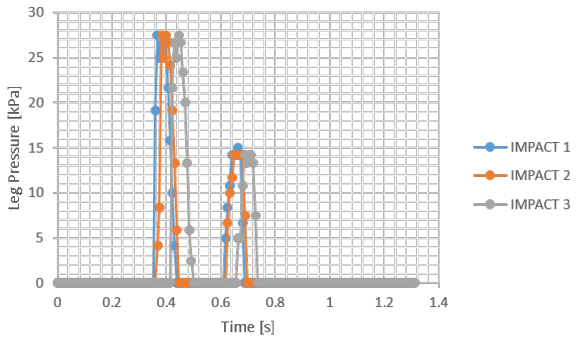


Figure C-33: Flexible Impactor, Free Artifact, 0.50 m/s - Impact Pressure

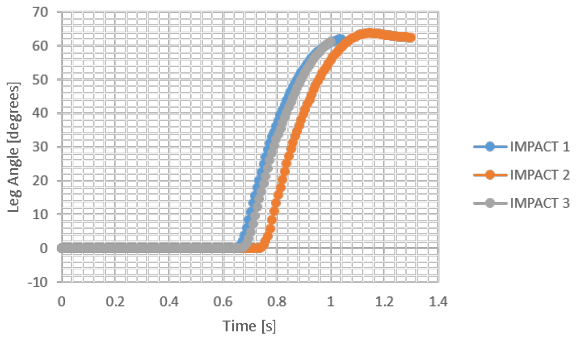


Figure C-34: Flexible Impactor, Free Artifact, 1.00 m/s - Artifact Angle

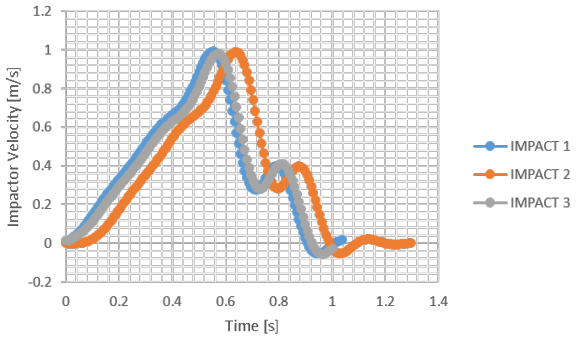


Figure C-35: Flexible Impactor, Free Artifact, 1.00 m/s - Impactor Velocity

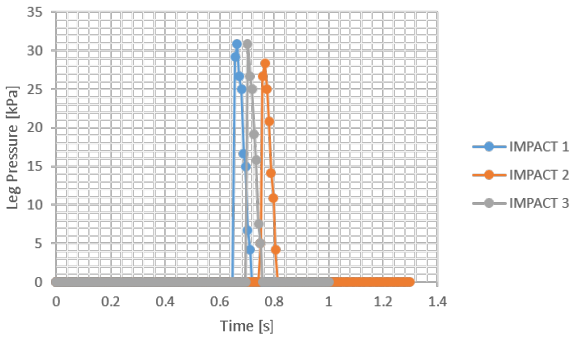


Figure C-36: Flexible Impactor, Free Artifact, 1.00 m/s - Impact Pressure

Appendix D

Universal I-DEAS to SUMMIT Connectivity Toolset

To enable a single-input driven human-suit modeling architecture, it was desired to develop an automatic utility for providing computational geometry, discretized mesh data, material properties, boundary conditions, and surfaces on which to enforce contact to SUMMIT. At the time this project began, the means by which to import data from a master CAD model was not well supported. While SUMMIT supports automatically importing mesh data from a variety of pre-processors, it did not have a means to translate the concept of computational surfaces, edges, or vertices present at the CAD level, as the FEM mesh had no insight into the CAD it was generated from. Additionally, it was convenient to use a CAD software that supported meshing, material property definition, and boundary condition application, as this aligned with the primary goal of driving all inputs to the human-suit model through a single source.

