Design and fabrication of 3D-printed hard-soft interfaces made from PLA and hydrogel

Carlos Pitta Kruize

Design and fabrication of 3D-printed hard-soft interfaces made from PLA and hydrogel

by

Carlos Pitta Kruize

in partial fulfilment of the requirements for the degree of

Master of Science
in Biomedical Engineering

at the Delft University of Technology,
to be defended publicly on Tuesday September 28, 2021 at 09:00 AM.

Supervisors:	Prof. dr. A. Zadpoor
	Dr. M. J. Mirzaali
Daily supervisors:	N. E. Putra
	M. Cruz Saldivar
Thesis committee:	Dr. ir. G. Smit
	Dr. M. J. Mirzaali
	Prof. dr. A. Zadpoor

This thesis is confidential and cannot be made public until September 28, 2023.

Acknowledgements

My name is Carlos Pitta Kruize and I am a master student Biomedical Engineering at the TU Delft, specializing in Musculoskeletal Biomechanics. My background is in Human Movement Sciences, where I performed a minor on the structure, function and control of the musculoskeletal system. Before that I spent six years in military service at the Royal Netherlands Marine Corps (*het Korps Mariniers*).

I studied three years at the TU Delft. Two years were spent on my internship and research project. This coincided with the COVID-19 pandemic, which made it even more challenging. During this time, I obtained new perspectives and acquired a great deal of knowledge. Now this has come to an end, I am excited to apply my acquired skills and knowledge to new challenges.

I would like to express my profound gratitude to all the people involved in my graduation period. I would like to thank my supervisor Mohammad Mirzaali Mazandarani for all his kindness and guidance along the way. I would like to thank Amir Zadpoor for inspiring me to choose this field of research and for the opportunity to work in his research group. I would like to thank my daily supervisors Niko Putra and Mauricio Cruz Saldivar for their continued support. I would like to thank the members of my thesis committee for taking the time to attend my graduation. And finally, I would like to thank my friends and family for their continued support during my study period, and to Manon, Hans, Willem and Minda in special.

Carlos Pitta Kruize
Amsterdam, September 2021

Abstract

The mechanical properties of biological hard-soft interfaces change gradually over a small surface area to prevent the formation of stress concentrations through variations in the interface's chemical composition, microstructure and geometry. The bone-tendon/ligament and bone-cartilage interface are the most prominent hard-soft interfaces within the human body and need restoration once injuries occur. It is difficult to repair these interfaces and simultaneously maintain the initial quality of the native tissues. This study used hybrid 3D-printing techniques, *i.e.*, fused deposition modeling (FDM) and an extrusion-based technique, to develop a proof of concept for the fabrication of hard-soft interface structures. The first part of the study concentrates on the design of geometrical interlocking structures. A parametric study with multiple simulations was performed for two specific geometries, an anti-trapezoidal and a double hook design. The double hook design demonstrated the highest stiffness under tensile stress conditions. The second part concentrates on manufacturing hard-soft interface structures, for which 2D and 3D models were fabricated. The 2D models were used to explore different geometrical interlocking designs and validate the computational models. The 3D models were used to create a proof-of-concept for a hard-soft interface made from PLA (FDM technique) and alginate (extrusion-based technique). The 3D-printing techniques were combined by extruding hydrogel into the interlocking system of the PLA part and printing a soft alginate scaffold on top of the interlocking structure. In conclusion, this study suggests several practical solutions to improve interfacial designs and to manufacture hard-soft interface structures. Combining 3D-printing techniques opens up new possibilities for the fabrication of state-of-the-art hard-soft interfaces structures.

Keywords: *hard-soft interfaces, bone-to-soft tissue interfaces, additive manufacturing, 3D-printing, biomimetic interfaces, computational modeling, geometrical interlocking designs*

Table of contents

1. Introduction

1.1 General introduction

Biological hard-soft interfaces are known to have changes in material properties through variations in chemical composition, microstructure, and geometry [1], *e.g.*, as is observed in musculoskeletal interfaces [2, 3], teeth [4], fish scales [5], sea shells [5], mussel byssal threads [6], and polychaeta jaws [6] (Figure 1A and 1B). The mechanical properties of such materials gradually change over the extent of the interface, providing structural and functional integration between dissimilar materials [7]. In the musculoskeletal system, these hard-soft interfaces are mostly known as bone-to-soft tissue interfaces, transferring loads from the force-generating muscles to the skeleton [8]. Two distinctive types of bone-to-soft tissue interfaces are the enthesis and the osteochondral interface, respectively connecting bone to tendons or ligaments and bone to cartilage.

Bone-to-soft tissue interfaces have proven to be durable under normal loading conditions, but extreme loading conditions and excessive cyclical loading can damage the interface [9, 10]. The incidence of, *e.g.*, anterior cruciate ligament (ACL) reconstructions varies between 60.000 and 200.000 every year in the US [11, 12], and the incidence of rotator cuff tendon reconstructions is approximately 270.000 every year [13]. A large portion of these injuries occurs during sports activities [14, 15]. However, even with modern surgical repair techniques, poor clinical outcomes and high re-tear rates are reported [12, 13]. After an ACL reconstruction, only 50% of the athletes return to their pre-injury level of sports, and between 50% to 100% of these athletes develop some form of knee osteoarthritis. Between 10% to 30% of the returning athletes re-tear their ACL within five years of reconstruction [12]. After rotator cuff reconstructions, for initial tears larger than 5 cm, the reported re-tear rates are even higher, ranging between 40% and 94% [16, 17]. This shows that once failure of the bone-to-soft tissue interfaces occurs, the quality of these tissues gets compromised, and the body cannot regenerate the native complexity of the tissues [18]. Abrupt changes in the mechanical properties can lead to interfacial stress concentrations and make the interfaces more prone to failure [19, 20].

Previous studies have attempted to manufacture bone-to-soft tissue interfaces with a wide range of different techniques [21-24]. However, most of these studies were primarily focused on the behavior of cells in relation to the interface scaffolds and less focused on the mechanical properties of these scaffolds [24-27]. For the most part, the mechanical properties of these scaffolds are insufficient to withstand the range of physiological forces that native bone-to-soft tissue interfaces must endure [28-30]. For example, osteochondral scaffolds with the highest stiffness are fabricated by combining PLGA with HA-particles, resulting in a scaffold that can withstand up to 142 MPa in the bony layer and 62 MPa in the cartilage layer [31]. The stiffness of the bony layer is approximately twice that of the cartilage layer. In contrast, native osteochondral interfaces have a gradient stiffness of an order of magnitude lower [32]. The aim should be to manufacture osteochondral interfaces that can withstand up to 1.8 MPa in the articular cartilage layer [33] and up to 3.7 GPa in the subchondral bone layer [34]. For enthesis scaffolds, the highest stiffness values were reported around 2.5 GPa [35] and 3.1 GPa [36] for the bony end of fibrous gradient scaffolds. These values were obtained by increasing the mineralization levels of polymer fibers towards the bony end of the scaffolds. Especially the stiffest region of these scaffolds compares poorly to the actual stiffness of cortical bone. The aim should be to manufacture enthesis scaffolds that can withstand between 17 to 21 GPa in the bony layer [37, 38] and between 300 to 800 MPa in the tendon or ligament layer [39].

These studies show that manufacturing bone-to-soft tissue interfaces with appropriate mechanical properties remains a considerable challenge. More research is needed to examine the effects of different geometries on the mechanical properties of the interface. The foundation for successful interfacial geometries can be found in nature. Biological materials are highly efficient in using a limited set of materials for a wide range of properties and functions [40]. Distinct geometrical interlocking designs have been found to improve specific interfacial properties, *e.g.*, balancing the strength and toughness of materials through an interfacial brick-and-mortar structure [41, 42] or increasing the energy absorption of a mammalian skull by increasing the sutural interdigitation (Figure 1C) [43]. Studying bio-inspired interlocking designs may help to construct improved solutions for attaching soft orthopedic materials to hard orthopedic implants. These improvements can be used in advanced implants containing pre-made hard-soft interfaces.

Building such structures requires technologies with a high spatial resolution and the ability to print multiple materials on a small scale. One of the few techniques with the potential to fabricate such intricate structures is additive manufacturing, also known as 3D-printing. In this study, we used a combination of extrusion-based 3D-printing techniques and poly-jet based 3D-printing to manufacture 2D (*i.e.*, actually 2.5D) or 3D hard-soft interface structures. The goal was to generate a proof of concept for the fabrication of hard-soft interfaces that is expandable to other 3D-printing techniques and biomaterials. We also explored different geometrical interlocking designs to improve the mechanical performance of hard-soft interfaces. The aim of this study is to develop the foundation for a new generation of bone-to-soft tissue interfaces in orthopedic implants.

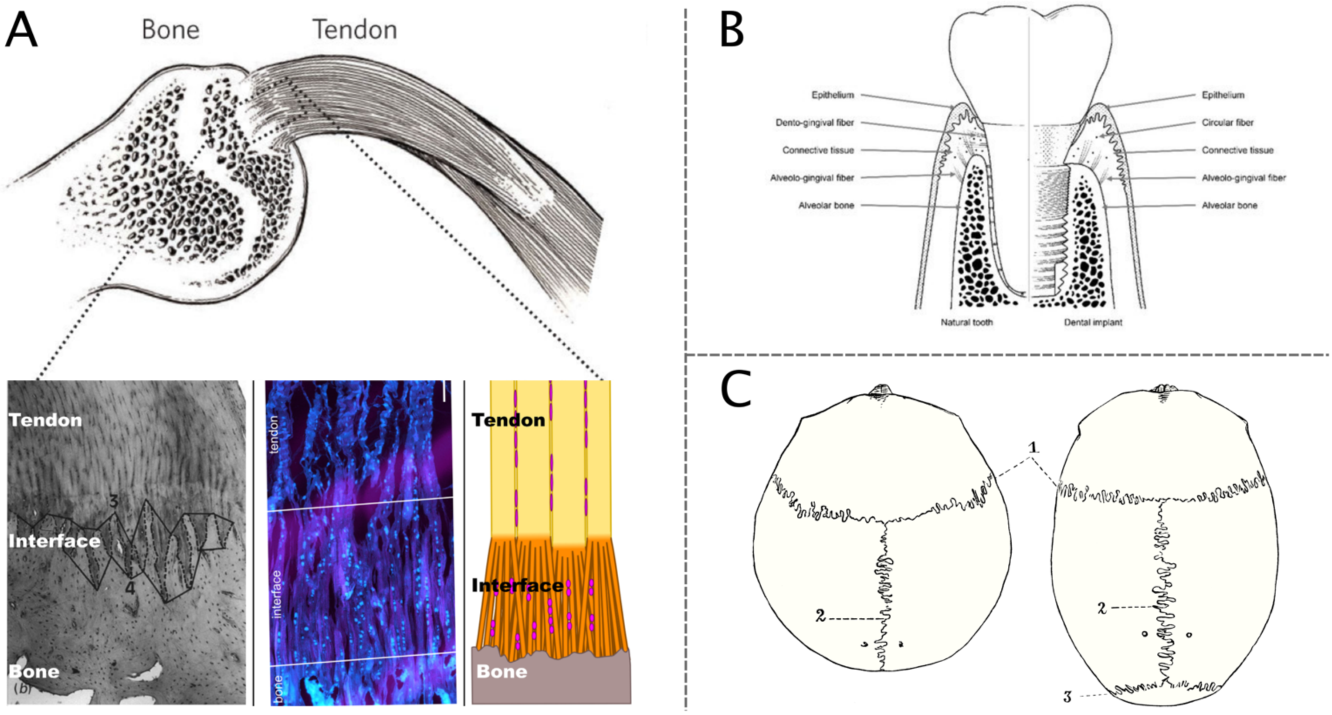


Figure 1. Examples of two hard-soft interfaces and a biological interlocking design. A) Bone-to-tendon interface including a close-up morphological image (left), cell-stained microscopical image (middle), and schematic representation (right). B) Dentin-enamel junction of a tooth (left) and artificial interface (right). C) Sutural interdigitation of a mammalian skull.

1.2 Geometrical interlocking designs

Two distinct geometrical interlocking designs were explored in this study, *i.e.*, anti-trapezoidal and double hook design. Both designs consist of an interface with the ability to mechanically interlock two phases without the need for chemical bonding. The geometry of the interfaces keeps the hard phase in proximity of the soft phase, geometrically interlocking the two separate parts.

Studies have shown that anti-trapezoidal patterns are highly suitable for tensile loading conditions, as they possess a high interfacial strength [42] and a decent toughness and damage tolerance [40]. Compared to other geometrical interlocking designs, *e.g.*, triangular or rectangular geometries, these properties result from an increased strain-to-failure and a favorable mechanical interlocking mechanism [40]. Especially the damage tolerance is an essential property in biomimetic interfaces, as it keeps the hard and soft phases close together, allowing for regeneration of the interfaces after damage has occurred [40].

The double hook design, which resembles an inverted letter T, contains a mechanical interlocking mechanism that is similar to the anti-trapezoidal design. Yet, the 90°-degree angle of the crossover beam has the potential to increase the overall level of mechanical interlocking. This could be beneficial for the performance of the interface under tensile loading conditions. A potential disadvantage could be an increase in stress concentrations around the hooking mechanism.

To control the number of simulations that had to be performed, both designs were simple enough to be parametrized without using a large number of variables, while maintaining the required complexity for their task. The focus of this study was on the mechanical strength of these different geometrical interlocking designs without using adhesives.

1.3 Research questions

The main research question for this study is:

- **How to adjust certain 3D-printing techniques to enable the manufacturing of biomimetic hard-soft interfaces using highly dissimilar materials?**

Additionally, several sub-questions were devised:

- **What are the effects of various geometrical parameters on the performance (*i.e.,* stiffness and stress distribution) of 3D-printed biomimetic interfaces (*e.g.*, anti-trapezoidal and double hook geometrical interlocking designs)?**

- **What are the effects of the geometrical interlocking designs on the strain distributions of those particular hard-soft interfaces?**
- **What are the differences between the 3D-printed hard-soft interfaces and their equivalent computational models?**

1.4 Structure of the study

This study is divided into three parts. First, computational models were created to simulate two series of geometrical interface designs and their parameters, which were systematically altered to examine their effect on the stiffness and stress distribution. In the second part the computational models were validated by 3D-printing a set of these interface designs on 3 mm thick polymer sheets and performing uniaxial tensile tests on them. The tensile tests were accompanied by digital image correlation to obtain the strain distributions of the different samples. The last part consisted of manufacturing hard-soft interface structures using a combination of biomaterials and additive manufacturing techniques. The resulting structures were characterized and tested, serving as a proof of concept for the fabrication of hybrid 3D-printed bone-to-soft tissue interfaces.

2. Materials and methods

An overview of the study is shown in Figure 2. Chapter 2 is divided into three sub-chapters, *i.e.*, computational modeling, 2D hard-soft interface structures and 3D hard-soft interface structures, representing the main lines of research in this study. The 2D hard-soft interface structures are actually 2.5D structures, as the 2D-designs have a third dimension due to the thickness of the samples.

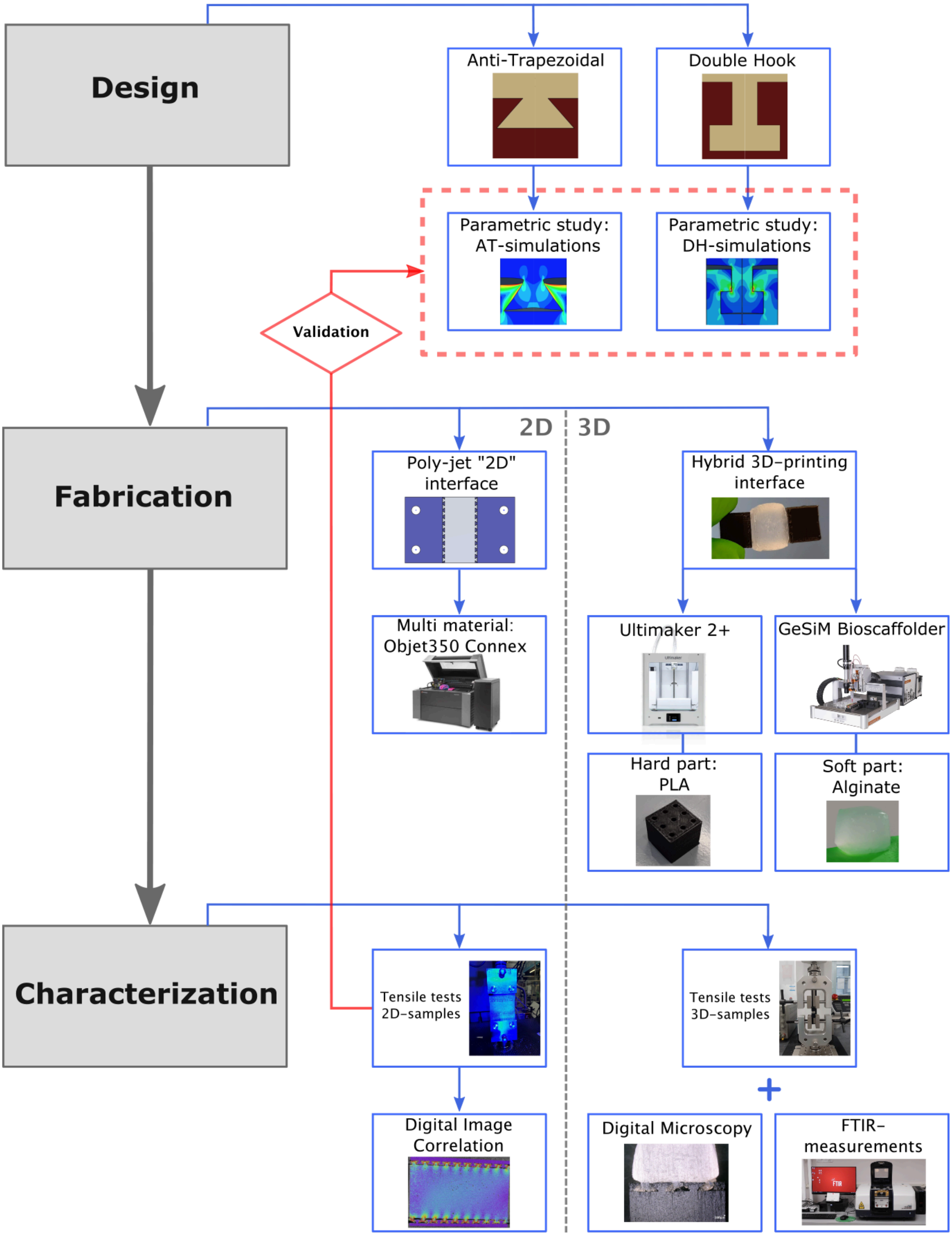


Figure 2. Schematic representation of the study. The materials and methods sections are divided based on different parts of this diagram. The computational modeling is based on the upper design section, the 2D interfaces (*i.e.*, 2.5D) on the left side, and the 3D interfaces on the right side of the fabrication and characterization sections (separated by a dashed gray line).

2.1 Computational modeling

All simulations were performed using a python script (presented in Appendix D & E) and executed in Abaqus/CAE 2017 finite element software from SIMULIA™ (Dassault Systèmes®). MATLAB R2018b (MathWorks®) was used to process the data from Abaqus. The simulations were set up as a parametric study, for which the variables were systematically altered.

2.1.1 Geometry and designs

Two interfacial designs, *i.e.*, an anti-trapezoidal and double hook design (Figure 3), were created and examined for a fixed set of variables in 2D. The anti-trapezoidal model was based on three independent parameters, *i.e.*, the angle of the trapezoid (A), the trapezoid depth (B) and the opening width (C). The double hook model was based on four independent parameters, *i.e.*, the opening width (A), the depth of the model (B), the width of the hook (C) and the thickness of the hook (D). Both models and their parameters are displayed in Figure 3 (step 1). All part lengths and coordinates were calculated from these initial parameters. More information about these calculations is found in Appendix D & E.

The overall design of the computational models was based on three unit-cells, *i.e.*, a hard unit-cell on bottom, an interfacial unit-cell in the middle and a soft unit-cell on top. Each unit-cell is 10 mm wide and high. The designs were symmetrical and therefore cut in half to reduce the computational requirements of the models (Figure 3, step 2). An additional fractional unit-cell is added, either to the top or bottom of the model, to compensate for a difference in surface area between the hard and soft phases. This was done to obtain a 1:1 material ratio between the two phases for all different models, meaning that 50% of the models is hard material and the other 50% is soft material (Figure 3, step 3). The final dimensions of the models are 5 mm wide and 30 mm high, plus the additional height of the fractional unit-cell.