As a result, a review of commercially available CAD software and preprocessors was performed, looking to identify a software which provided the following:

- Material property definition and assignment to CAD and FEM mesh
- Ability to generate simplicial FEM mesh from CAD
- Ability to prescribe boundary conditions to simplicial FEM mesh
- Ability to select subset of surfaces on which to enforce contact
- Ability to export FEM mesh, boundary conditions, assigned material properties, and desired contact surfaces to an ASCII-readable file
- Exported ASCII-readable file format must be a standardized format (i.e., have documented format conventions)

The student editions of ABAQUS, ANSYS Workbench, Hypermesh, and SolidWorks were interrogated for capability to satisfy all of the above. While capabilities for meshing in ABAQUS, ANSYS, and Hypermesh vastly exceed those of SolidWorks, the preprocessing capability and ability to export to an ASCII-readable file format resulted in the selection of SolidWorks as the CAD for this model. The export file format was identifed as the Universal I-DEAS (UNV) file format, as it contains the

ability to capture FEM mesh data, material properties, boundary conditions and constrains, along with thermal state data as well, which was leveraged for identification of contact surfaces for this work.

With the UNV file format selected as the exported dataset from SolidWorks, it was necessary to construct a toolset that could automatically parse the output UNV file and consistently generate input data for SUMMIT with consistency and speed. The developed toolset was written in Python 3.0 and stored in the "meshing-utilities" bzr repository maintained by Raul Radovitzsky's research group (RRGroup). The overall utility is the "UNVToSummit2.0.py" driver, and is invoked in the following fashion:

UNVToSummit2.0.py <UNV File Name>.UNV <output file name>

The UNVToSummit driver will parse the provided UNV file and identify if FEM mesh data, material property data, boundary condition data, or contact surface data is present, and prompt the user if additional file sets beyond the SUMMIT mesh file should be generated. The primary file types generated by the UNVToSummit2.0 driver are as illustrated in Figure D-1.

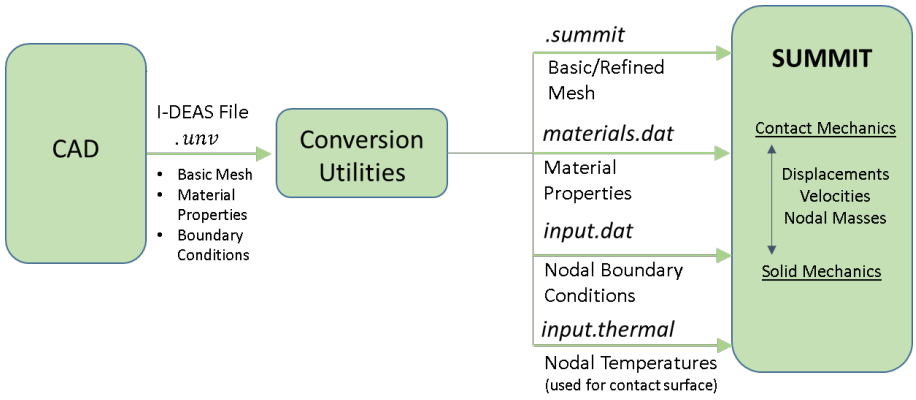


Figure D-1: UNVToSummit Driver High-Level I/O and Output File Intent for coupling with SUMMIT

It should be stressed that to use the output of this driver, two primary conditions must be met:

1) The FEM mesh must not be modified in the SUMMIT simulation If the mesh is refined or coarsened, mapping between boundary conditions and the mesh will be broken and any imported boundary conditions or material properties will be erroneous

2) The FEM mesh must be simplicial (first order) tetrahedrons, which can be enforced by selecting "Draft Quality Mesh" in the SolidWorks mesh options This is currently required as the DCR methods only support triangular mesh entities at the current time. Attempting to bring in quadrilateral surface elements will cause the DCR code to fail.

3) The material properties file (.dat) must be updated by the user to the constitutive model that matches their needs after conversion from UNV. This is because the current verions of the UNVToSummit driver does not know the intended constitutive material model to be used in SUMMIT. At present, it assumes a Neohookean model and provides that information, but if this material model is not sufficient, it should be changed accordingly. However, the material mapping to the FEM mesh remains valid.

Additional improvements to the driver could be made if computational geometry data can be reconstructed in SUMMIT. Application of constraints is not currently supported as driven by SolidWorks, as the UNV constraint data is local to the coordinate system of the computational geometry; information not known to a FEM mesh. As such, if the toolset is improved to reconstruct computational geometry to interpret the UNV constraint data, this method could be improved. Finally, it should be stressed that this driver was developed based on the restrictions of the student version of SolidWorks 2017. Commercial licenses may have additional export data available, such as support for more robust boundary condition types than nodal-only. If this is the case, the toolset can be updated to account for the additional data available, by following the UNV file format information included in the UNVToSummit2.0.py driver file.

Appendix E

Coupled Contact-Mechanics (CCM) Model Development Gates

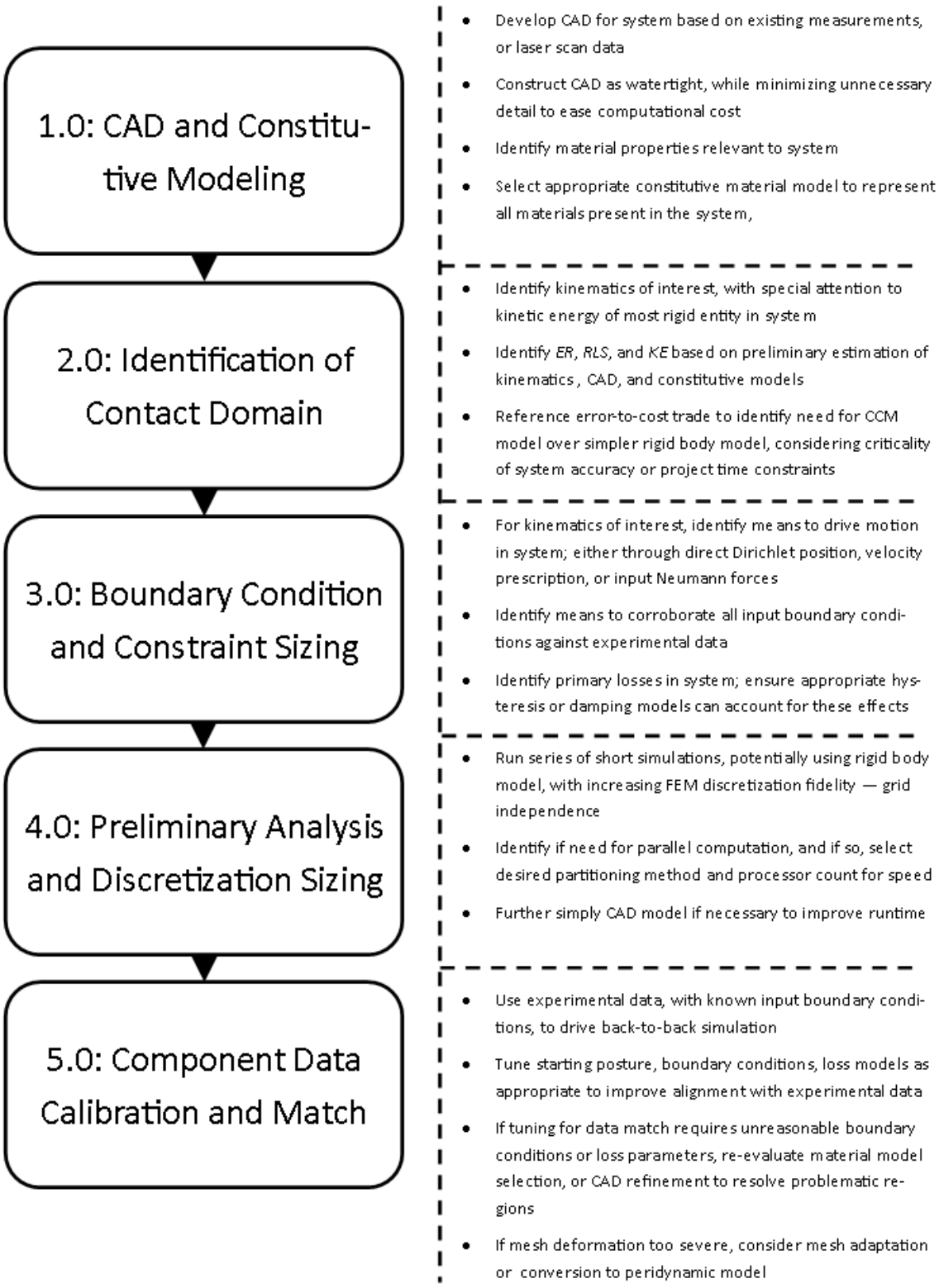


Figure E-1: Recommended stages and steps in developing a simulation for and calibrating the CCM model to gain alignment with experimental human-suit testing data for broader human-suit modeling