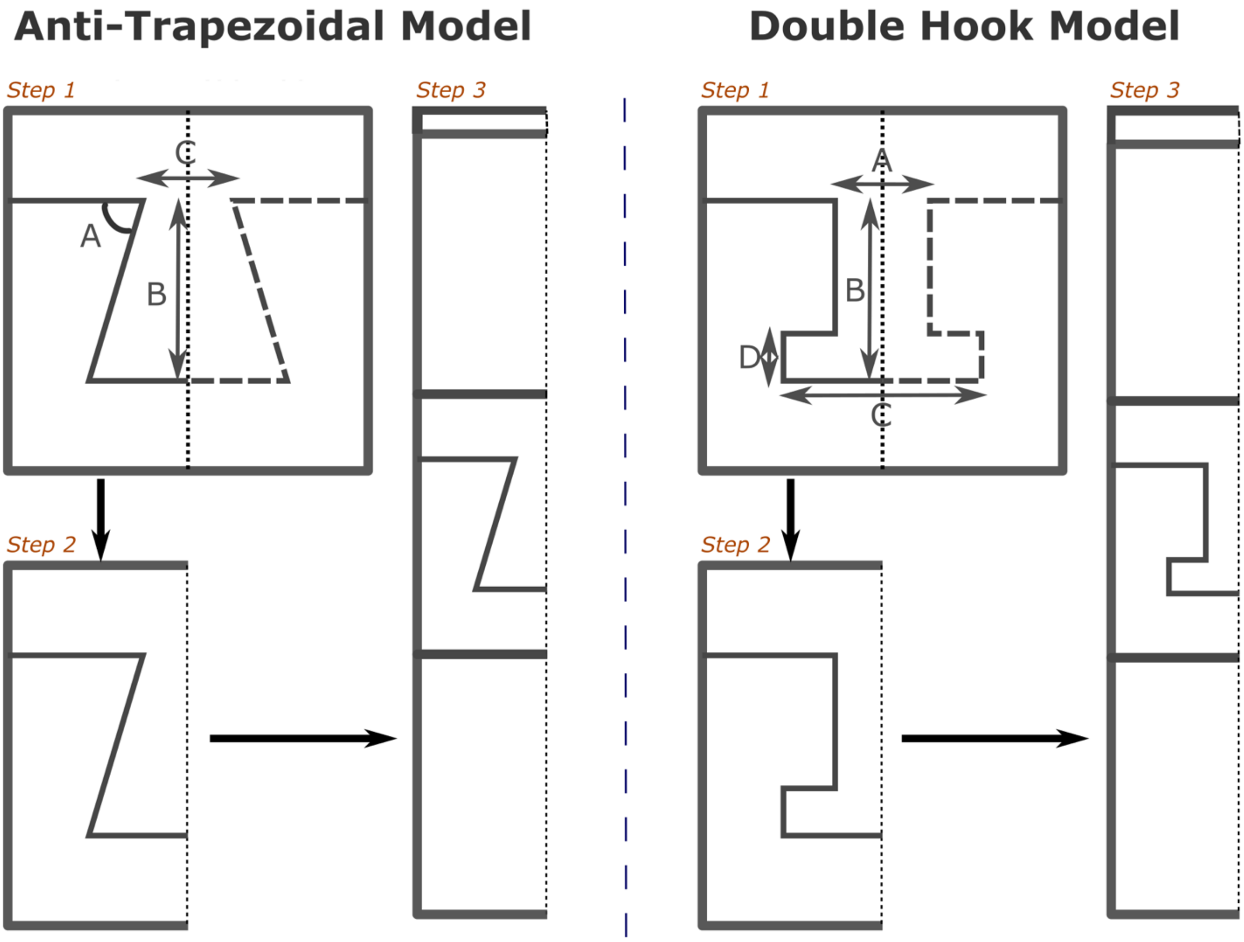


Figure 3. Schematic representation of the two designs and corresponding models, *i.e.*, anti-trapezoidal (left) and double hook model (right). Step 1 illustrates their designs with parameters, step 2 the half unit-cells, and step 3 the complete half unit-cell models with an added fractional half unit-cell.

2.1.2 Material properties

The model contains linear elastic material properties, using polylactic acid (PLA) for the hard phase and crosslinked sodium alginate for the soft phase. The stiffness of sodium alginate was set at 190 MPa with a Poisson's ratio of 0,495, which were obtained after tensile testing sodium alginate blocks that were crosslinked in a 200 mM calcium chloride solution for 24 h [44]. The stiffness of PLA was set at 3500 MPa with a Poisson's ratio of 0,36, which were retrieved by the authors after reviewing the literature on the properties of bulk PLA [45].

Table 2.1. Overview of the material properties used in Abaqus [44, 45].

	Young's Modulus (E) in MPa	**Poission's ratio (ν)**
Hard phase	3500	0,36
Soft phase	190	0,495

2.1.3 Meshing of the model

The model was meshed using plane strain elements, which assume zero strain in the z-direction, *i.e.*, the out of plane direction. This can be visualized as a model that is very long in the out of plane direction. The element types were 4-node bilinear elements with reduced integration and hourglass control (CPE4R). The elements were shaped as quad-shaped elements, which are the only elements that accurately measure warpage (how "bent" an element is). The global seed size between the nodes was 0.4 mm with a decreased local seed size of 0.1 mm around the area of interest, *i.e.*, the interface. The distance between the local seeds was selected after a mesh refinement study at 0.4, 0.3, 0.2, 0.1 and 0.05 mm (see Appendix A). The coarsest mesh with a local seed size of 0.4 mm had approximately 1000 elements and the finest mesh at 0.05 mm had 16000 elements. Point A was defined as the top left corner of the hard part of the anti-trapezoidal model and Point I was defined as the upper corner of this same part (Figure S1, Appendix A). Point I was also the zone with the highest stress concentrations. The models were examined for the von Mises stresses (point A and I) and for the nodal displacement (point I). The results were inspected for convergence as well as the required CPU time to run the simulation. The mesh refinement study shows that the simulations converge around a local seed size of 0.1 mm. Further refinement of the mesh produces negligible changes in the solution, but significantly increases the computational time required for the simulations.

2.1.4 Model interactions

The contact between the hard and soft parts is a surface-to-surface contact, accommodating the tangential behavior with a friction coefficient of 0,05. This is a relatively low friction value, which was chosen to model an interface with little interaction between the different surfaces. The friction formulation was set at "penalty", resulting in a model that had a small fraction of allowable elastic slip.

2.1.5 Loads and boundary conditions

The load of the half unit-cell model was placed at a reference point on the top surface of the model, connecting all nodes on this surface to a single reference point in the middle of it. A tensile displacement of 10 mm in the y-direction was applied at this point. Additionally, the model contained a reference point on the right edge of the model, connecting all nodes on this edge to a second reference point. The nodes on this edge remained aligned with the reference point in the x-direction according to the following equation (2.1).

$$Ni_x - RPx = 0 \tag{2.1}$$

Hereby, Ni_x is the position of node number i in the x-direction and RPx is the position of the reference point in the x-direction.

This equation generates periodic boundary conditions (PBCs) for the model. The PBCs create a large set of constraints in the x-direction, resembling a virtually infinite system of unit-cells. The left edge of the model contains a boundary condition producing symmetry for the model in the x-direction. The bottom edge of the model contains a symmetrical boundary condition in the y-direction, keeping the bottom edge of the model in place during the tensile displacement.

2.1.6 Type of analyses

The simulations were performed under static general conditions, ignoring the time-dependent behavior of materials. The model accounted for non-linear effects as a result of large potential displacements of the soft phase. The automatic stabilization function was activated, considering the unstable nature of non-linear static problems. This function was set to dissipate the energy fraction with adaptive automatic stabilization, varying the damping factor based on the strain energy of the model. For this function the default tolerance of 0.05 was used, which is suitable for most applications. A fixed time incrementation of 10^{-3} seconds created equal intervals between the output data points.

2.1.7 Set-up of simulations

The anti-trapezoidal simulations were examined for three independent parameters. The first parameter, *i.e.*, the angle (A), was examined at 90°, 75°, 60°, 45° and 30°. Angles larger than

90° were not considered, as the model would stop resembling an anti-trapezoidal design. The depth (B) and opening width (C) of the trapezoid were examined from 1 mm until 5 mm with intervals of 1 mm. For some combinations of parameters, the resulting geometry exceeded the boundaries of the unit-cell. These simulations were excluded from the study, resulting in a total of 72 anti-trapezoidal simulations performed.

The double hook simulations consisted of four independent parameters, *i.e.*, opening width (A), depth (B), width of the hook (C) and depth of the hook (D), resulting in a substantially larger number of possible simulations than the anti-trapezoidal model. For this reason, a different approach was chosen for the double hook simulations. To start, a set of rules was created to ensure that the model's shape consistently resembled an inverted letter "T" (presented in Figure 4). Then the simulations were performed by examining three essential aspects for this design, *i.e.*, the relative size of the design, the A:C ratio, and the B:D ratio. This set-up allowed for a lower number of simulations, *i.e.*, 12 simulations in total, while systematically extracting the most relevant information.

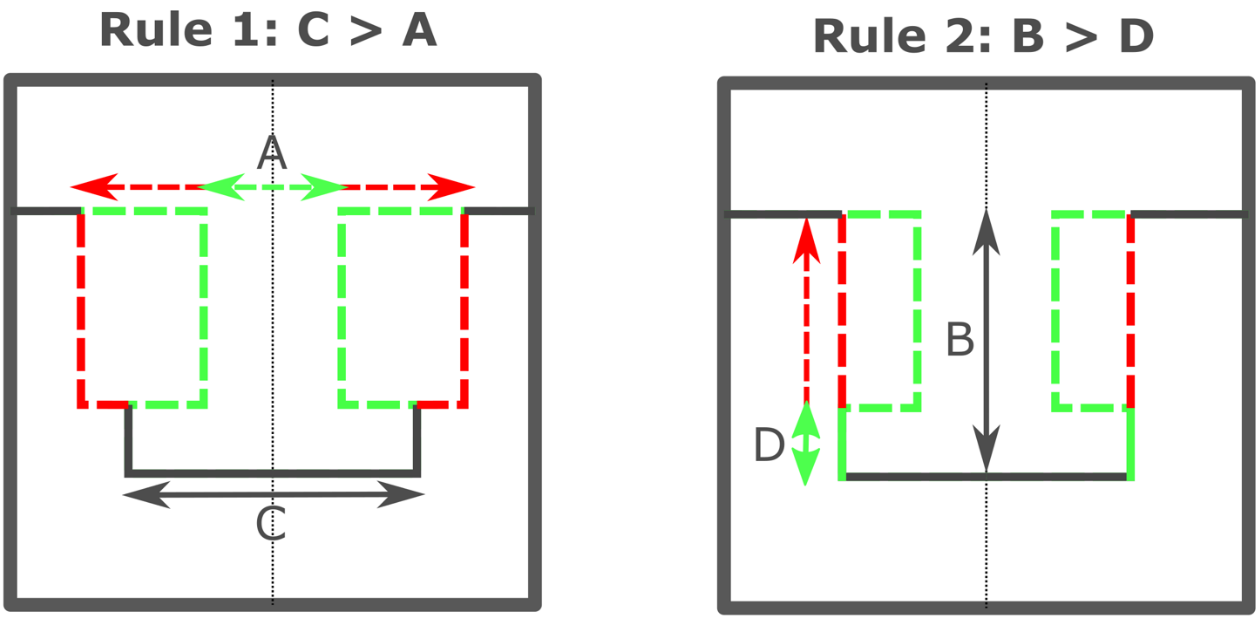


Figure 4. Schematic representation of the rules for the double hook design. The image visualizes in red and green what happens to the shape of the inverted "T", if parameter A changes in relation to C, or if D changes in relation to B. Parameters C and B have a fixed length in this example.

2.2 Manufacturing of 2D hard-soft interface structures

The 2D hard-soft interface structures were based on the computational models and named after these models. The anti-trapezoidal samples follow the naming convention “AT A-B-C”, *i.e.*, anti-trapezoidal followed by parameter A, B and C. The double hook samples follow the convention “DH A-B-C-D”, *i.e.*, double hook followed by parameter A, B, C and D.

2.2.1 Poly-jet designs

Four interface models were selected, *i.e.*, two anti-trapezoidal (AT 45-3-3 & AT 60-4-3) and two double hook (DH 6-8-8-7 & DH 4-8-6-6) models, for the 2D hard-soft interface structures (Figure 5A-B). These models were chosen from the computational models and consisted of two top designs. These designs were not necessarily the two best performing designs, but one of the better performing designs, while making sure that they differed noticeably and the smallest material features were not too thin. A cut-off value of 1 mm was chosen for the smallest feature, otherwise the materials might become too fragile. The 2D samples consisted of two hard parts (top and bottom) and one soft part (middle), connected by two identical interfaces (Figure 5C).

2.2.2 Printing of 2D samples

The 2D hard-soft interface structures were printed with the Objet350 Connex3 poly-jet 3D-printer (Stratasys® Ltd., USA). Each structure consisted of two parts of hard material, *i.e.*, VeroCyanTM (RGD841, Novamatrix) and one part of soft material, *i.e.*, Agilus30TM (FLX935, Novamatrix). Their material properties were obtained from the literature and are presented in Table 2.2 [46]. All parts were printed separately and assembled after the removal of support material. No adhesives were used at the interface. The width and height of the samples was 75 x 125 mm^2 and they were 3 mm thick (detailed dimensions in Figure 5C). The height of the interface varied slightly between the samples to maintain a 50:50% material distribution. This 50:50% material distribution was defined over the 75 mm of the sample (height) that was not enclosed by grippers (Figure 5C-D).

Table 2.2. Material properties for 2D poly-jet samples [46].

Biomaterials	Young's Modulus (E) in MPa	Poission's ratio (ν)
VeroCyanTM	735	0.33
Agilus30TM	0.8	0.45

2.2.3 Digital image correlation and testing machine

The force and displacement of the 2D samples was measured with the Ametek LR-5K mechanical testing machine (Lloyds Instruments Ltd., UK) under tensile loading conditions. The testing machine was equipped with a 100 N XLC-series load cell (Lloyds Instruments Ltd., UK). The samples were loaded with a rate of 2 mm/min until complete separation of one of the interfaces. The force (in N) and displacement (in mm) were extracted from the Lloyds machine and used to calculate the (engineering) stresses, strains and stiffness of the samples.

The local strains and displacements on the sample's surface were examined using digital image correlation (DIC) with the Q400 DIC (LIMESS Messtechnik & Software GmbH, Germany). The DIC-software uses a pattern of black speckles on a white painted surface to calculate changes in coordinates of the speckles. It was important to obtain a speckled mesh that was as fine as the airbrush allowed for maximal accuracy of the correlations. The sample's surface was painted with the Evolution Two-in-One airbrush (Harder & Steenbeck, the Netherlands). The von Mises strains, *i.e.*, a measure for the normal and shear strains, were analyzed using Istra4D software (Dantex Dynamics), providing strain patterns for the different interface designs. The experimental set-up is shown in Figure 5D-E.

2.2.4 Post-processing

The strain distributions on the sample's surface are examined at 25%, 50% and 75% of the total strain applied, for which equation 2.2 provides a clarification:

$$\mathcal{E} = \; 25\% \times \mathcal{E}_{ult} \qquad (2.2)$$

Hereby, $\mathcal{E}$ is the current strain and $\mathcal{E}_{ult}$ is the total strain applied. The total strain is defined as the moment of complete separation between the hard and soft phase for either one of the interfaces, *i.e.*, the top or bottom interface. All samples were stretched in the vertical direction of the reader (Figure 5). The evaluated strain is the true effective (von Mises) strain, considering the strains in all directions and the shear strains (equation 2.3) [47]:

$$\mathcal{E}_{vM} = \frac{2}{3}\sqrt{\frac{3(e_{xx}^2 + e_{yy}^2 + e_{zz}^2\,)}{2} + \frac{3(\gamma_{xy}^2 + \gamma_{yz}^2 + \gamma_{zx}^2)}{4}} \qquad (2.3)$$

The deviatoric strains are defined as:

$$\begin{aligned} e_{xx} &= \frac{2}{3}\mathcal{E}_{xx} - \frac{1}{3}\mathcal{E}_{yy} - \frac{1}{3}\mathcal{E}_{zz} \\ e_{yy} &= -\frac{1}{3}\mathcal{E}_{xx} + \frac{2}{3}\mathcal{E}_{yy} - \frac{1}{3}\mathcal{E}_{zz} \\ e_{zz} &= -\frac{1}{3}\mathcal{E}_{xx} - \frac{1}{3}\mathcal{E}_{yy} + \frac{2}{3}\mathcal{E}_{zz} \end{aligned} \qquad (2.4)$$

The engineering strains are defined as:

$$\gamma_{ij} = 2 \times \mathcal{E}_{ij} \qquad (2.5)$$

The 3D strain matrix is given by $\boldsymbol{\epsilon}$:

$$\boldsymbol{\epsilon} = \begin{pmatrix} \mathcal{E}_{xx} & \mathcal{E}_{xy} & \mathcal{E}_{xz} \\ \mathcal{E}_{yx} & \mathcal{E}_{yy} & \mathcal{E}_{yz} \\ \mathcal{E}_{zx} & \mathcal{E}_{zy} & \mathcal{E}_{zz} \end{pmatrix} \qquad (2.6)$$

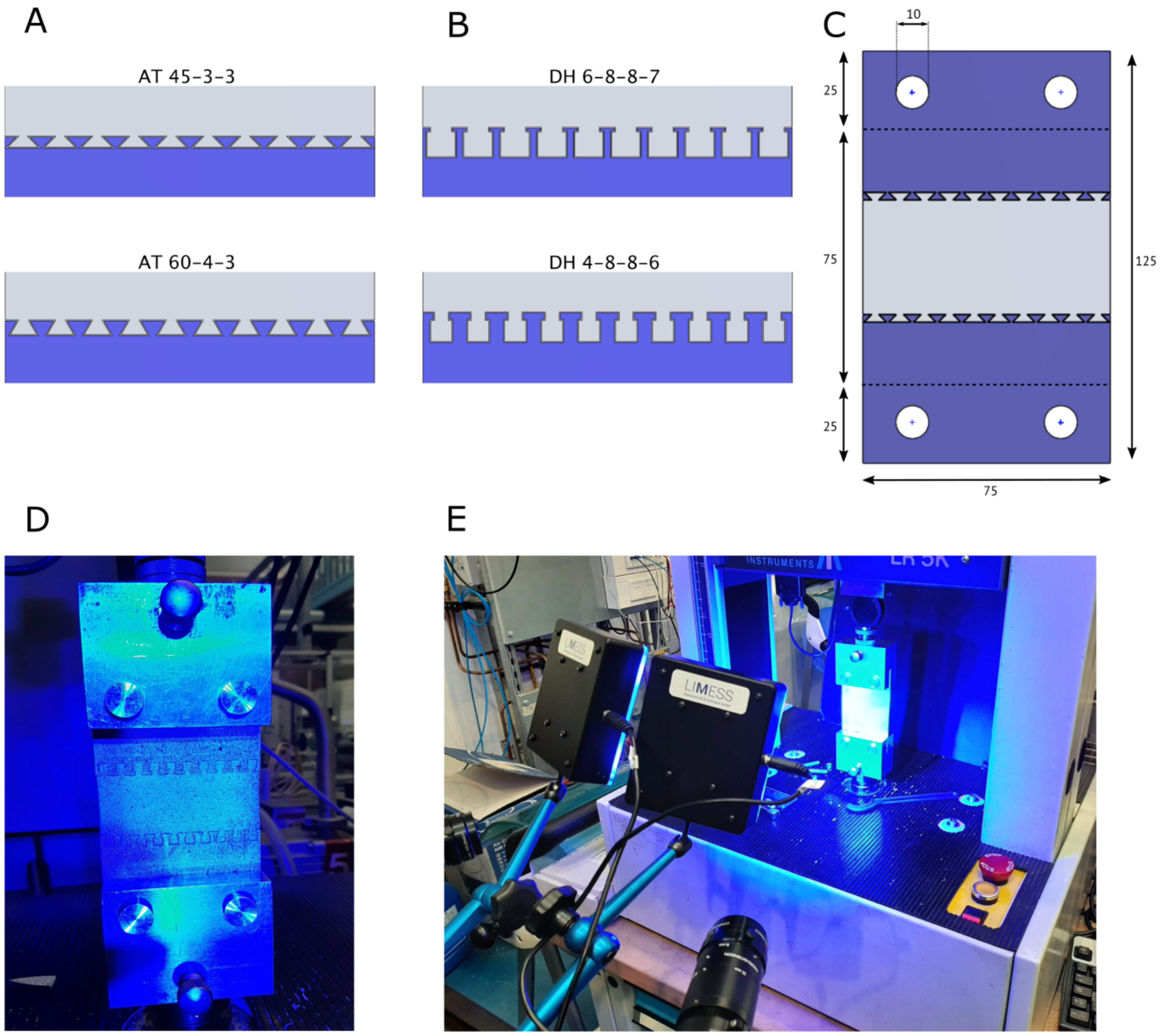


Figure 5. A) Anti-trapezoidal interface designs for poly-jet printing, and B) double hook designs. C) Schematic illustration of 2D poly-jet sample including dimensions. Blue parts are hard material and gray soft material. D) Sample attached to grippers for tensile test. E) Experimental set-up with DIC and mechanical testing machine.

2.3 Manufacturing of 3D hard-soft interface structures

The 3D hard-soft interface structures were based on the same interface designs as the 2D structures, *i.e.*, AT 45-3-3, AT 60-4-3, DH 6-8-8-7 and DH 4-8-6-6.

2.3.1 Biomaterials

The 3D hard-soft interfaces structures consisted of a hard material, *i.e.*, a mixture of polylactic acid (PLA) and acrylic (Tough PLA, Ultimaker B.V.), and a soft material, *i.e.*, ultrapure sodium alginate (PRONOVA, Novamatrix). A 0.9% (w/v) NaCl solution (saline, Sigma Aldrich) was used to fabricate alginate-based hydrogel. An aqueous solution of $CaCl_2$ (Sigma Aldrich) was used to crosslink the alginate.

Table 2.3. Biomaterials used to manufacture combined PLA-hydrogel structures

Biomaterials	Manufacturer
PLA	Ultimaker B.V.
Sodium Alginate	Novamatrix
Calcium Chloride	Sigma-Aldrich
Sodium Chloride	Sigma-Aldrich

2.3.2 Additive manufacturing of hard PLA parts

The hard PLA parts were printed with FDM technique using the Ultimaker 2+ (Ultimaker USA Inc., USA) 3D-printer (Figure 6A). The 3D-printing was performed with a 0.4 mm nozzle size, a nozzle temperature of 220° C, a build plate temperature of 60° C, and a material flow of 110%. Other settings were kept at the initial values set by the company. The PLA cubes with width, length and height of 10 x 10 x 8 mm^3 (Figure 6B) were designed with Solidworks (Dassault Systèmes®). Before printing, the CAD-models were sliced using Ultimaker Cura 4.6 software. A 100% infill density and a line-based infill pattern were selected to create solid PLA structures. Afterwards, the G-code was generated for 3D-printing (Figure 6C-D).

2.3.3 Fabrication of bio-ink

The 3D printable alginate-based bio-ink was prepared by adding 2.5% (w/v) of sodium alginate into a saline solution, followed by a 4-h of magnetic stir at room temperature. The alginate solution was stored at 4° C for 24 h to allow the resting and swelling process to take place for hydrogel formation. Thereafter, the alginate hydrogel was mixed with $CaCl_2$ solution to acquire a pre-crosslinked alginate bio-ink with 20 mM $CaCl_2$ (see appendix B for further information on solution preparation). The solutions were mixed together with the double syringe method, passing the substances back and forth through the syringes connected by a double-sided Luer-lock connector (Figure 6F). The bio-ink was homogenized by manually extruding it into a syringe with an increasingly smaller nozzle size. Finally, the bio-ink was centrifuged to remove air bubbles with the Allegra X-22 Centrifuge (Beckman Coulter), at 200

RCF for 5 minutes. The 20 mM $CaCl_2$ value was obtained after preliminary trial-and-error experiments used to examine the printability of different viscosities of bio-ink.

2.3.4 Additive manufacturing of soft alginate parts

The softer alginate structures were printed with the GeSiM Bioscaffolder 3.2 (GeSiM Bio-Instruments and Microfluidics, Germany; Figure 6E). The printing was performed with a 0.58 mm nozzle tip (SmoothFlow, Nordson EFD), an air pressure of 20 kPa, and a printing speed of 10 mm/s. Solid alginate cubes with a width, length and height of 10 x 10 x 10 mm^3 were designed using the tool-related GeSiM Robotics software.

2.3.5 Manufacturing of combined hard-soft interface structures

The location of the interlocking holes relative to the build plate of the GeSiM Bioscaffolder were carefully measured. The printer filled up the holes by hoovering the nozzle tip above the opening and extruding hydrogel for 0.85 seconds per hole (Figure 6G). Afterwards, the soft alginate cubes were printed on top (Figure 6H). The combined structure is submerged in 200 mM $CaCl_2$ solution and left to crosslink for 24 h. It was important not to crosslink the structure during printing as the layers would not bond this way. The tensile structures were manufactured by repeating the previous procedure twice. One scaffold was printed 8 mm and the other 2 mm high. The separate scaffolds are joined together on a well-plate with the soft parts facing each other (Figure 6I), after which they are also crosslinked for 24 h (Figure 6J & 6K).

2.3.7 Characterization

The interface characteristics of the PLA-alginate structures, *i.e.*, the inflow of alginate into the interlocking system, were observed using a Keyence digital microscope (Digital Microscope VHX-5000, Keyence America) and VH-Z20T lens with magnification of 20 to 200x. Cross-sections of the structures were obtained using a water-jet cutter, *i.e.*, the Omax MicroMAX abrasive waterjet machine with a precision of 0,1 micron and advanced pressure control for piercing through delicate materials. The chemical compounds of the hydrogel were characterized using Fourier-transform infrared spectroscopy (FTIR, Nicolet™ FTIR Spectometer, ThermoFisher Scientific).

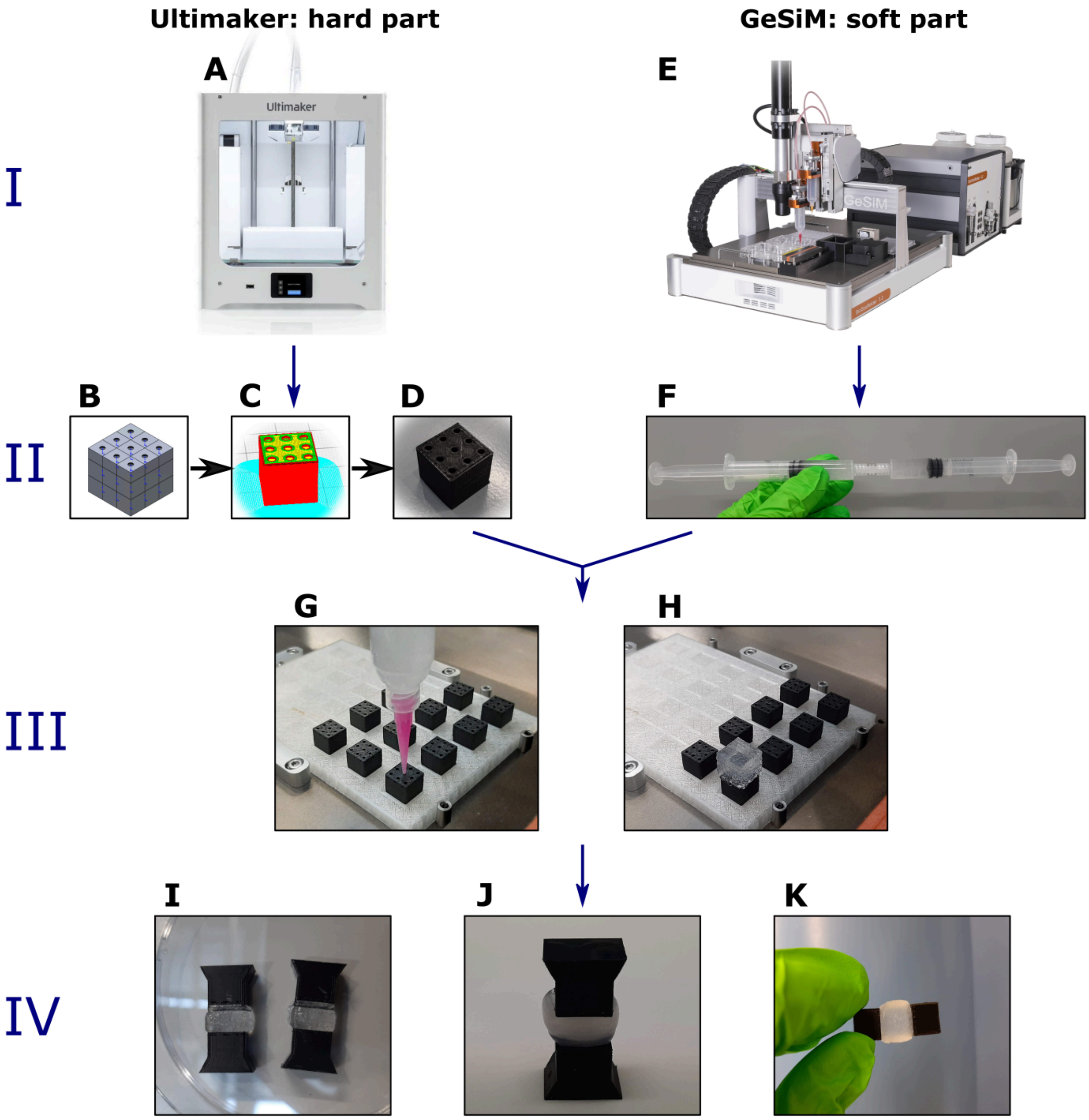


Figure 6. Schematic representation of the fabrication process of the 3D hard-soft interface structures. The process is divided into four layers, *i.e.*, the 3D-printers, the requirements for printing with the machines, the manufacturing process and the final products. A) The Ultimaker 2+ 3D-printer with illustration of the printing process from B) CAD-model, C) sliced model in Cura, to D) 3D-printed PLA cube. E) The GeSiM Bioscaffolder with illustration of F) the double syringe method to obtain a printable bio-ink. G) Extrusion of hydrogel into the interlocking system, followed by H) a printed hydrogel scaffold on top of a PLA cube after the filling process was completed. I) Image of tensile PLA-alginate-PLA structures before crosslinking. J) Tensile sample after crosslinking for 24 hours in a 200 mM calcium chloride solution. K) Example of a sample demonstrating some interfacial strength.

3. Results

3.1 Computational modeling

This chapter presents the results of the anti-trapezoidal and double hook simulations. The force-displacement curves of the models were extracted from Abaqus and used to calculate the stiffness of the different designs. The stiffness determines the amount of resistance to deformation that the different interface designs exhibit. It would have been preferable to also derive the ultimate tensile strength from the data, but the simulations were aborted by Abaqus before the hard and soft phase could completely detach. Therefore, the ultimate tensile strength of the models was not obtainable.

3.1.1 Stiffness anti-trapezoidal models

The stiffness values for the anti-trapezoidal simulations are reported in Appendix C. Some combinations of A-, B- and C-parameters, resulted in a geometry that exceeded the boundaries of the unit-cell model and were disregarded. These combinations are shown in the tables by a dash (-). The simulation at 90° was performed as a control simulation, because the tensile force vector is completely parallel to the resisting surfaces. This results in a non-resistive motion of the soft interlocking piece, moving the piece out of the cavity of the hard phase. It is important to note that there was no chemical bonding or adhesive layer between the phases. The resulting stiffness for the control simulation was 0 MPa.

The most important results from the anti-trapezoidal simulations are shown in Figure 7-11. Figure 8 shows the stress-strain curves of the designs with the highest stiffness values per angle. The control simulation at 90° was not included, because it was a horizontal line with no stiffness. It was to be expected that angles close to the 90° would lead to low stiffness values, as there is less resistance between the phases. The stress distributions of the models with the highest stiffness per angle support this statement (Figure 7). This can be seen from the increasing stress concentrations for each consecutive angle and the relatively low stress concentrations in the 75° angle. The upper piece represents the soft part and the lower piece represents the hard part.

The stress-strain curve at 75° supports this observation, showing the lowest stiffness of the four examined angles (Figure 8A). The intermediate angles, *i.e.*, 45° and 60°, have relatively similar stress-strain curves for their best performing designs and show the highest stiffness values. The more pronounced angle, *i.e.*, 30°, shows a stiffness that is lower than the intermediate angles and higher than the less pronounced 75° angle. All angles show a linear graph before the aborting the simulation, except the 30° angle.

Similar trends are observed for the median stiffness per angle. The median stiffness of the intermediate angles, *i.e.*, 45° and 60°, are identical and show the highest stiffness values. The 30° angles rank below the intermediate angles and the 75° angles show the lowest stiffness values (Figure 8B).

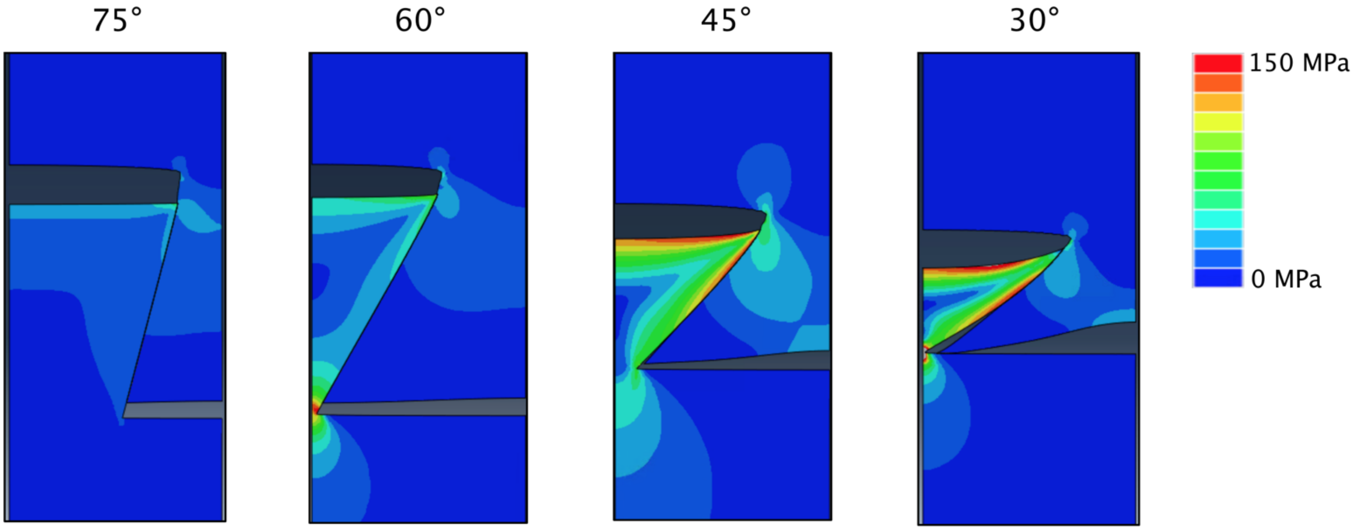


Figure 7. Strain distributions (von Mises) of the designs with the highest stiffness per angle. All strain distributions were captured at 0.025 strain. The stress range is illustrated by the color scale on the right.

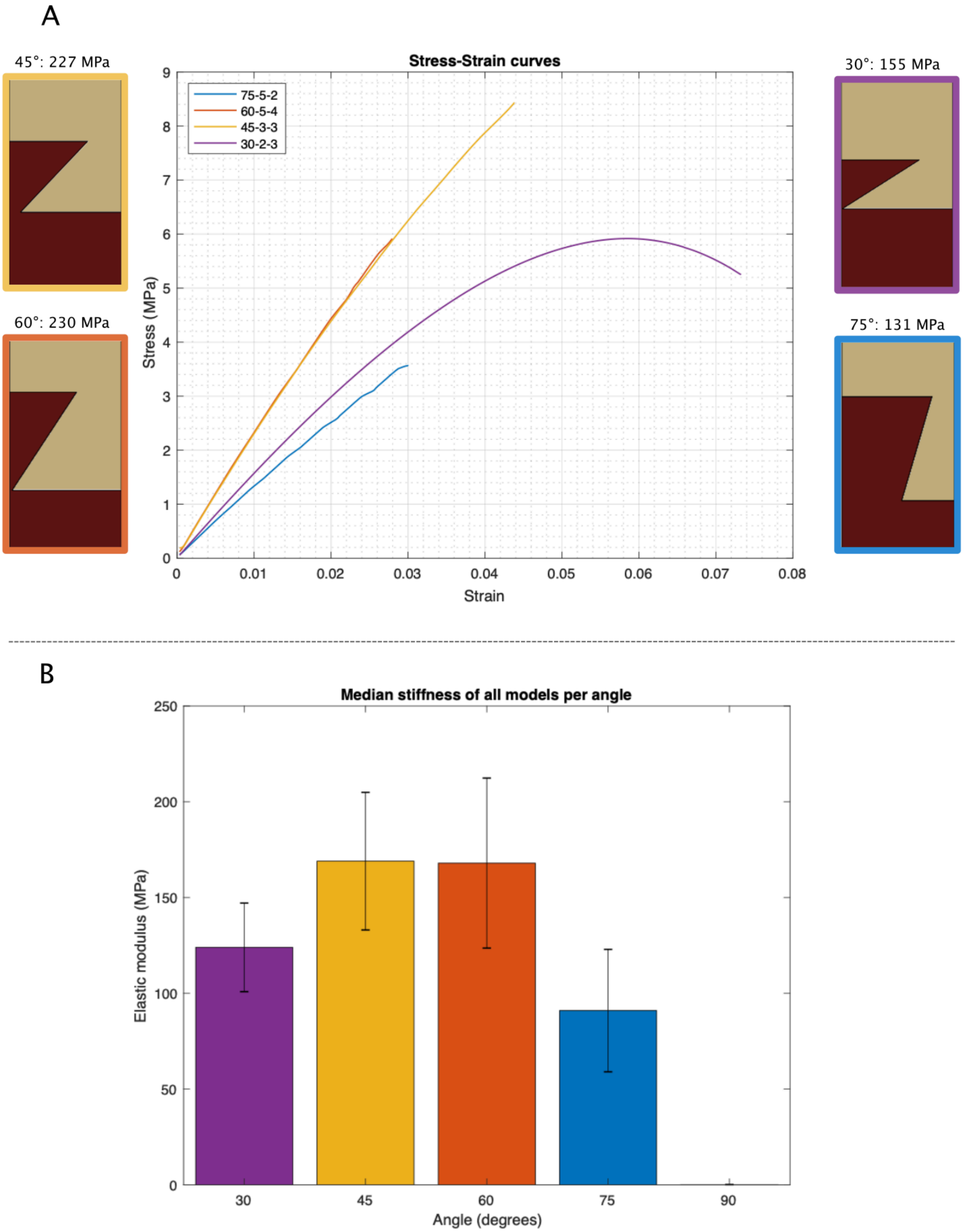


Figure 8. A) Stress-strain curves of the designs with the highest stiffness per angle, *i.e.*, 30°, 45°, 60° and 75°, including illustrations and stiffness values. The beige part is soft and dark brown part is hard. B) Bar plot showing median stiffness and standard deviation for each angle, including the 90° angle.

Figure 9 compares the model with the highest stiffness versus the model with the lowest stiffness, revealing differences in their geometrical characteristics. A comparison between these extremes shows, besides the angle, differences in relative size of the models, the contact area of the resisting edges, the opening width and depth of the models. The resisting edge is the inclined edge of the model that is able to exert forces under tensile loading conditions.

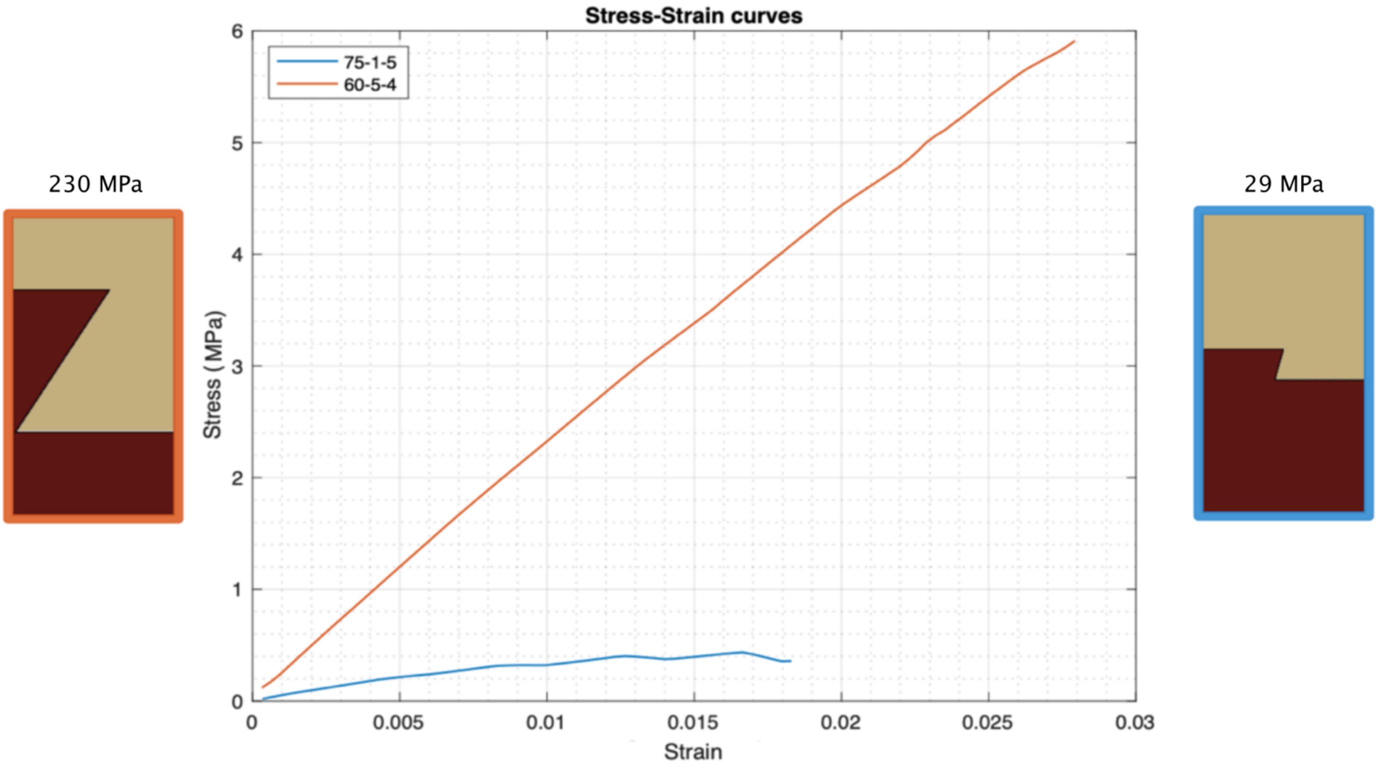


Figure 9. A) Stress-strain curves of the highest stiffness versus lowest stiffness design, including illustrations and stiffness values.