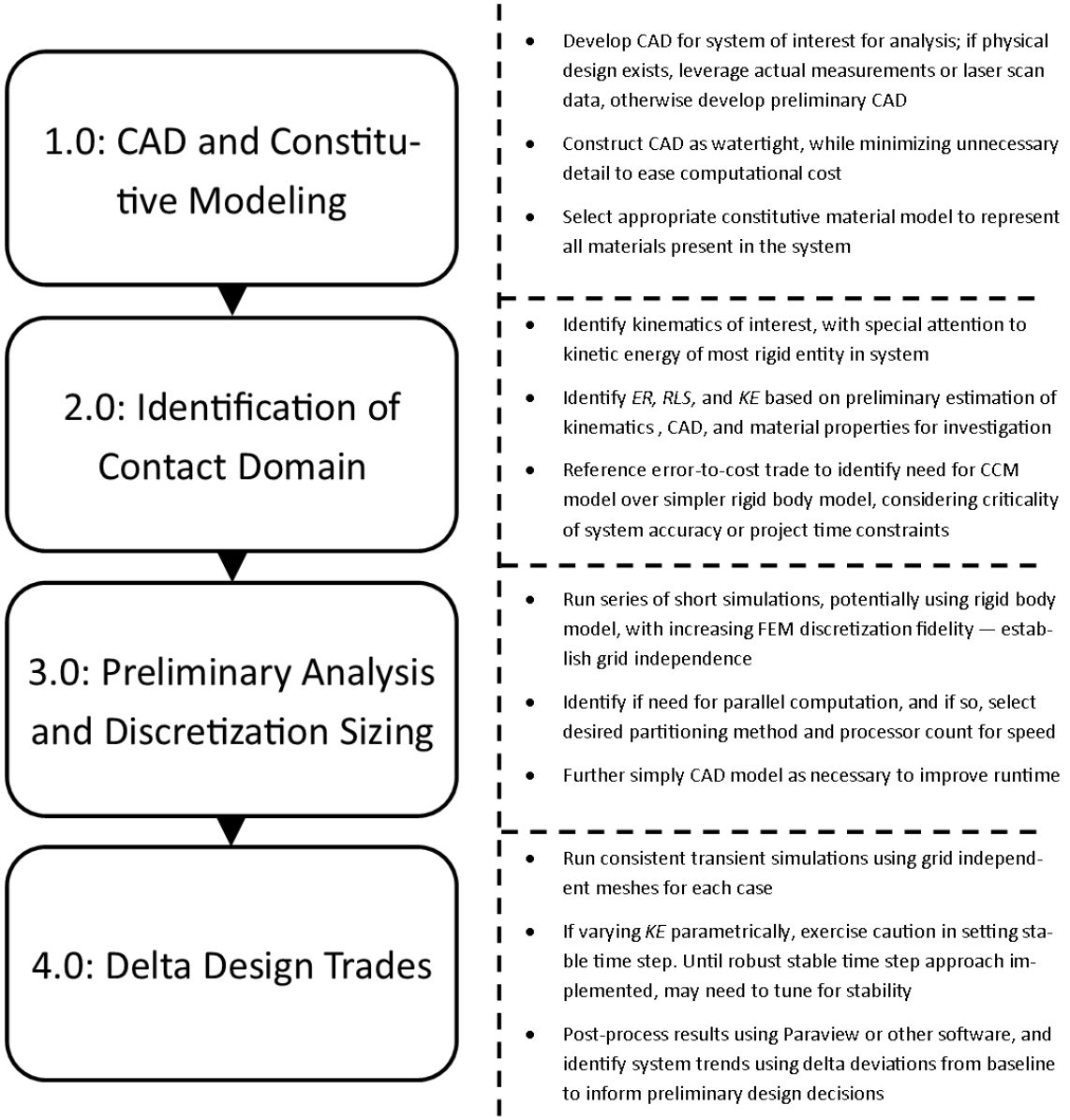


Figure E-2: Recommended stages and steps in performing a preliminary design-level trade study in suit design using the CCM model

Bibliography

[1] "Mark iii space suit." `https://lsda.jsc.nasa.gov/PhotoGallery/detail_result/2954`. Accessed: 2018-04-24.