Figure 10 presents the scatterplots for several geometrical characteristics and stiffness values, showing some notable trends for the anti-trapezoidal simulations. It appears that designs with a lower surface area of the trapezoid generally have lower stiffness values than designs with a higher surface area (Figure 10A). This trend is even more apparent when considering the angles separately, which is visualized in the scatterplots by using different colors per angle. A similar observation is made for the contact area of the resisting edge and the stiffness of the models. Models with higher contact area of the resisting edge have higher stiffness values than models with lower contact area (Figure 10B).

The scatterplots show no linear regression line or R^2-values as the trends of data points appear to be non-linear. Both the surface area of the trapezoid and the contact area of the resisting edge contribute to the stiffness of the designs. It is understandable that both these features contribute to a higher stiffness, as they also appear highly correlated to each other (Figure 10C). An increase of the surface area, almost by definition, results in an increase of the contact area of the resisting edge for all models. The dots on horizontal lines in the third plot, showing an identical contact area of the resisting edge, are models of the same angle and depth but with different opening widths.

Figure 10D shows that models with larger opening widths and low depths of the trapezoid, *i.e.*, broad but shallow interlocking designs, have lower stiffness values than models with smaller opening widths and larger depths, *i.e.*, deep but thin interlocking designs. Even if the structures are shallow, but less broad, the stiffness is higher than for shallow and broad designs. The opening width and depth of the trapezoid are translated to a C:B ratio, which is the opening width divided by the depth of the trapezoid.

The illustrations in Figure 9 clarify the effects of the geometrical characteristics, *i.e.*, a lower surface area, lower contact area of the resisting edge, and higher opening width vs. depth ratio, resulting in a lower stiffness for the AT 75-1-5 model compared to the AT 60-5-4 model. Additionally, the scatterplots in Figure 10A, B and D, show that the observed trends hold true for all angles. The scatterplots also confirm that the 75° angles have the lowest stiffness values, followed by the 30° angles and finally the intermediate 45° and 60° angles, showing the highest stiffness values.

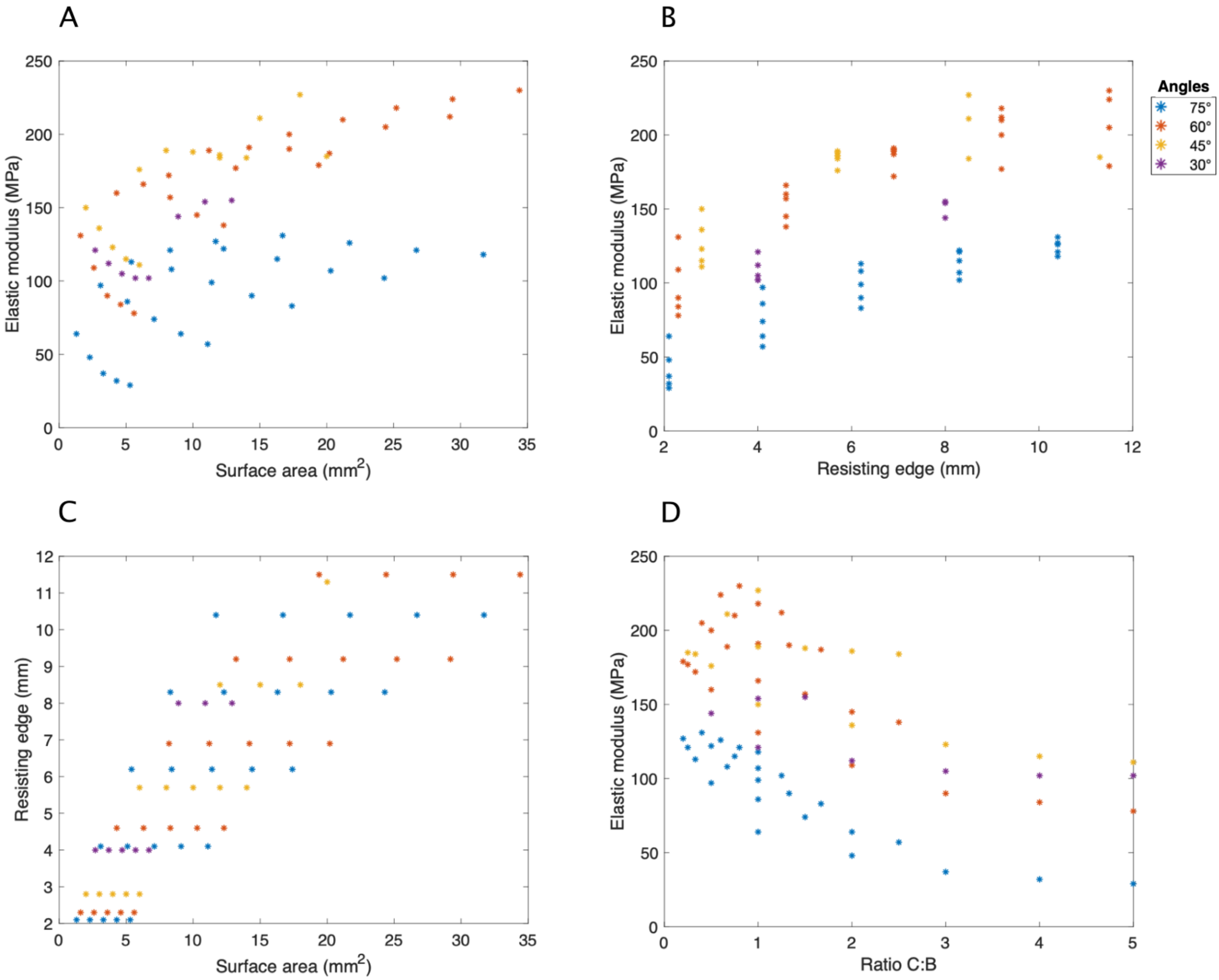


Figure 10. A) Scatterplots comparing different characteristics of the anti-trapezoidal designs, *i.e.*, a comparison between A) surface area vs. stiffness, B) contact area of the resisting edge vs. stiffness, C) surface area vs. contact area of the resisting edge, and D) opening width divided by the depth (C:B ratio) vs. stiffness.

Figure 11 visualizes the three anti-trapezoidal designs with the highest stiffness values. The stress-strain curves and respective stiffness values of these designs are remarkably similar. This could suggest that a limit in stiffness was reached for this set of parameters. Further increasing the opening width and depth of the trapezoid within the boundaries of the model is not possible, because the trapezoid would exceed the unit-cell. It would be possible to increases these parameters, *i.e.*, B and C, for angles closer to 90°, but the simulations have shown that the stiffness decreases significantly at these angles. This suggests that additional features are necessary to improve the model further, since the current parameters appear to have reached their limits.

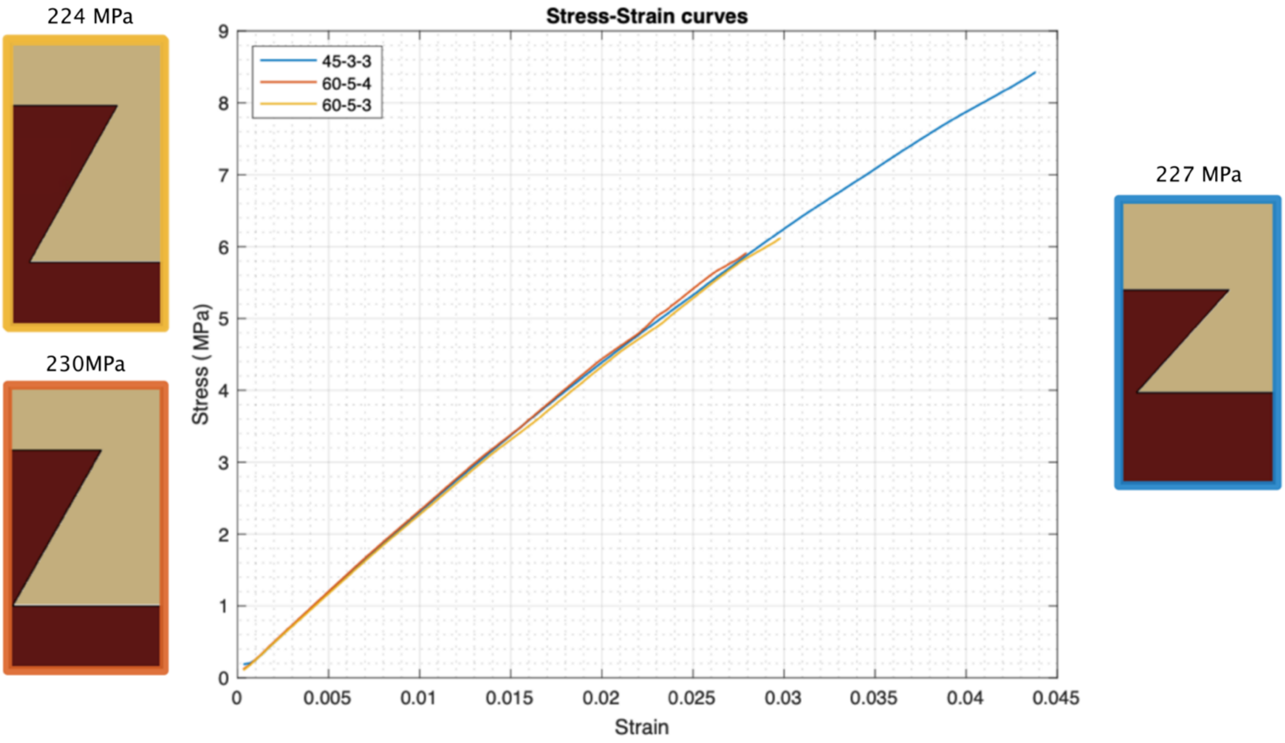


Figure 11. A) Stress-strain curves of the three highest stiffness designs, including illustrations and stiffness values.

3.1.2 Stiffness double hook models

The first set of double hook simulations examined the relative size of the models in relation to the size of the unit-cell. Parameter A was kept equal to parameter D, and parameter B was kept equal to parameter C. This ensured that the model maintained its shape throughout this set of simulations. Parameter B was defined as two times parameter A, for which parameter A was increased from 1 to 4 mm with steps of 1 mm. Figure 12 presents the stress-strain curves for this first set of simulations. The double hook illustrations in Figure 12-15 detail the design of the interfacial (half) unit-cells and not their entire models. The stress-strain curves for the first set of simulations show a similar pattern of linearly increasing stress per unit of strain. There is a clear trend observable between the size of the interlocking pieces and the stiffness of the models. Larger interlocking pieces have higher stiffness values than smaller interlocking pieces. The DH 4-8-8-4 model possesses the highest stiffness for this set of simulations. For this reason, this model was taken as the starting point for the second set of simulations, *i.e.*, the simulations examining the A:C ratio of the models.

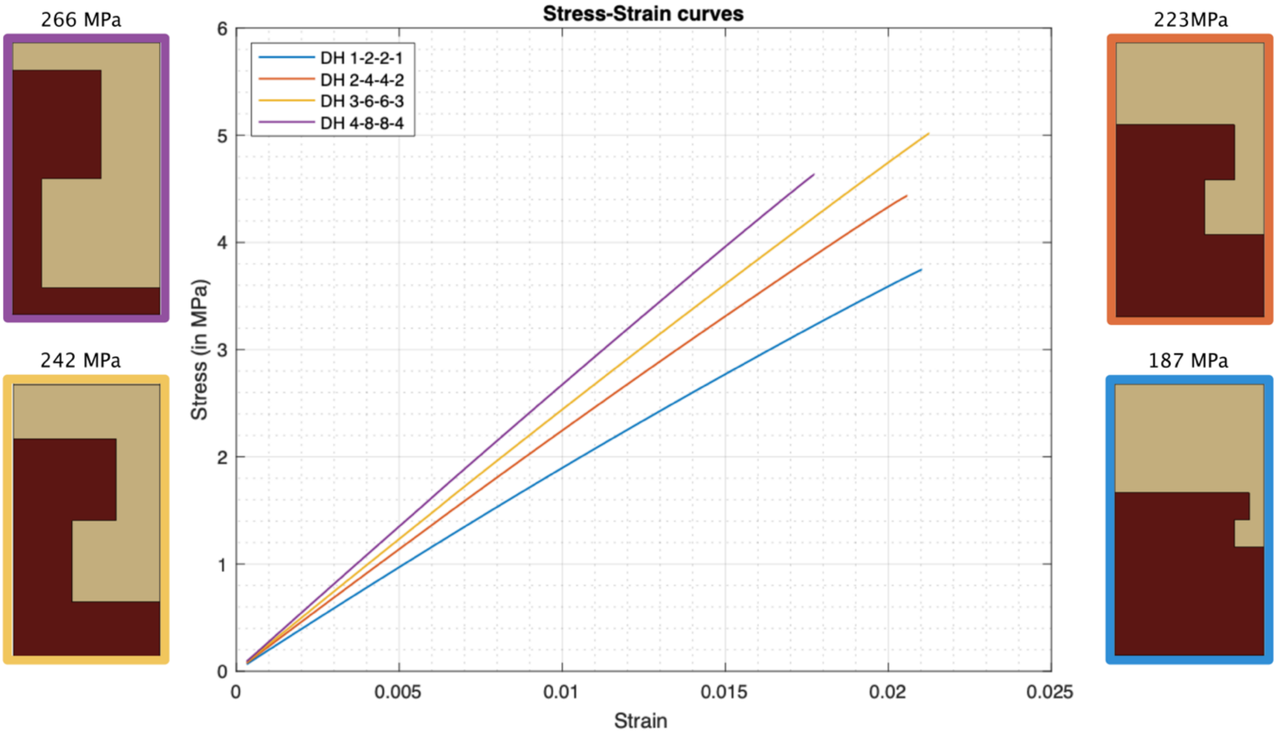


Figure 12. Stress-strain curves of four different sizes of double hook designs, including illustrations and stiffness values.

The second set of simulations examines the A:C ratio of the models, *i.e.*, the ratio between the opening width of the interlocking piece and the width of the hooking mechanism on the bottom (Figure 13A). For these simulations, parameter A was changed while keeping parameters B, C and D fixed. Parameter A was examined from 3 mm to 6 mm with steps of 1 mm, while the remaining parameters were fixed at 8, 8, and 4 mm. These were the best performing values from the previous set of simulations. Figure 13B presents the stress-strain curves for the A:C ratios. There is a clear trend observable between the size of the opening width and the resulting stiffness of the models. Models with larger opening widths, *i.e.*, a thicker neckline for the hooking mechanism, have higher stiffness values than models with lower opening widths. This translates to a higher stiffness for models with a higher A:C ratio. The DH 6-8-8-4 model possesses the highest stiffness from this set of simulations.

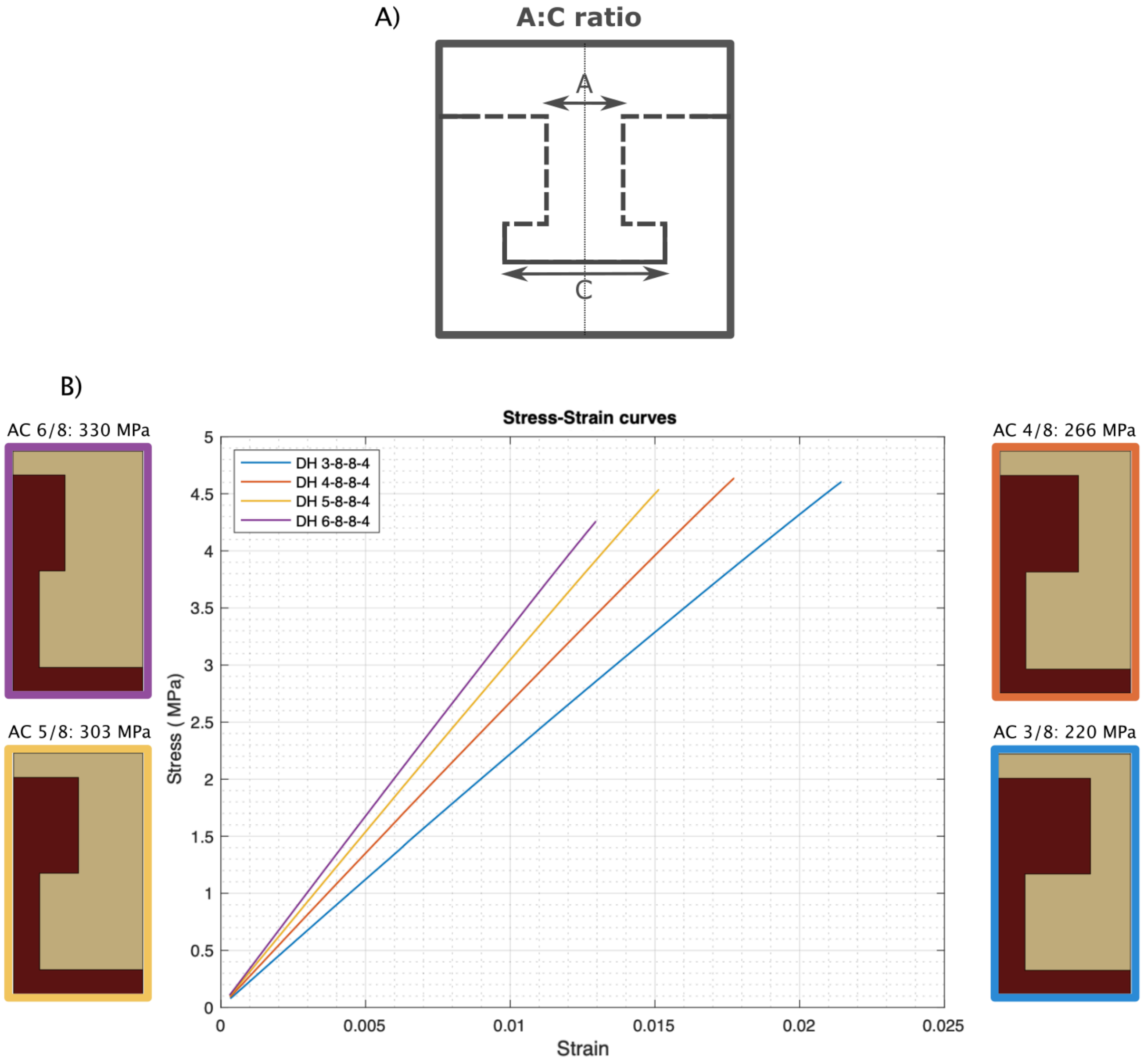


Figure 13. A) Schematic illustration of the A:C ratio. B) Stress-strain curves of four different A:C ratios, including illustrations and stiffness values.

The DH 6-8-8-4 model was used as the starting point for the third set of simulations, examining the B:D ratio. The B:D ratio is the ratio between the depth of the interlocking piece and the thickness of the hooking mechanism (Figure 14A). Parameter B was fixed at 8 mm, which was the maximum value from previous simulations, and parameter D was changed from 1 mm to 7 mm with steps of 2 mm. Parameters A and C were fixed at 6 and 8 mm respectively.

Figure 14B presents the stress-strain curves for the different B:D ratios. There is a clear directional trend observable between the thickness of the hooking mechanism and the stiffness of the models. Models with thicker hooking mechanisms have higher stiffness values than models with thinner hooking mechanisms. This translates to higher stiffness values for models with lower B:D ratios. The DH 6-8-8-7 model possesses the highest stiffness from all sets of simulations, having the thickest hooking mechanism, the thickest neckline and the largest relative size.

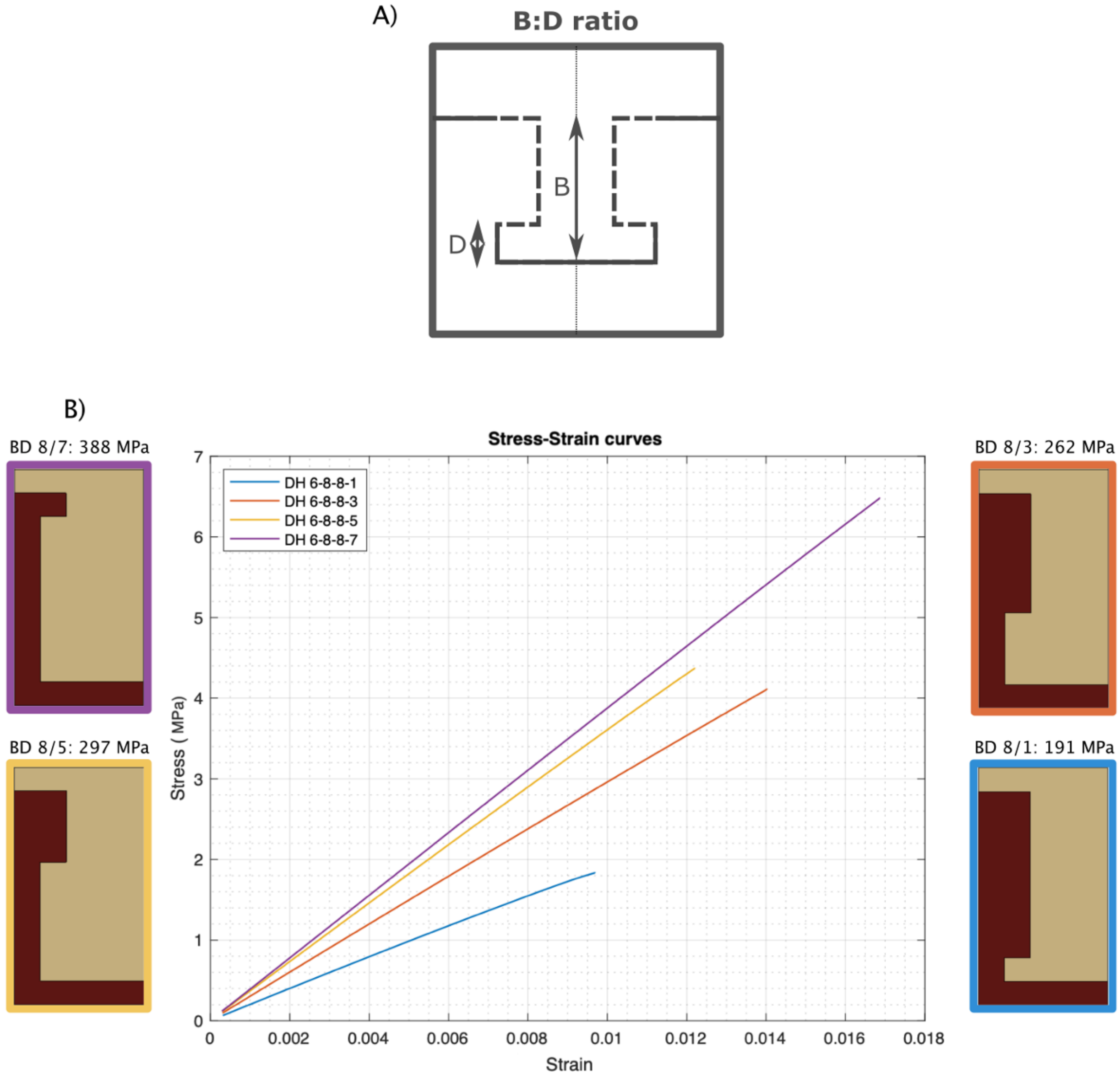


Figure 14. A) Schematic illustration of the B:D ratio. B) Stress-strain curves of four different B:D ratios, including illustrations and stiffness values.

The simulations regarding the relative size of the interlocking piece, the A:C ratio, and B:D ratio all show that larger designs, *i.e.*, more soft material within the unit-cell, result in higher stiffness values than smaller designs. Following the observations for the anti-trapezoidal models, this suggests that higher surface areas of the soft interlocking pieces result in a higher stiffness values for both the interfacial designs. In practice this would mean that hard-soft interfaces, consisting of multiple unit-cells in a row, have higher stiffness values when the distance between the interlocking pieces is small and the pieces are large than when the distance between the pieces is large and the pieces are small. This appears to be a reasonable assumption, because the interfacial stresses are spread out over a larger surface area, resulting in a better performance of the interface.

The (von Mises) stress distributions for the models with the highest stiffness per set of simulations are shown in Figure 15. This shows that the model with the highest stiffness, *i.e.*, DH 6-8-8-7, has relatively smaller hard features and larger soft features, while the models with lower stiffness have relatively larger hard features and smaller soft features, *i.e.*, DH 4-8-8-4 and 6-8-8-4. The illustrations also show an increase in stress concentrations around the hooking mechanism for DH 6-8-8-7 compared to the other models. The upper sections represent the hard part and the lower sections represent the soft part of the models.

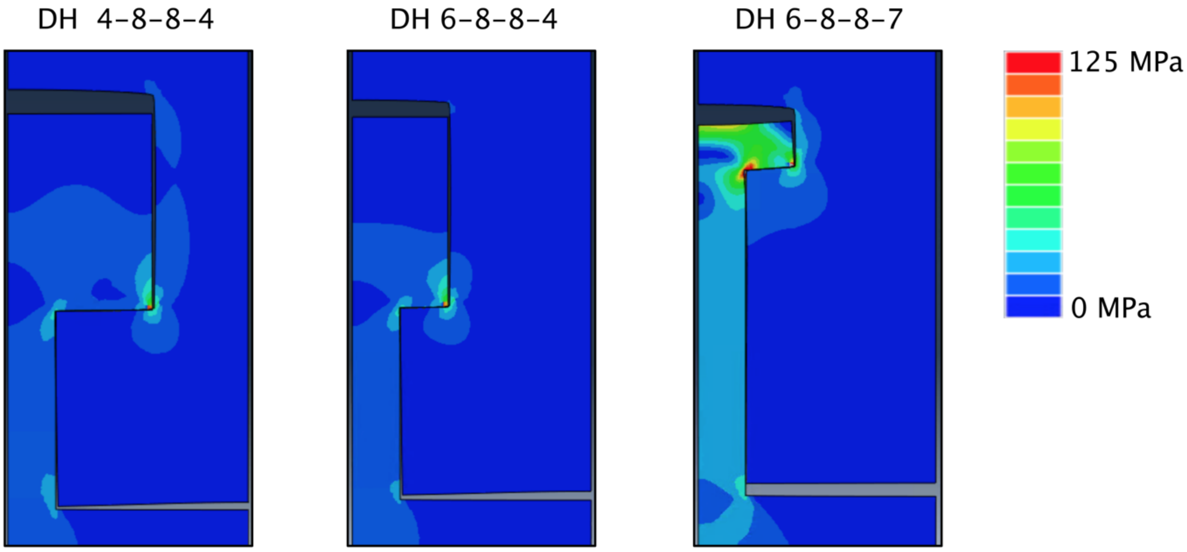


Figure 15. Strain distributions (von Mises) of the designs with the highest stiffness from each set of simulations. All strain distributions were captured at 0.012 strain. The color scale on the right illustrates the stress range.

A scatterplot of the surface area against the stiffness of the models is presented in Figure 16A, confirming the previous observations. Models with a larger surface area of the soft interlocking piece generally have higher stiffness values than models with a lower surface area. Figure 16B shows that the contact area of the resisting edge is not correlated to the stiffness of the models. This is in contrast to the anti-trapezoidal models, for which an increase of the surface area and the size of the resisting edge were both correlated to an increase in stiffness. Unlike the anti-trapezoidal models, the surface area and the size of the resisting edge of the double book models are not correlated to each other (Figure 16C). It is important to note that the set-up of the double hook simulations resulted in fewer data points than the anti-trapezoidal models, decreasing the strength of the correlations.

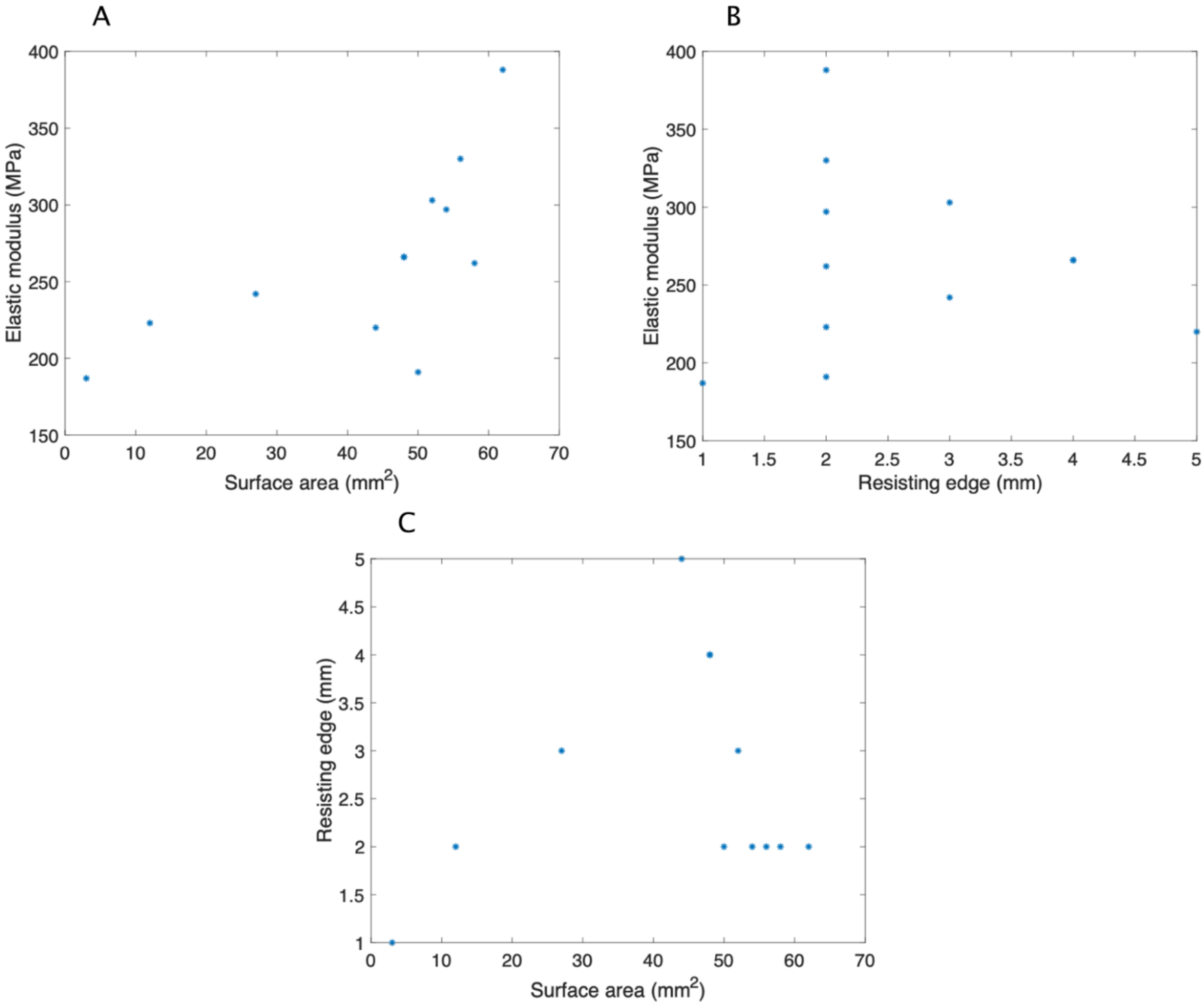


Figure 16. Scatterplots comparing different characteristics of the double hook designs, *i.e.*, a comparison between A) surface area vs. stiffness, B) contact area of the resisting edge vs. stiffness, and C) surface area vs. contact area of the resisting edge.

3.1.3 Validation of half unit-cell model

The half unit-cell model is a symmetric model with two-dimensional periodic boundary conditions. The periodic boundary conditions (PBCs) make the model approximate a large system of unit-cells that are virtually infinite long. The PBCs were used to remove the effects of an edge at the boundary of the unit-cell. Usually, hard-soft interface structures consist of one or more unit-cells without neighboring unit-cells at the outer edges of the structure. This cuts in half the size of the hard features at the outer edge. PBCs make it possible to focus on geometrical interlocking designs without considering edge effects. Therefore, it was necessary to validate these PBCs to investigate if the half unit-cell model represents an extensive system of unit-cells. The model that was created for validation consisted of 10 unit-cells side by side, totaling 100 mm in width. This validation model was compared to the half unit-cell model with PBCs for a randomly selected interface design. The selected design was an anti-trapezoidal design with a 70° angle, 2 mm opening width, and 2 mm depth of the trapezoid. All other properties for these simulations were kept equal. The stress-strain curves for the half unit-cell model and the validation model are shown in Figure 17. The curves and their stiffness are almost identical, with 125,5 MPa for the half unit cell model and 125,7 MPa for the validation model. This shows that the half unit-cell model accurately models an extensive system of unit-cells.

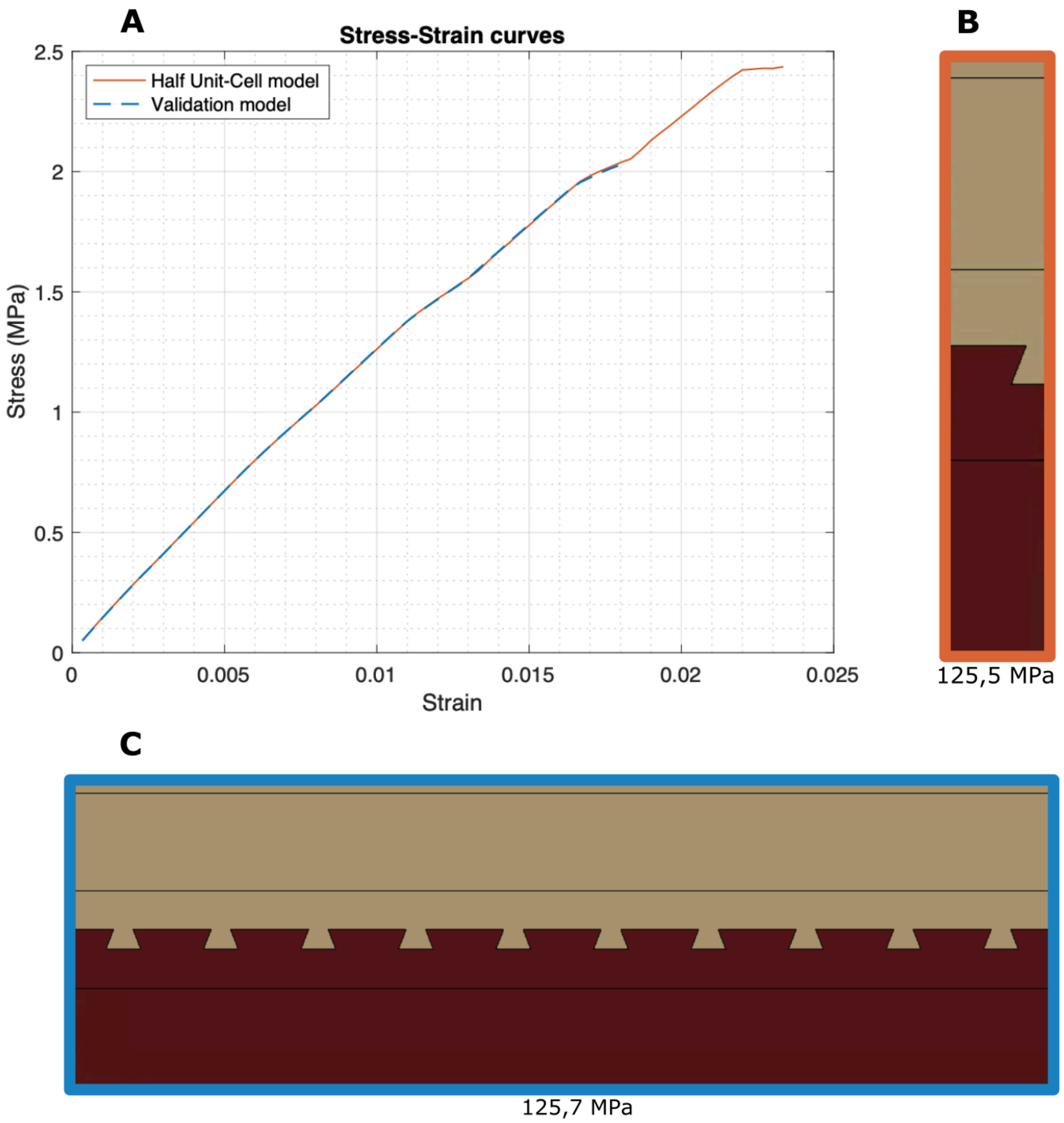


Figure 17. A) Stress-strain curves of the half unit-cell model (with PBCs) and validation model. B) Illustration of the half unit-cell model and, C) validation model with stiffness values.

3.2 Manufacturing of two-dimensional hard-soft interface structures

3.2.1 Tensile data 2D poly-jet samples

The stress-strain curves for the AT 45-3-3, AT 6-4-3, DH 6-8-8-7 and DH 4-8-6-6 models are shown in Figure 18A and B, providing four analyses per model. All curves show a similar trend, by which the stress increases linearly per unit of strain until the simulations reach their yield and ultimately their failure point. The failure point is when the hard and soft phases are entirely separated at either one interface, *i.e.*, the top or bottom interface. The descending phase of the stress-strain curves is different for the anti-trapezoidal and double hook models. The downwards slope of the anti-trapezoidal models is quite sharp, whereas the slope of the double hook models is more gradual. This difference probably arises due to differences in the mechanism of separation between these models. Regarding the anti-trapezoidal models, the stress-strain curves of the AT 60-4-3 model overlie the curves of the AT 45-3-3 model, resulting in a higher stiffness and ultimate tensile strength for the AT 60-4-3 model. For the double hook models, the division is less apparent, showing a more dispersed image of the curves for the two models.

The overall performance of the models is better captured in the plots on Figure 18C-E. Model DH 6-8-8-7 has the highest stiffness, whereby the stiffness of the other models is relatively similar with model AT 45-3-3 scoring slightly lower than the others. The stiffness of the double hook models, in general, is higher than the stiffness of the anti-trapezoidal models. This follows the results of the computational models. Considering the ultimate tensile strength, model DH 4-8-6-6 has the highest strength, followed by the AT 60-4-3 and DH 6-8-8-7 model and finally the AT 45-3-3 model. Taking the double hook and anti-trapezoidal models together shows that the combined double hook models have higher stiffness and ultimate tensile strength than the combined anti-trapezoidal models. On all measures the AT 45-3-3 model performed worst, while the DH 4-8-6-6 model showed a clear inclination towards strength and DH 6-8-8-7 showed an inclination towards stiffness.

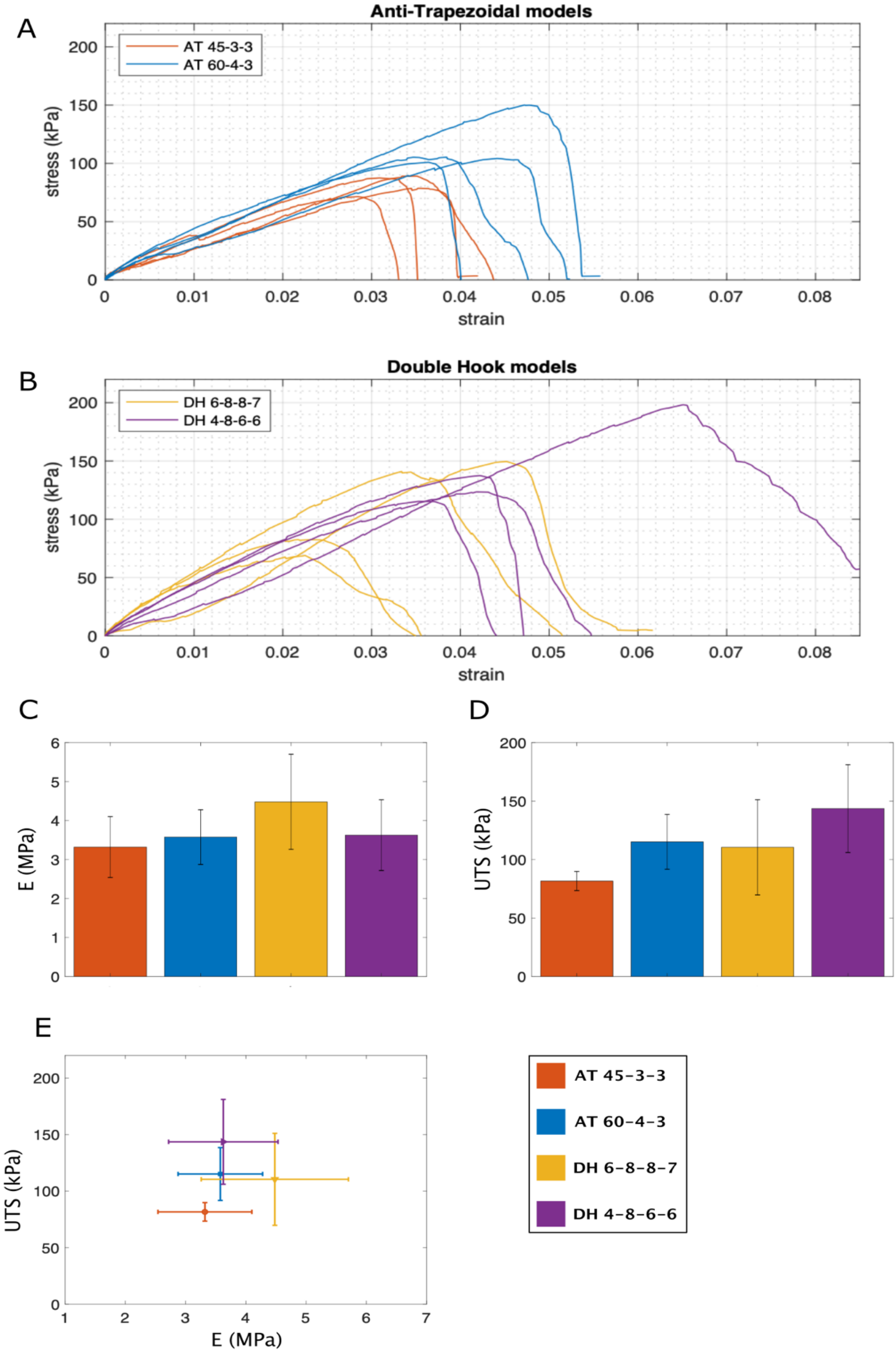


Figure 18. A) Stress-strain curves of the anti-trapezoidal samples, and the B) double hook samples. C) Bar plots showing the mean stiffness per design, and D) the ultimate tensile strength (UTS). E) Plot of mean stiffness vs. UTS. The colors from all illustrations match the same legend.