[2] S. Jacobs and D. Tufts, "Follow-on development of the demonstrator suit for post-shuttle operations," *41st International Conference on Environmental Systems*, 2011.

[3] A. Seth, M. Sherman, J. Reinbolt, and S. Delp, "Opensim: A musculoskeletal modeling and simulation framework for in silico investagations and exchange," *Procedia IUTAM*, vol. 2, pp. 212–232, 2011.

[4] E. Todorov, "Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in mujoco," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[5] R. Wang, Y. Liu, X. Luo, Y. Li, and S. Ji, "A finite-element mechanical contact model based on mindlin-reissner shell theory for a three-dimensional human body and garment," *Journal of Computational and Applied Mathematics*, vol. 236, pp. 867–877, 2011.

[6] P. Volino and N. Magnenat-Thalmann, "Accurate garment prototyping and simulation," *Computer-Aided Design and Applications*, vol. 2, pp. 645–654, 2005.

[7] M. Gernhardt, J. Jones, R. Scheuring, A. Abercromby, J. Tuxhorn, and J. Norcross, "Risk of compromised eva performance and crew health due to inadequate eva suit systems," in *Human Health and Performance Risks of Space Exploration Missions* (J. Mcphee and J. Charles, eds.), ch. 14, pp. 333–358, US National Aeronautics and Space Administration, 2010.

[8] R. Scheuring, J. Jones, J. Polk, D. Gillis, J. Schmid, J. Duncan, and J. Davis, "The apollo medical operations project: recommendations to improve crew health and performance for future exploration missions and lunar surface operations," Tech. Rep. 214755, NASA, Aug. 2007.

[9] S. Strauss, "Extravehicular mobility unit training suit symptom study report," Tech. Rep. 212075, NASA, June 2004.

[10] M. Wehner, B. Quinlivan, P. Aubin, E. Martinez-Villalpando, M. Baumann, L. Stirling, K. Holt, R. Wood, and C. Walsh, "A lightweight soft exosuit for gait assistance," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[11] D. Newman and M. Barratt, "Life support and performance issues for extravehicular activity (eva)," in *Fundamentals of Space Life Sciences* (S. Churchill, ed.), ch. 22, Krieger Publishing Company, 1997.

[12] A. Ross and R. Rhodes, "Z-2 prototype space suit development," in *44th International Conference on Environmental Systems*, 2014.

[13] R. Johnston, L. Dietlein, and C. Berry, "Biomedical results of apollo," Tech. Rep. 368, NASA, 1975.

[14] C. E. Carr, *The Bioenergetics of Walking and Running in Space Suits.* PhD dissertation, Massachusetts Institute of Technology, Harvard-MIT Division of Health Sciences and Technology, May 2005.

[15] D. Morgan, R. Wilmington, A. Pandya, J. Maida, and K. Demel, "Comparison of extravehicular mobility unit (emu) suited and unsuited isolated joint strength measurements," Tech. Rep. 3613, NASA, June 1996.

[16] J. Norcross, K. Clowers, T. Clark, L. Harvill, R. Morency, L. Stroud, L. Desantis, J. Vos, and M. Gernhardt, "Metabolic costs of biomechanics of level ambulation in a planetary suit," Tech. Rep. 216115, NASA, Feb. 2010.

[17] C. Cullinane, R. Rhodes, and L. Stirling, "Mobility and agility during locomotion in the mark iii space suit," *Aerospace Medicine and Human Performance*, 2017.

[18] P. B. Schmidt, *An Investigation of Space Suit Mobility with Applications to EVA Operations.* PhD dissertation, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Aug. 2001.

[19] B. Holschuh, J. Waldie, J. Hoffman, and D. Newman, "Characterization of structural, volume, and pressure components to space suit joint rigidity," in *Proceedings of the International Conference on Environmental Systems*, 2009.

[20] D. Parry, L. Curry, D. Hanson, and G. Towle, "A study of techniques and equipment for the evaluation of extravehicular protective garments," Tech. Rep. 66-4, Hamilton Standard Division of United Aircraft Corporation, 1966.

[21] M. Di Capua, *Inertial Motion Capture System for Biomechanical Analysis in Pressure Suits.* PhD dissertation, University of Maryland, Department of Aerospace Engineering, 2012.

[22] A. Anderson, *Understanding Human-Space Suit Interaction to Prevent Injury During Extravehicular Activity.* PhD dissertation, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2014.

[23] A. Anderson and D. Newman, "Pressure sensing for in-suit measurement of space suited biomechanics," *Acta Astronautica*, vol. 115, pp. 218–225, 2015.

[24] "Solidworks online help manual." `http://help.solidworks.com/2017/english/solidworks/cworks/c_meshing.htm`. Accessed: 2018-04-09.

[25] S. Delp, F. Anderson, A. Arnold, P. Loan, A. Habib, C. John, E. Guendelman, and D. Thelen, "Opensim: Open-source software to create and analyze dynamic simulations of movement," in *IEEE Transactions on Biomedical Engineering*, 2007.

[26] D. Krüger and S. Wartzack, "A contact model to simulate human-artifact interaction based on force optimization: implementation and application to the analysis of a training machine," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, pp. 1589–1598, 2017.

[27] P. Volino and N. Magnenat-Thalmann, "Developing simulation techniques for an interactive clothing system," in *Proceedings of the 1997 International Conference on Virtual Systems and MultiMedia*, pp. 109–118, 1997.

[28] N. Magnenat-Thalmann and P. Volino, "From early draping to haute couture models: 20 years of research," *Visual Computing*, vol. 21, pp. 506–519, 2005.

[29] Y. Meng, P. Mok, and X. Jin, "Interactive virtual try-on clothing design systems," *Computer-Aided Design*, vol. 42, pp. 310–321, 2010.

[30] J. Collier, B. Collier, G. O'Toole, and S. Sargand, "Drape prediction by means of finite-element analysis," *Journal of the Textile Institute*, vol. 82, pp. 96–107, 1991.

[31] J. Eischen, S. Deng, and T. Clapp, "Finite-element modeling and control of flexible fabric parts," *IEEE Computer Graphics and Applications*, vol. 16, pp. 71–80, 1996.

[32] B. de Kruif, E. Schmidhauser, K. Stadler, and L. O'Sullivan, "Simulation architecture for modelling interaction between user and elbow-articulated exoskeleton," *Journal of Bionic Engineering*, vol. 14, pp. 706–715, 2017.

[33] J.-R. Cho, D.-Y. Lee, and Y.-J. Ahn, "Finite element investigation of the biomechanical responses of human foot to the heel height and a rigid hemisphere cleat," *Journal of Mechanical Science and Technology*, vol. 30, pp. 4269–4274, 2016.

[34] "Ansys mechanical apdl contact technology guide." `https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v190/ans_ctec/ctectoc.html`. Accessed: 2018-04-07.

[35] "3.5.3 parallel execution in abaqus/explicit - domain-level parallelization." `http://abaqus.software.polimi.it/v6.14/books/usb/default.htm?startat=pt01ch03s05aus34.html`. Accessed: 2018-04-03.

[36] "Febio 2.5 user manual." http://help.mrl.sci.utah.edu/help/index.jsp?topic=%2Forg.febio.help.febio_um%2Fhtml%2F8.4.html. Accessed: 2018-04-07.

[37] T. Singh and S. Ambike, "A soft-contact model for computing safety margins in human prehension," *Human Movement Science*, vol. 55, pp. 307–314, 2017.

[38] K. Danelson, J. Bolte, and J. Stitzel, "Assessing astronaut injury potential from suit connectors using a human body finite element model," *Aviation, Space, and Environmental Medicine*, vol. 82, pp. 79–86, 2011.

[39] I. Eskinazi and B. Fregly, "An open-source toolbox for surrogate modeling of joint contact mechanics," *IEEE Transactions on Biomedical Engineering*, vol. 63, pp. 269–277, 2016.

[40] M. Marra, M. Andersen, M. Damsgaard, B. Koopman, D. Janssen, and N. Verdonschot, "Evaluation of a surrogate contact model in force-dependent kinematic simulations of total knee replacement," *Journal of Biomechanical Engineering*, vol. 139, 2017.