3.2.2 Digital image correlation

Several tests were performed to examine the strain distributions of the different designs, using samples printed in different batches of poly-jet printing. It is meaningful to use several batches of 3D-printed samples, because each batch may contain small fluctuations in the material properties due to environmental variations, *e.g.*, the ambient temperature and relative humidity. Additionally, the removal of support material from the smaller interlocking features (during post-processing) may also slightly affect the performance of the samples. For this reason, Figure 19 shows the strain distributions of two batches of 3D-printed anti-trapezoidal samples per design, *i.e.*, two evaluations per design, and Figure 20 shows the same for the double hook samples. Some designs, *e.g.*, the DH 6-8-8-7 model in Figures 20C and D, show a significant difference between the samples, illustrating the importance of multiple evaluations. The color schemes on Figures 19 and 20 show the percental strain on the sample's surface, varying from 0% to 15% strain for all evaluations.

Figures 19A and B show two batches of the AT 45-3-3 model with the highest percental strain within the anti-trapezoidal interlocking pieces. The strain pattern is butterfly-shaped, with a lower strain core around the opening of the interlocking pieces and higher strain zones towards the sides of the trapezoid. The higher strain zones are the points where the soft trapezoids come into contact with the hard features. An important observation for this design is the inwards bending of the outer surfaces of the trapezoid. This bending results in a shape shifting motion of the trapezoids that decreases the connection between the phases and releases the trapezoids from its place in the interlocking cavities.

Figures 19C and D present the AT 60-4-3 model, showing a similar strain pattern to the AT 45-3-3 model. There is a lower strain core visible around the opening of the trapezoids with higher strain zones concentrating towards the sides. However, the overall strain appears to be more homogenously distributed throughout the trapezoid, reaching further into the corners of the soft piece than the AT 45-3-3 model. Additionally, the higher strains appear to propagate further into the main body of the soft material. The bending of the trapezoids is still visible for the AT 60-4-3 models, but less pronounced than for the AT 45-3-3 design. This is probably related to the shallower 60° angle compared to the 45° angle, producing a higher degree of interlocking as the pieces bend less out of place.

The double hook models are presented in Figure 20, with the DH 6-8-8-7 model in Figure 20A and B. For this design, the strains concentrate around the start of the thicker part of the interlocking piece, which come into contact with the resisting edges of the hard features. A portion of these strains propagates down the sides of the neckline into the main body of the soft part. The high strain zone is very localized, running horizontally across the interlocking piece, leaving the outer ends of the rectangular parts untouched. The hard and soft phase separate due to a squeezing motion of the soft interlocking pieces, releasing them from their place in the interlocking system.

The strains for the DH 4-8-6-6 model are less localized than for the DH 6-8-8-7 model, spreading out further and more homogenous into the neckline and main body of the soft part, as can be seen in Figure 20C and D. A similar deeper strain pattern was observed for the AT 60-4-3 model as well as a lower strain core around the openings of the interlocking pieces. The hard and soft phase separate according to the same mechanism as the DH 6-8-8-7 model, squeezing itself through the narrower opening of the hard parts. The double hook models appear to show no form of bending, as is the case for the anti-trapezoidal models.

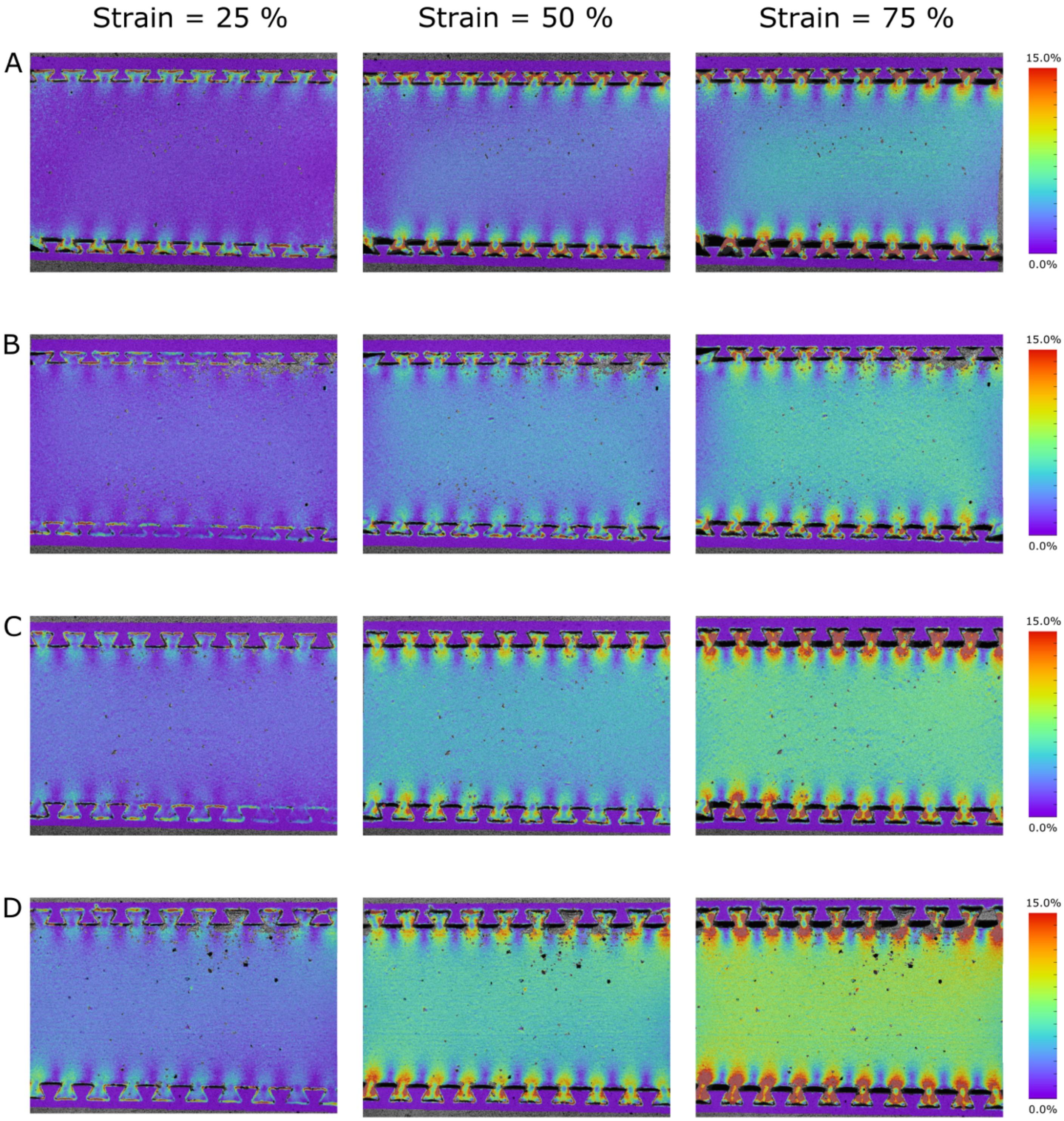


Figure 19. Strain distribution of the true effective strain (von Mises) for anti-trapezoidal models, *i.e.*, AT 45-3-3 and AT 60-4-3. The color scales show the percentage strain. A) Strain distribution for the first batch of AT 45-3-3 model at 25%, 50% and 75% of the total strain applied, and B) second batch. C) First batch of AT 60-4-3 model and D) second batch.

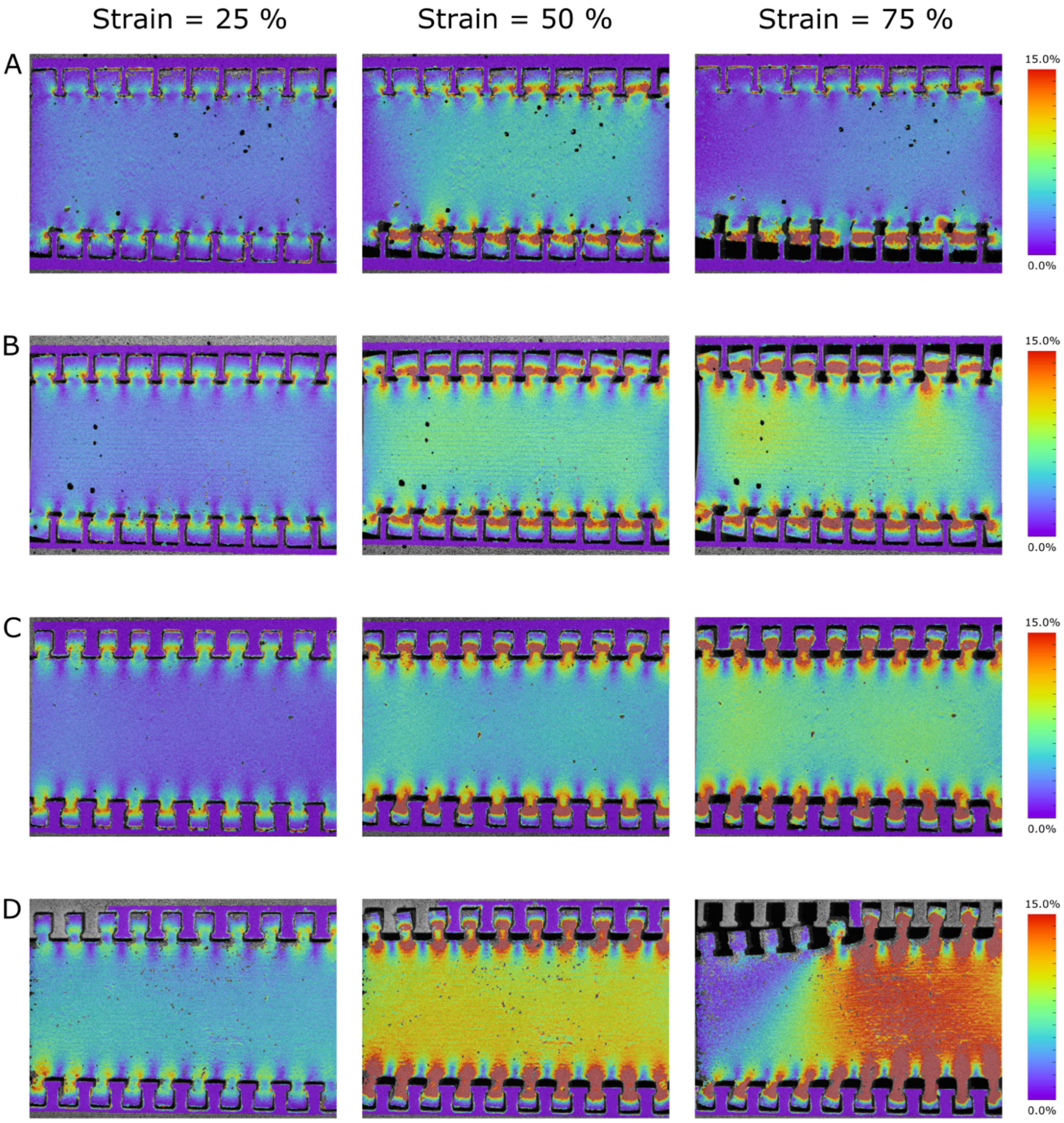


Figure 20. Strain distribution of the true effective strain (von Mises) for double hook models, *i.e.*, DH 6-8-8-7 and DH 4-8-6-6. The color scales show the percentage strain. A) Strain distribution for the first batch of DH 6-8-8-7 model at 25%, 50% and 75% of the total strain applied, and B) second batch. C) First batch of DH 4-8-6-6 model and D) second batch.

3.2.3 Validation of computational models

The computational models were validated by demonstrating similarities between the 2D physical experiments and the simulations. The material properties used for the simulation were that of the poly-jet samples (Table 2.2). The stress-strain curves of the physical experiment and its equivalent simulation were plotted in Figure 21A, together with a comparison of the (von Mises) strain distribution in Figure 21B. The examined model was the AT 45-3-3 model. The strain distribution of the physical experiment was captured at 25% of the total strain applied and at the final step of the simulation. The point of aborting the simulation coincided with the 25% mark of the physical experiment, making this a valid comparison. The resulting stress-strain curves of the physical experiment and the simulation appear to be relatively similar for the duration of the simulation. The same holds true for the strain distributions, revealing similar patterns for both experiments. Both strain patterns have a low strain core around the opening of the interlocking piece with a high strain zone around it. This high strain zone is butterfly-shaped with the highest strains concentrations around the upper point of contact between the hard and soft phase. The strains propagate down into the trapezoidal interlocking piece and upwards into the main body of the soft part in a semi-circular fashion. This leaves the outer corners of the soft trapezoidal pieces relatively unstrained. Figure 21 C and D present the results of the normalized stiffness values of the four physical experiments plotted against their equivalent simulations. The stiffness was normalized by dividing the elastic moduli by the first value of the four, resulting in a value of 1 for the first physical experiment and its simulation. The plots show that these values are highly correlated, showing a similar trend for the stiffness of the models. The only model that deviates from this trend is the AT 60-4-3 model, showing a slightly higher stiffness for the physical experiment and a lower stiffness for the simulation. Altogether, these results show that the simulations are relatively adequate predictors of the physical experiments, validating the computational models.

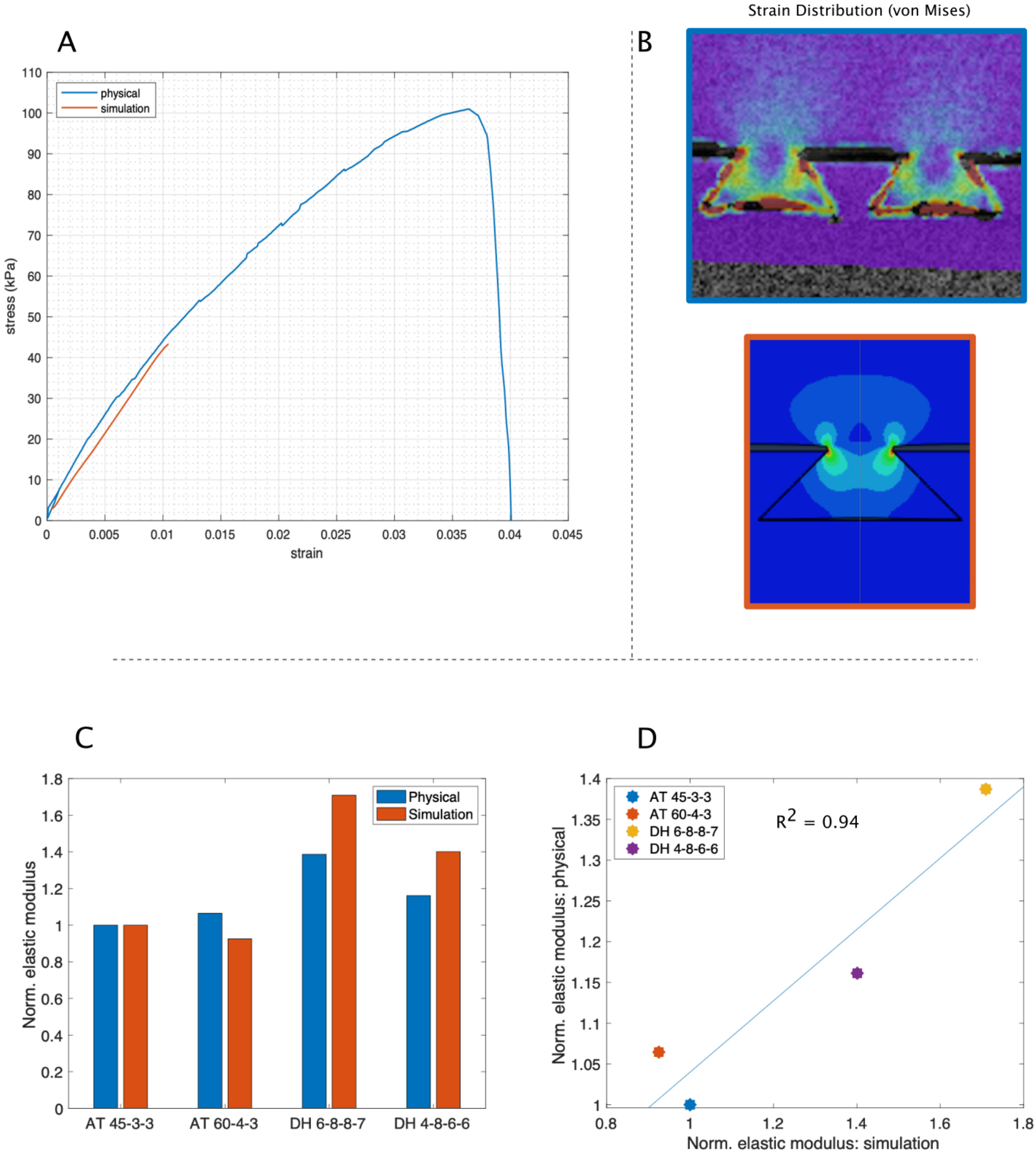


Figure 21. A) Stress-strain curves of the physical experiment and simulation for the AT 45-3-3 model, with B) comparison of the strain distributions. Blue is physical and orange is simulation. C) Comparison of normalized stiffness values of the four physical models and their equivalent simulations, and a D) scatterplot showing the normalized values with a linear regression line.

3.3 Manufacturing of three-dimensional hard-soft interface structures

3.3.1 Interfacial digital microscopy

To study the inflow of hydrogel into the interlocking cavities, the hard-soft interface structure was cut open. This was done with a water-jet cutter piercing the structure through the middle with a 0,1-micron precision. The hydrogel had to be handled delicately, as the weak spots of the soft material were the small pieces of hydrogel connecting the main scaffold to the interlocking structure. When the water-jet entered the structure to aggressively, these small pieces of hydrogel around the openings of the interface tore, ripping off the soft part. The samples that stayed intact were used for evaluation.

Two samples were examined with a digital microscope, ranging the magnification from 20x to 100x (Figure 22). The black part of the cross-sectional images shows the PLA structure (printed with the Ultimaker) and the white part shows the alginate hydrogel (printed with the GeSiM). The cross-sectional images on Figure 22C and H show examples of where the interlocking cavities are satisfactorily filled with hydrogel and well connected to the main body. However, this is not the case for all cavities as some parts of the interlocking structure are filled to a lesser extent with hydrogel, as can be seen in Figure 22D-E and 22I-J.

There are at least two factors to consider when examining the potential tensile performance of these three-dimensional hard-soft interface structures. The first factor is the total surface area connecting the soft interlocking pieces to the main body of the soft part. This area is presented by the surface area of the hydrogel at all the openings of the interlocking cavities, characterized by parameter C for the anti-trapezoidal design. Figure 22D, E and J show examples of where the soft interlocking pieces do not fully attach to the first layer of the main body, decreasing the connecting surface area between the layers. Since this is already a weak spot for the entire structure, it is likely that it will decrease the tensile performance of the structures. The second factor is the total volume of the combined interlocking pieces, which resists tensile forces when a stress is placed upon them. Figure 22D and I show examples of where the interlocking cavities are not adequately filled with hydrogel, decreasing the total volume of the combined interlocking pieces.

Other problems that can arise during the filling of the interlocking structure are air bubbles. Figure 22D and J show examples of large air bubbles that got trapped within the interlocking cavity, while Figure 22C and H show remnants of trapped air at the bottom of the cavity. It is important to note that the cross-sectional images might not be telling the full story. The water-jet cutting of the structures could lead to structural damage that is difficult to differentiate from filling problems during fabrication. Figure 22I for example, shows an interlocking cavity that appears completely empty, while there are still remnants of hydrogel in the bottom left corner of the cavity. This could suggest that the hydrogel got ripped out during cutting. The same could be said for Figure 22E, where the hydrogel appears to be curled up towards the viewer.