[41] Y. Bei and B. Fregly, "Multibody dynamic simulation of knee contact mechanics," *Medical Engineering and Physics*, vol. 26, pp. 777–789, 2004.

[42] "Opensim plugin to estimate contact forces using static optimization." https://simtk.org/projects/statopt-contact. Accessed: 2018-04-25.

[43] J. Lee, M. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. Srinivasa, M. Stilman, and C. Liu, "Dart: Dynamic animation and robotics toolkit," *The Journal of Open Source Software*, vol. 3, p. 500, 2018.

[44] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx," in *International Conference on Robotics and Automation*, 2015.

[45] K. Namima, Z. Wang, and S. Hirai, "Simulation of soft fingertip deformation under contact and rolling constraints using fem and csm," in *Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics*, 2009.

[46] J.-P. Gourret and N. Magnenat-Thalmann, "Simulation of object and human skin deformations in a grasping task," *Computer Graphics*, vol. 23, pp. 21–30, 1989.

[47] M. Tupek, J. Rimoli, and R. Radovitzky, "An approach for incorporating classical continuum damage models in state-based peridynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 263, pp. 20–26, 2013.

[48] Y.-J. Liu, D.-L. Zhang, and M. M.-F. Yuen, "A survey on cad methods in 3d garment design," *Computers in Industry*, vol. 61, pp. 576–593, 2010.

[49] Q. Li and K.-M. Lee, "An adaptive meshless method for analyzing large mechanical deformation and contacts," *Journal of Applied Mechanics*, vol. 75, 2008.

[50] "Opensim: How inverse dynamics works." `https://simtk-confluence.stanford.edu/display/OpenSim/How+Inverse+Dynamics+Works`. Accessed: 2018-04-26.

[51] "Opensim: How static optimization works." `https://simtk-confluence.stanford.edu/display/OpenSim/How+Static+Optimization+Works`. Accessed: 2018-04-26.

[52] E. Kandel, J. Schwartz, T. Jessell, S. Siegelbaum, and A. Hudspeth, *Principles of Neural Science*, ch. 34, pp. 776–788. McGraw-Hill, fifth ed., 2013.

[53] A. Seagraves, "A scalable computational approach for modeling dynamic fracture of brittle solids in three dimensions," master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Feb. 2010.

[54] B. Fagan, "Large-scale simulation of the mechanical response of porcine cranial bone," master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Feb. 2017.

[55] T. Fronk, "Accelerated computational modeling of ballistic helmet test protocols," master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Feb. 2017.

[56] G. Karypis and V. Kumar, "A parallel algorithm for multilevel graph partitioning and sparse matrix ordering," *Journal of Parallel and Distributed Computing*, vol. 48, pp. 71–95, 1998.

[57] S. Timoshenko and J. Goodier, *Theory of Elasticity*. McGraw-Hill, 1970.

[58] L. Noels and R. Radovitzky, "A general discontinuous galerkin method for finite hyperelasticity. formulation and numerical applications," *International Journal for Numerical Methods in Engineering*, 2006.

[59] D. Benson and J. Hallquist, "A single surface contact algorithm for the post-buckling analysis of shell structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 78, pp. 141–163, 1990.

[60] T. Laursen, *Computational Contact and Impact Mechanics*. Springer, 2002.

[61] L. Campos, J. Oden, and N. Kikuchi, "A numerical analysis of a class of contact problems with friction in elastostatics," *Computer Methods in Applied Mechanics and Engineering*, vol. 34, pp. 821–845, 1982.

[62] "Theory and modeling guide volume i: Adina." `http://www.adina.com/adinadownloads/docs/tmg-a_89.pdf`. Accessed: 2018-04-07.

[63] R. Weyler, J. Oliver, T. Sain, and J. Cante, "On the contact domain method: A comparison of penalty and lagrange multiplier implementations," *Computational Methods in Applied Mechanics and Engineering*, vol. 205-208, pp. 68–82, 2012.

[64] M. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, pp. 303–320, 1969.

[65] F. Cirak and M. West, "Decomposition contact response (DCR) for explicit finite element dynamics," *International Journal for Numerical Methods in Engineering*, vol. 64, pp. 1078–1110, 2005.