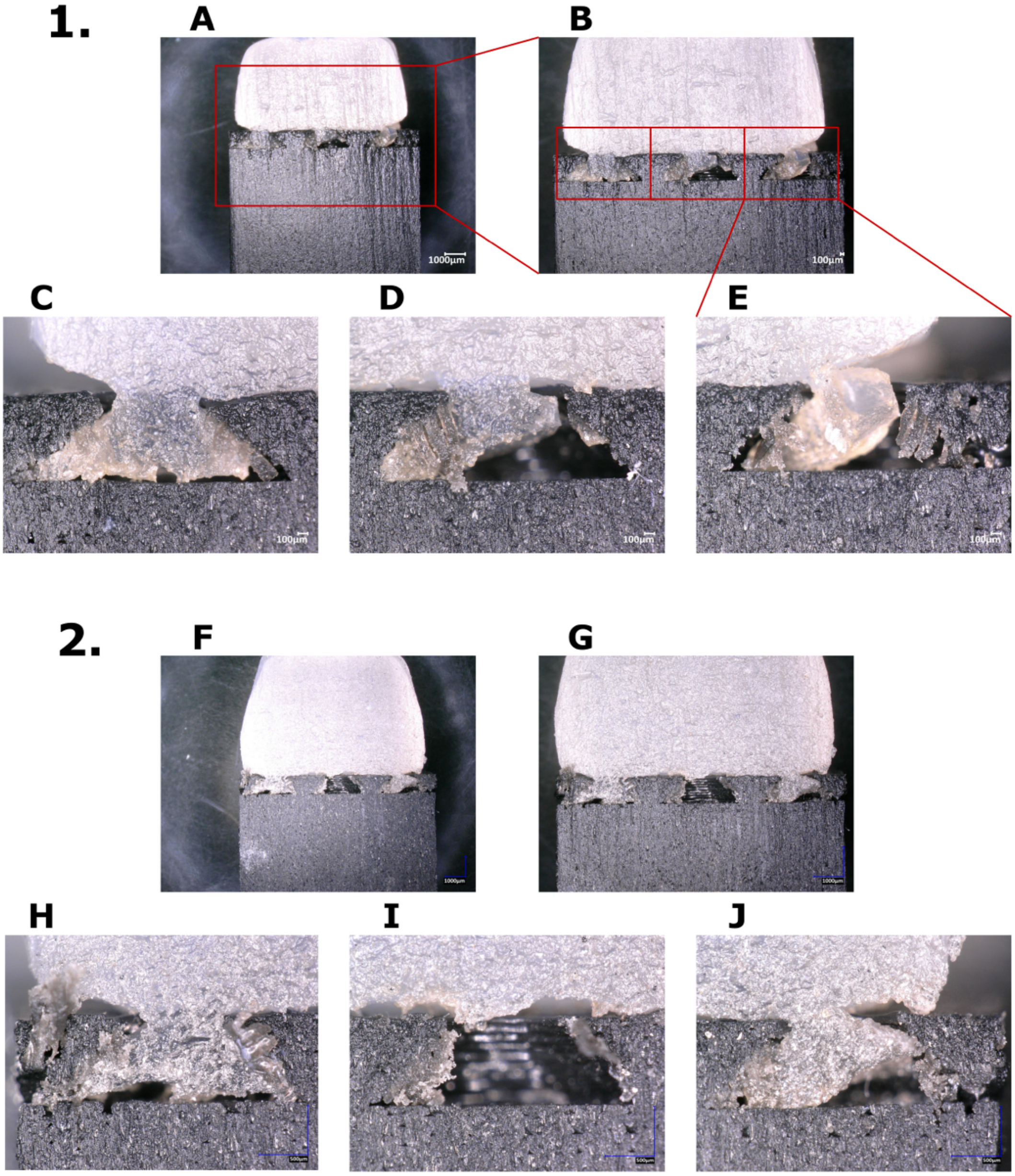


Figure 22. Cross-sectional images of hard-soft interface structures for two samples. A) Sample 1 at 20x magnification, B) 30x magnification, C) left interlocking hole at 100x magnification, D) middle hole (100x) and E) right hole (100x). F) Sample 2 at 20x magnification, G) 30x magnification and H-I-J) left, middle and right hole at 100 x magnification, respectively. Scale bars are unique for each set of images.

3.3.2 Characterization of chemical properties soft part

Fourier-transform infrared spectroscopy (FTIR) was performed to examine the chemistry of the organic compounds of hydrogel, *i.e.*, 2,5% sodium alginate crosslinked with non-organic Ca^{2+}. The spectra of the hydrogel used in this study are shown in Figure 23A, with clear peaks of the absorption bands around 3266 cm^{-1}, 2123 cm^{-1}, 1631 cm^{-1}, 1421 cm^{-1}, 1087 cm^{-1} and 1029 cm^{-1}. The alginate scaffolds for these measurements were prepared using the GeSiM Bioscaffolder with the same bio-ink that was used for the tensile structures.

The stretching vibration around 3266 cm^{-1} shows a strong peak, corresponding to O-H stretching of the carboxylic acid groups in sodium alginate. The weak peak around the 2123 cm^{-1} stretching vibration, could correspond to the presence of alkyne groups in alginate. The medium sharp peak around the 1631 cm^{-1} stretching vibration, could corresponds to either C=C stretching or C=O stretching and the small symmetric peak around the 1421 cm^{-1}, could correspond to O-H bending of carboxylic acid groups. Finally, the small peaks around the 1087 cm^{-1} and 1030 cm^{-1} stretching vibrations, could correspond to C-O stretching. Table 3.1 shows an overview of these peaks including the potential compound class or vibration group.

The current stretching vibrations were compared to the absorption bands from other alginate-related FTIR studies to identify similarities between alginate samples [48, 49]. The study by Saarai et al. (2013) identified five clear peaks in their spectra that are shown in Figure 23B [48]. The most relatable spectrum from this study is shown by the upper graph from Figure 23B, representing an 80:20 material distribution between sodium alginate and gelatin, respectively. The precise content of the sodium alginate in the study by Saarai et al. was very similar to the sodium alginate in this study, being a 2,5% dry weighted sodium alginate concentration crosslinked with calcium chloride (Ca^{2+}). However, the gelatin might have contributed to some differences between the spectra. Overall, the wavenumbers of the peak stretching vibrations differ slightly between the studies, but the overall trend appears to be the same.

The study by Daemi et al. (2012) presents three sodium alginate spectra that appear to be less identical to the FTIR spectra from this study (Figure 23C) [49], but the similarities become more apparent when taking a closer look. The specific spectrum from Daemi et al. that was used for comparison is marked by an "(a)" in the illustration. This spectrum represents a 0,12% sodium alginate concentration crosslinked with calcium chloride (Ca^{2+}). The other two spectra are 0,06% (b) and 0,03% (c) sodium alginate concentrations, respectively. The biggest difference between the spectra from this study and Daemi et al. is the level of noise, making it more difficult to identify relevant peaks. Daemi et al. identified a small peak around 2200 cm^{-1} that is observed in this study, but not in the study by Saarai et al. Nevertheless, most of the peaks are similar to the current study and the study by Saarai et al. (2013). The resemblance between the absorption bands is clear (Table 3.1), confirming the material composition of sodium alginate after the printing process. This will be important when introducing gradients into the structures in future research.

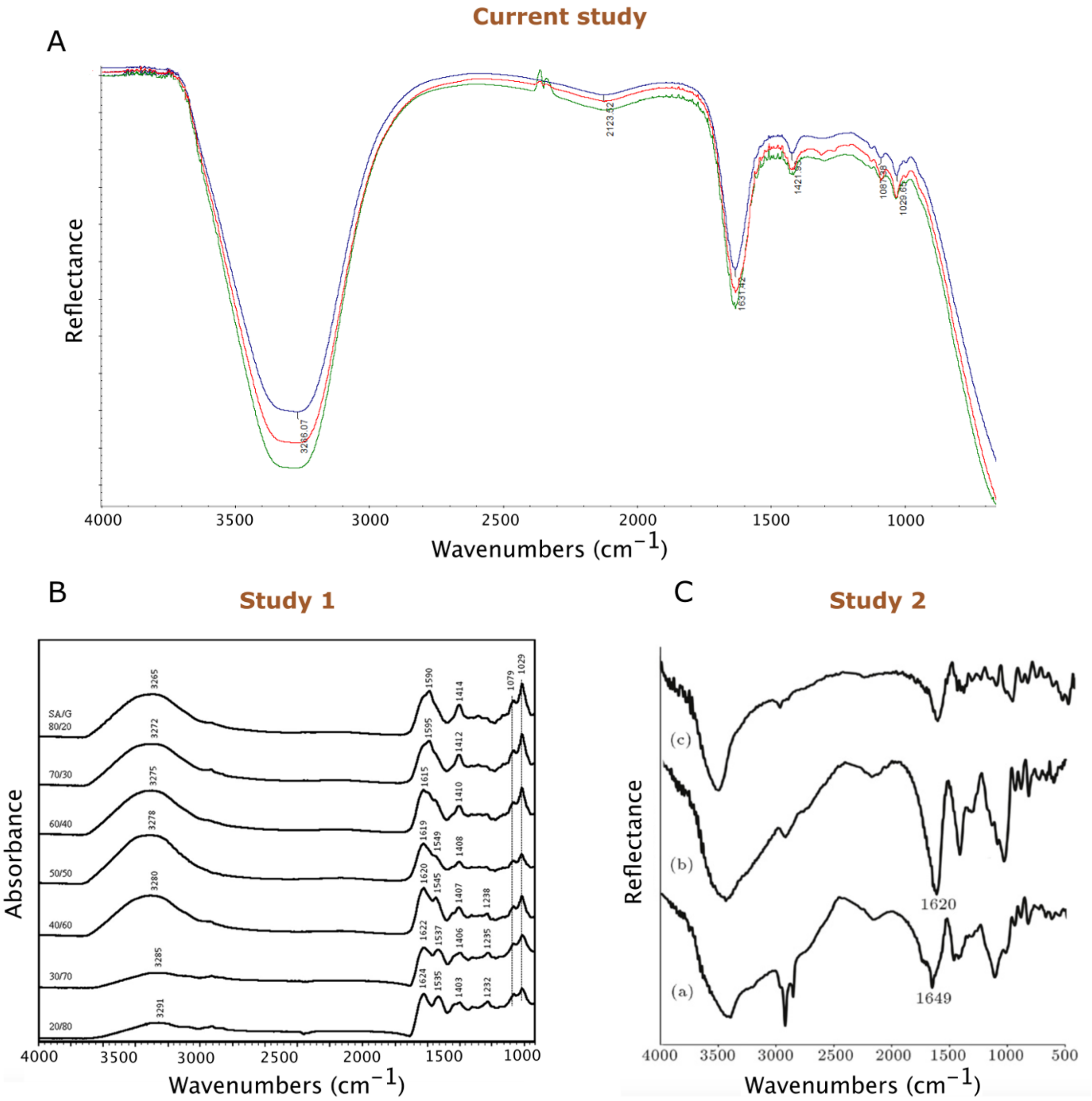


Figure 23. A) Spectra for three different samples of sodium alginate from the current study (2,5% dry weight concentration (w/v), crosslinked with Ca^{2+}). B) Sodium alginate/gelatin spectra from Saarai et al. (2013) [48]. C) Sodium alginate spectra from Daemi et al. (2012) [49].

Table 3.1. Wavenumbers of main stretching vibrations (in cm^{-1}) and related compound class or vibration group.

MAIN PEAKS	CURRENT STUDY	SAARAI ET AL. (2013)	DAEMI ET AL. (2012)	REMARKS
1	3266	3265	2900-3400	carboxylic acid
2	2123	-	2200	alkyne
3	1631	1590	1649	C=C or C=O stretching
4	1421	1414	1400	carboxylic acid
5	1087	1079	1100	C-O stretching
6	1029	1029	1020	C-O stretching

4. Discussion

4.1 Computational modeling

Comparing the results of the anti-trapezoidal simulations to the double hook simulations, it becomes evident that the double hook models in general reach higher stiffness values than the anti-trapezoidal models. The highest reported stiffness for the anti-trapezoidal models was 230 MPa and 388 MPa for the double hook models. For the stiffness range, it was 29-230 MPa for the anti-trapezoidal models and 187-388 MPa for the double hook models. The simulations showed that both types of models favored bigger designs filling a larger part of the unit-cell than smaller designs. The hard-soft ratio appears to be a large determinant of the stiffness.

This was a relatively surprising finding, especially for the double hook models, as it was expected that the size of the resisting edge would play a bigger role in the stiffness than the simulations have shown. Figure 24 shows an example, where model DH 3-8-8-4 has a larger resisting edge than model DH 6-8-8-7, but a lower stiffness. The hard-soft ratio of these models is defined by equation (4.1):

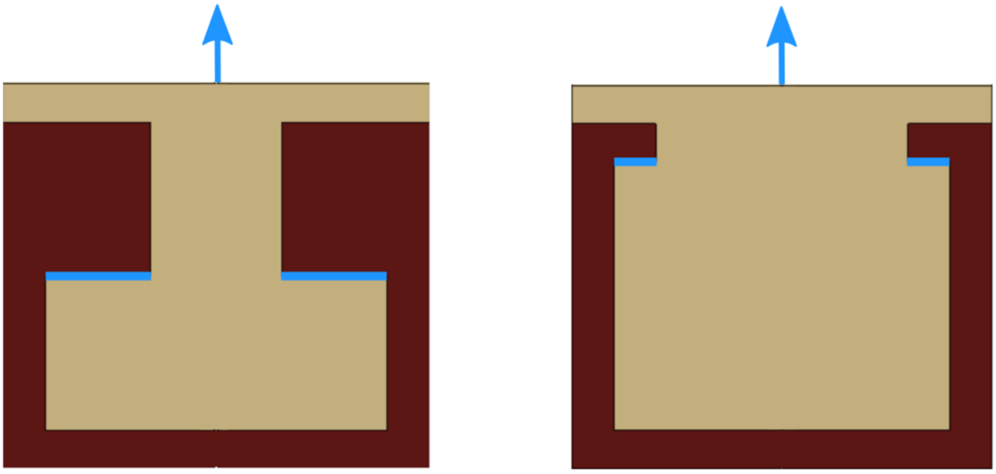
$$k = \frac{S}{H} \tag{4.1}$$

Hereby, S is the total surface area of soft material within the unit-cell, H is the total surface area of hard material within the unit-cell and k is a non-zero proportionality constant. A higher hard-soft constant k represents a larger soft surface divided by a smaller hard surface.

The example in Figure 24 shows that a higher hard-soft ratio and smaller resisting edge, *i.e.*, model DH 6-8-8-7, results in higher stiffness, while a lower hard-soft ratio and larger resisting edge, *i.e.*, model DH 3-8-8-4, results in lower stiffness. This illustrates that for the double hook models an increase of the surface area of the (soft) interlocking piece is more important than the size of the resisting edge. The surface area increases through a higher value for either one of the parameters, *i.e.*, the opening width, depth, width of the hooking mechanism or thickness of the hooking mechanism, but preferably by all four. Interestingly, a larger opening width of the double hook model is directly related to a decrease in size of the resisting edge. This explains the lack of correlation between the surface area and the size of the resisting edge of these models. A larger surface area is the largest predictor of a higher stiffness for the double hook models.

For the anti-trapezoidal models, the size of the resisting edge is highly correlated to the size of the surface area of the interlocking piece. Meaning that a higher hard-soft ratio, *i.e.*, a larger surface area of the soft piece, leads to a larger resisting edge for a trapezoid of the same angle. Higher stiffness values can be attributed to either one of these properties. Another factor that plays a meaningful role in the stiffness of the anti-trapezoidal models is the angle of the trapezoid. The highest stiffness was observed for the intermediate 45° and 60° angles, which showed relatively similar results. Combining a larger surface area with one of these angles results in the highest stiffness for the anti-trapezoidal models.

The absence of fracture properties is an important limitation of the computational models. Especially, designs with a low opening width combined with a large interlocking piece are at risk of rupture of the soft material at the openings of the interface. The same could be said about designs for which the soft interlocking piece becomes so large that the hard material features become tiny. This could lead to problems with premature failure of the interface, even after favorable simulation results. Another limitation was the absence of friction between the phases. In this study, this was done to focus on the geometrical properties of the interface designs without worrying about the unknown frictional properties of PLA and alginate. Further research is necessary to examine realistic friction properties for these materials.



Model	DH 3-8-8-4
Surface area: Soft	46 mm^2
Surface area: Hard	54 mm^2
HS-ratio	≈ 1:1
Resisting edge	5 mm
Stiffness	220 MPa

Model	DH 6-8-8-7
Surface area: Soft	72 mm^2
Surface area: Hard	28 mm^2
HS-ratio	≈ 3:1
Resisting edge	2 mm
Stiffness	388 MPa

Figure 24. Illustration of two double hook models and the resisting edge, including tables with specifications. Blue arrows illustrate the force direction and blue edges illustrate the resisting parts. The beige part is the soft material and dark brown part is the hard material.

4.2 Manufacturing of two-dimensional hard-soft interface structures

The selected poly-jet samples were based on computational models with higher stiffness, consisting of relatively large soft interlocking pieces and small hard material features. This resulted in two particular models with overly thin features, *i.e.*, AT 45-3-3 and DH 6-8-8-7, especially at the edges. The thin material features made both these models more fragile, which could have influenced their performance during the tensile tests. For the AT 45-3-3 model, this was noticeable as the hard parts on the outer edge felt vulnerable during the removal of support material and broke off during the tests (Figure 25A). For the DH 6-8-8-7 model, this was noticeable as the thin features on the outer edge of the hard material were bending during the tests (Figure 25B). These observations illustrate the need for including plastic behavior and fracture properties in the computational models.

A second factor that could have influenced the experiments was the removal of support material. The support material was relatively difficult to remove, especially around the delicate features of the interface, which could have led to some support material being left behind. This support material could have led to differences in tightness between the separate pieces and the frictional properties of the interface. It accounts for a variability in between different samples and between different places of the same sample. The result is that each evaluation can lead to slight differences of the interface properties. The variability was counteracted by performing multiple experiments per design.

To further examine the results of the poly-jet samples, the tensile data and DIC images have to be combined. There are two noticeable mechanisms by which the models appear to separate, *i.e.*, squeezing for the double hook models and bending for the anti-trapezoidal models, which seems to be confirmed when the stress-strain curves are examined. The downwards slope of the anti-trapezoidal models drops very steeply, whereas the slope of the double hook models drops more gradually. This is in accordance with the proposed mechanisms, as the trapezoids start to bend and suddenly snap-out of the interlocking cavity losing their resistance all at once, while the rectangular double hook pieces are squeezed through the openings and gradually lose their resistance towards the end of the piece.

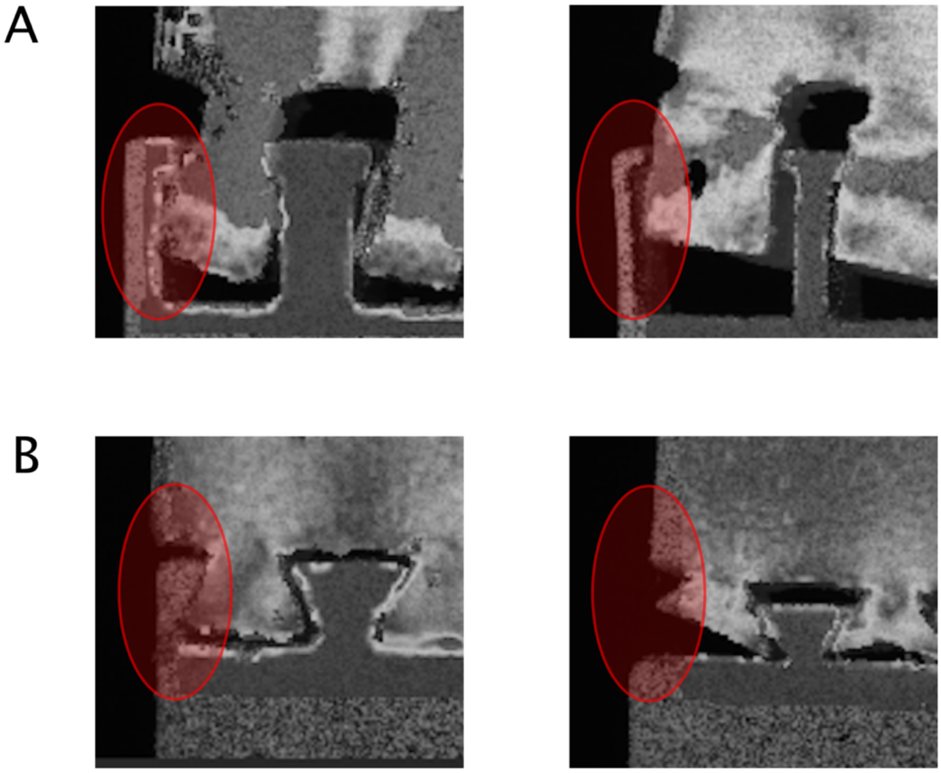


Figure 25. A) Thick feature (DH 4-8-8-6) that stays intact vs. thin feature (DH 6-8-8-7) that bends. B) Thich feature (AT 60-4-3) that stays intact vs. thin feature (AT 45-3-3) that breaks.