[66] N. Newmark, "A method of computation for structrual dynamics," *Journal of Engineering Mechanics*, 1959.

[67] C. Kane, J. Marsden, M. Ortiz, and M. West, "Variational integrators and the newmark algorithm for conservative and dissipative mechanical systems," *International Journal for Numerical Methods in Engineering*, vol. 49, pp. 1295–1325, 2000.

[68] X. Liu, "On the stability of a newmark's scheme-based predictor-corrector algorithm," *Computers and Structures*, vol. 53, pp. 27–33, 1994.

[69] D. Mangoni, A. Tasora, and R. Garziera, "A primal-dual predictor-corrector interior point method for non-smooth contact dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 330, pp. 351–367, 2018.

[70] F. Cirak, M. West, S. Mauch, and R. Radovitzky, "Parallel finite element computation of contact-impact problems with large deformations," *Computational Fluid and Solid Mechanics*, 2003.

[71] J. Weidendorfer, M. Kowarschik, and C. Trinitis, "A tool suite for simulation based analysis of memory access behavior," in *Proceedings of the 4th International Conference on Computational Science (ICCS 2004), Krakow, Poland*, 2004.

[72] J. Seward and N. Nethercote, "Using valgrind to detect undefined value errors with bit-precision," in *Proceedings of the USENIX'05 Annual Technical Conference, Anaheim, California, USA*, 2005.

[73] N. Nethercote and J. Seward, "Valgrind: A framework for heavyweight dynamic binary instrumentation," in *Proceedings of the ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation (PLDI 2007), San Diego, California, USA*, 2007.

[74] "3.10 - options that control optimization." `https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html`. Accessed: 2018-05-03.

[75] K. Danielson and R. Namburu, "Nonlinear dynamic finite element analysis on parallel computers using fortran 90 and mpi," *Advances in Engineering Software*, vol. 29, pp. 289–299, 1998.

[76] M. Berger and S. Bokhari, "A partitioning strategy for nonuniform problems on multiprocessors," *IEEE Trans. Computers*, vol. 36, pp. 570–580, 1987.

[77] K. Devine, E. Boman, R. Heaphy, B. Hendrickson, and C. Vaughan, "Zoltan data management management services for parallel dynamic applications," *Computing in Science and Engineering*, vol. 4, no. 2, pp. 90–97, 2002.

[78] K. Devine, E. Boman, R. Heaphy, B. Hendrickson, J. Teresco, J. Faik, J. Flaherty, and L. Gervasio, "New challenges in dynamic load balancing," *Applied Numerical Mathematics*, vol. 52, pp. 133–152, 2005.

[79] J. Hursey, E. Mallove, J. M. Squyres, and A. Lumsdaine, "An extensible framework for distributed testing of mpi implementations," in *Proceedings, Euro PVM/MPI*, (Paris, France), October 2007.

[80] X. Liang and S. Boppart, "Biomechanical properties of in vivo human skin from dynamic optical coherence elastopgraphy," *IEEE Trans Biomed Eng.*, vol. 57, pp. 953–959, 2010.

[81] D. Hartman, M. Greenwood, and D. Miller, *High Strength Glass Fibers.* agy, 2014. Accessed via `https://www.agy.com/wp-content/uploads/2014/03/High_Strength_Glass_Fibers-Technical.pdf`.

[82] "Vicon nexus." `https://www.vicon.com/file/vicon/nexus-2-7web-24392.pdf`. Accessed: 2018-05-07.

[83] "Pliance sensors." `http://www.novelusa.com/index.php?fuseaction=systems.plianceSensors`. Accessed: 2018-04-13.

[84] "Apdm wearable sensors." `https://www.apdm.com/wearable-sensors/`. Accessed: 2018-04-16.

[85] J. Ahrens, B. Geveci, and C. Law, *ParaView: An End-User Tool for Large Data Visualization.* Elsevier, 2005.

[86] R. Radovitzky and M. Ortiz, "Lagrangian finite element analaysis of newtonian fluid flows," *International Journal for Numerical Methods in Engineering*, vol. 43, pp. 607–619, 1998.

[87] S.A. Silling, "Reformulation of elasticity theory for discontinuities and long-range forces," *Journal of the Mechanics and Physics of Solids*, vol. 48, pp. 175–209, 2000.