The AT 60-4-3 model shows a lower degree of bending and a more homogenous strain distribution than the AT 45-3-3 model (Figure 26A), coinciding with an overall better performance of this design. The performance differences between the individual double hook models are less obvious, showing a specific preference per design.

For the DH 4-8-6-6 model, the size of the edges on the hard parts that resist motion of the soft interlocking piece are larger than for the DH 6-8-8-7 model. This keeps the soft interlocking pieces longer in close proximity to the hard part, resulting in a larger high strain zone that runs down the neckline into the main body of the soft part. The overall strain distribution is less localized and more homogenously spread out (Figure 26B), contributing to a higher ultimate strength for this model. A disadvantage could be the thinner neckline that is more prone to failure.

The DH 6-8-8-7 model on the contrary has a higher stiffness than the DH 4-8-6-6 model. The computational models have shown that this higher stiffness is attributed to a larger interlocking piece, containing more soft material within the unit-cell. A disadvantage is that the hard parts contain smaller resisting edges that allow the interlocking pieces to be more easily squeezed through the openings. This results in a higher initial stiffness, but a lower ultimate tensile strength for this model. A benefit could be that the thicker neckline makes the model less prone to failure, through rupture of the interlocking pieces at their weakest and thinnest point.

In practice, it is conceivable to use these different qualities for specific applications. If the tasks at hand requires the interface to resist short bursts of power, a higher stiffness design might be more convenient. While a higher ultimate strength design might be more convenient when the interface is confronted with lower longer lasting periods of stress. Finally, it is important to mention that the use of adhesives could influence the choice of geometrical interlocking designs. Designs with a thinner neckline are more prone to failure, as the stresses accumulate without the subsequent motion of the interlocking pieces.

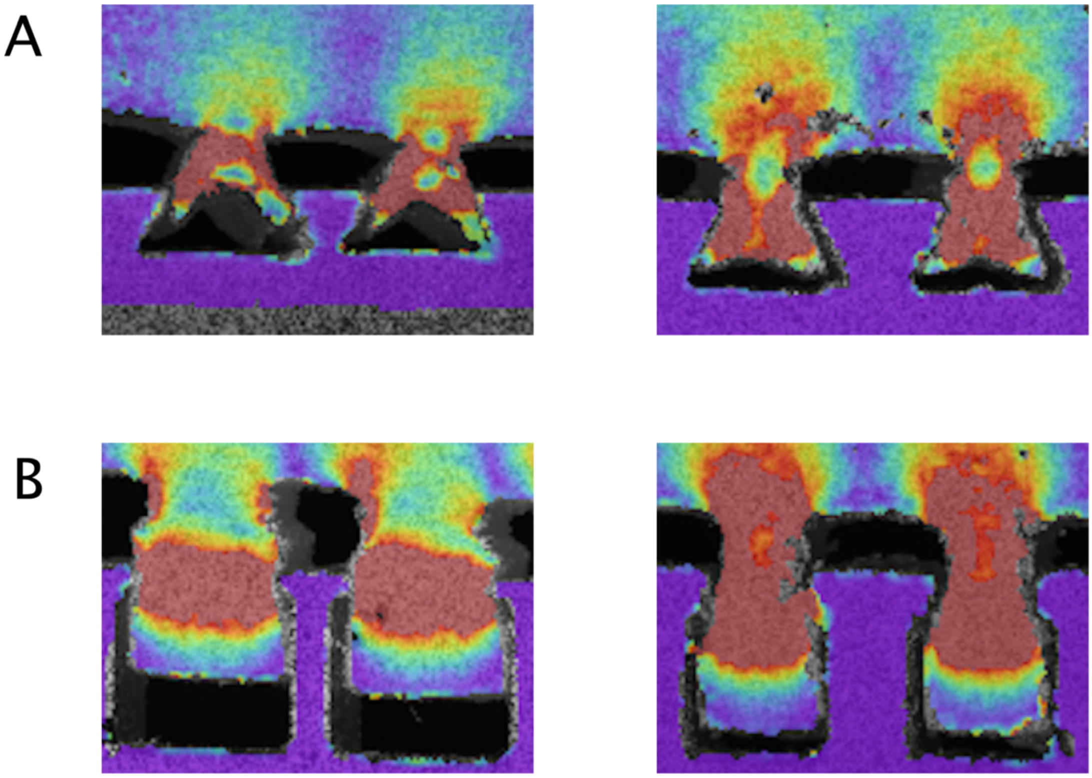


Figure 26. Separation mechanisms per design. A) Bending, AT 45-3-3 vs. AT 60-4-3. B) Squeezing, DH 6-8-8-7 vs. DH 4-8-8-6.

4.3 Manufacturing of three-dimensional hard-soft interface structures

It is crucial to properly fill-up the interlocking system to get the desired performance of the hard-soft interface structures. This was examined using close-ups of the cross-sectional images, as can be seen in Figure 27. The material-related filling kinetics are an important consideration when filling the interlocking structure. The filling kinetics are based on several factors, *i.e.*, filling time, air pressure, viscosity of the bio-ink, nozzle size and placement of the nozzle in relation to the openings. Changes to one of these factors will lead to different dynamics when extruding bio-ink into the interlocking cavity. A less obvious but considerable factor is the placement of the nozzle relative to the openings of the interlocking system. For example, if the nozzle tip is just a little bit smaller than the opening and the nozzle is placed off center instead of exactly on center will lead to different filling kinetics, as the air has a clear path to move. Air that gets trapped within the interlocking system leaves an empty space behind that could decrease the strength and stiffness of the structure (Figure 27A). Including air channels to the structure could be a solution to release the trapped air.

Another factor to consider is the consistency of the bio-ink. The fabrication process of the bio-ink leads to small differences between batches, relating to the viscosity and amount of air bubbles within the ink. A clear procedure helps to avoid these problems, but does not completely solve it. The biggest challenge is to create a pre-crosslinked bio-ink with an appropriate viscosity throughout the entire tube. This pre-crosslinking process requires forceful movement of the solutions leaving behind air bubbles, which are not all removed during centrifuging. Additional substances such as methylcellulose, could help to reform the fabrication process, while maintaining the viscosity of the ink [50-53]. Figure 27B shows an example of an interlocking cavity that is partly filled with an air bubble, decreasing the integrity of the whole structure.

To properly connect the interlocking layer to the first layer of the scaffold it is essential to regulate over- and under extrusion of hydrogel into the interlocking cavities. Over extrusion of hydrogel will lead to problems with the connection of the first layer of strands onto the PLA-scaffold, adding up with each consecutive layer until the scaffold loses its shape. Under extrusion will lead to problems with the connection between the interlocking layer and the first layer of the scaffold, as there is a gap between the materials that (partially) has to be bridged (Figure 27C). The latter poses the biggest risk, as the gap between the materials is hardly filled by gravity pulling the hydrogel down, leading to weak spots of the interface at a critical point. Over extrusion on the contrary can be handled by the printer to a certain extent.

A final problem is the crosslinking kinetics, resulting in substantial shrinkage of the hydrogel. Due to shrinkage of the hydrogel the outer parts of the scaffold buckle in, potentially exposing the outer section of the interlocking structure (Figure 27D). To prevent this the hydrogel scaffold was printed oversized, producing a scaffold of the desired shape after crosslinking.

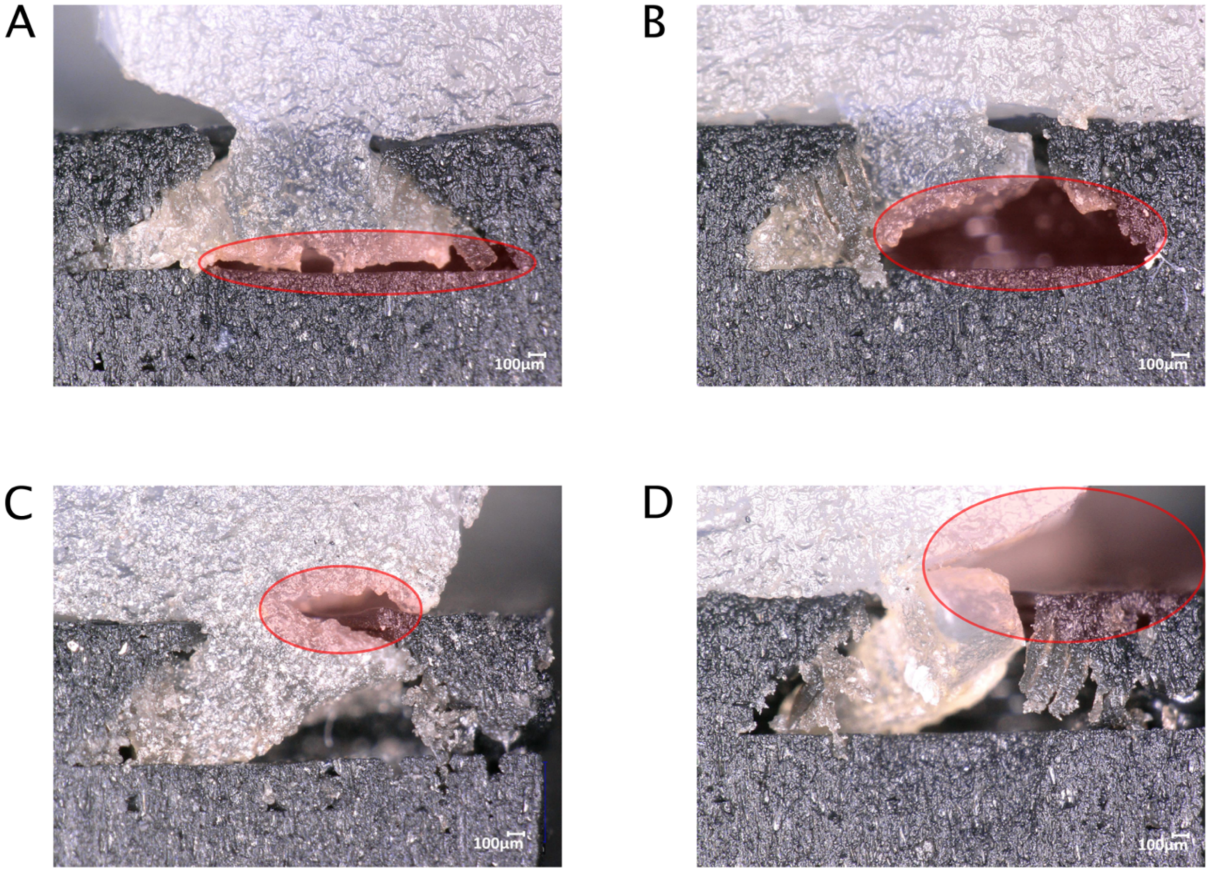


Figure 27. Illustration of different challenges during filling of the interlocking system. A) Small pockets of trapped air, B) air bubbles within the bio-ink, C) under extrusion, D) shrinkage of the hydrogel.

5. Future outlook and conclusion

5.1 Future outlook

The use of hybrid 3D-printing techniques is a novel approach to manufacture hard-soft interfaces. It combines two separate printing techniques with unique characteristics. 3D-printing techniques are accessible, while having the potential to fabricate geometrically complex structures on small length scales. The high spatial resolution of 3D-printers allows for the incorporation of multiple gradients regarding the material composition, orientation, structure and biochemical properties of the interfaces.

For the next phase of this study, it is possible to manufacture the same structures using different materials. Some of these materials could relate better to the underlying tissues, *i.e.*, bone, tendon and ligament. The hydrogel in this study was quite fragile, which made it challenging to test the samples at forces that are relevant to the musculoskeletal system. Additionally, the swelling and uncontrollable nature of hydrogels, *e.g.*, shrinkage after crosslinking, makes it harder to control the final shape of the structures. Biomaterials that are stronger and relate better to the underlying tissues can improve the quality of the study. An interesting combination of biocompatible materials is titanium (higher stiffness) and PCL (lower stiffness). This combination can be manufactured using the same fabrication principles, while mimicking the bone-to-tendon interfaces to a greater degree.

The addition of biomimetic gradients can help to improve the strength and durability of the interfaces [54, 55]. These gradients could bridge the gap in stiffness between the various materials [56]. Other concepts that could be incorporated are toughening mechanisms to improve the interfacial strength [57] and biochemical factors and cells to improve the biological properties of the interfaces [58]. Additive manufacturing techniques are highly suitable to fabricate such complex structures, especially using bioprinters such as the GeSiM Bioscaffolder.

With regards to the geometrical interlocking designs, it could be interesting to examine the use of fractal patterns. Fractal patterns are hierarchical systems of identically repeating geometries, which are often observed in nature to strengthen certain materials [43, 59-62]. The use of adhesives could be another strategy to improve the durability of the interfaces [63].

5.2 Conclusion

In this study two additive manufacturing techniques were combined to create state-of-the-art hard-soft interface structures. The advantage of 3D-printing is its ability to create geometrical and compositional complex structures that can mimic native bone-to-soft tissue interfaces. Here we showed that it is possible to create hybrid structures without using conventional manufacturing techniques, contributing to a promising direction for future research. Additionally, a parametric study was performed to investigate the effects of different geometrical parameters on the stiffness and strain distribution of several geometrical interlocking designs. The knowledge that is obtained on geometrical interlocking designs can contribute to improved interfaces between materials with dissimilar material properties. This study suggests several practical solutions to improve the interfacial performance of biomimetic “anti-trapezoidal” and “double hook” geometries and to manufacture hard-soft interface structures using hybrid 3D-printing techniques.

References

1. Seidi, A., et al., *Gradient biomaterials for soft-to-hard interface tissue engineering.* Acta Biomaterialia, 2011. **7**(4): p. 1441-1451.
2. Di Luca, A., C. Van Blitterswijk, and L. Moroni, *The osteochondral interface as a gradient tissue: From development to the fabrication of gradient scaffolds for regenerative medicine.* Birth Defects Research Part C: Embryo Today: Reviews, 2015. **105**(1): p. 34-52.
3. Apostolakos, J., et al., *The enthesis: a review of the tendon-to-bone insertion.* Muscles, ligaments and tendons journal, 2014. **4**(3): p. 333.
4. Chan, Y.L., A.H. Ngan, and N.M. King, *Nano-scale structure and mechanical properties of the human dentine-enamel junction.* Journal of the mechanical behavior of biomedical materials, 2011. **4**(5): p. 785-95.
5. Liu, Z., et al., *Functional gradients and heterogeneities in biological materials: Design principles, functions, and bioinspired applications.* Progress in Materials Science, 2017. **88**: p. 467-498.
6. Waite, J.H., et al., *Exploring molecular and mechanical gradients in structural bioscaffolds.* Biochemistry, 2004. **43**(24): p. 7653-7662.
7. Rao, R.T., et al., *An overview of recent patents on musculoskeletal interface tissue engineering.* Connect Tissue Res, 2016. **57**(1): p. 53-67.
8. Kjaer, M., *Role of extracellular matrix in adaptation of tendon and skeletal muscle to mechanical loading.* Physiol Rev, 2004. **84**(2): p. 649-98.
9. Boys, A., et al., *Next Generation Tissue Engineering of Orthopedic Soft Tissue-to-Bone Interfaces.* MRS Communications, 2017. **7**: p. 289-308.
10. Armitage, O.E. and M.L. Oyen, *Hard-Soft Tissue Interface Engineering.* Adv Exp Med Biol, 2015. **881**: p. 187-204.
11. Buller, L., et al., *Trends in Anterior Cruciate Ligament Reconstruction in the United States.* Orthopaedic Journal of Sports Medicine, 2015. **3**.
12. Paterno, M.V., *Incidence and Predictors of Second Anterior Cruciate Ligament Injury After Primary Reconstruction and Return to Sport.* Journal of athletic training, 2015. **50**(10): p. 1097-1099.

13. Jain, N., et al., *Epidemiology of musculoskeletal upper extremity ambulatory surgery in the United States.* BMC musculoskeletal disorders, 2014. **15**: p. 4.

14. *Sports-related injuries among high school athletes--United States, 2005-06 school year.* MMWR Morb Mortal Wkly Rep, 2006. **55**(38): p. 1037-40.

15. Joseph, A.M., et al., *A multisport epidemiologic comparison of anterior cruciate ligament injuries in high school athletics.* J Athl Train, 2013. **48**(6): p. 810-7.

16. Hein, J., et al., *Retear Rates After Arthroscopic Single-Row, Double-Row, and Suture Bridge Rotator Cuff Repair at a Minimum of 1 Year of Imaging Follow-up: A Systematic Review.* Arthroscopy: The Journal of Arthroscopic & Related Surgery, 2015. **31**(11): p. 2274-2281.

17. Galatz, L.M., et al., *The Outcome and Repair Integrity of Completely Arthroscopically Repaired Large and Massive Rotator Cuff Tears.* JBJS, 2004. **86**(2).

18. Galatz, L.M., et al., *Characteristics of the rat supraspinatus tendon during tendon-to-bone healing after acute injury.* J Orthop Res, 2006. **24**(3): p. 541-50.

19. Waite, J.H., et al., *Exploring molecular and mechanical gradients in structural bioscaffolds.* Biochemistry, 2004. **43**(24): p. 7653-62.

20. Mirzaali, M.J., et al., *Fracture Behavior of Bio-Inspired Functionally Graded Soft-Hard Composites Made by Multi-Material 3D Printing: The Case of Colinear Cracks.* Materials (Basel), 2019. **12**(17).

21. Nie, X., et al., *Engineering a multiphasic, integrated graft with a biologically developed cartilage–bone interface for osteochondral defect repair.* Journal of Materials Chemistry B, 2019. **7**(42): p. 6515-6525.

22. Qu, D., et al., *Ectopic osteochondral formation of biomimetic porous PVA-n-HA/PA6 bilayered scaffold and BMSCs construct in rabbit.* J Biomed Mater Res B Appl Biomater, 2011. **96**(1): p. 9-15.

23. Li, X., et al., *Nanofiber Scaffolds with Gradations in Mineral Content for Mimicking the Tendon-to-Bone Insertion Site.* Nano Letters, 2009. **9**(7): p. 2763-2768.

24. Reyes, R., et al., *Repair of an osteochondral defect by sustained delivery of BMP-2 or TGFβ1 from a bilayered alginate-PLGA scaffold.* J Tissue Eng Regen Med, 2014. **8**(7): p. 521-33.

25. Xue, D., et al., *Osteochondral repair using porous poly(lactide-co-glycolide)/nano-hydroxyapatite hybrid scaffolds with undifferentiated mesenchymal stem cells in a rat model.* J Biomed Mater Res A, 2010. **94**(1): p. 259-70.

26. Jiang, J., et al., *Bioactive stratified polymer ceramic-hydrogel scaffold for integrative osteochondral repair.* Ann Biomed Eng, 2010. **38**(6): p. 2183-96.

27. Olvera, D., B.N. Sathy, and D.J. Kelly, *Spatial Presentation of Tissue-Specific Extracellular Matrix Components along Electrospun Scaffolds for Tissue Engineering the Bone–Ligament Interface.* ACS Biomaterials Science & Engineering, 2020. **6**(9): p. 5145-5161.

28. Khanarian, N.T., et al., *A functional agarose-hydroxyapatite scaffold for osteochondral interface regeneration.* Biomaterials, 2012. **33**(21): p. 5247-5258.

29. Moffat, K.L., et al., *Novel nanofiber-based scaffold for rotator cuff repair and augmentation.* Tissue Eng Part A, 2009. **15**(1): p. 115-26.

30. Criscenti, G., et al., *Triphasic scaffolds for the regeneration of the bone-ligament interface.* Biofabrication, 2016. **8**(1): p. 015009.

31. Shalumon, K.T., et al., *Microsphere-Based Hierarchically Juxtapositioned Biphasic Scaffolds Prepared from Poly(Lactic-co-Glycolic Acid) and Nanohydroxyapatite for Osteochondral Tissue Engineering.* Polymers, 2016. **8**(12): p. 429.

32. Mente, P.L. and J.L. Lewis, *Elastic modulus of calcified cartilage is an order of magnitude less than that of subchondral bone.* J Orthop Res, 1994. **12**(5): p. 637-47.

33. Beck, E.C., et al., *Approaching the compressive modulus of articular cartilage with a decellularized cartilage-based hydrogel.* Acta Biomater, 2016. **38**: p. 94-105.

34. Gardner-Morse, M.G., et al., *In situ microindentation for determining local subchondral bone compressive modulus.* Journal of biomechanical engineering, 2010. **132**(9): p. 094502-094502.

35. Kolluru, P.V., et al., *Strong and tough mineralized PLGA nanofibers for tendon-to-bone scaffolds.* Acta Biomaterialia, 2013. **9**(12): p. 9442-9450.

36. Lipner, J., et al., *The mechanics of PLGA nanofiber scaffolds with biomimetic gradients in mineral for tendon-to-bone repair.* J Mech Behav Biomed Mater, 2014. **40**: p. 59-68.

37. Augat, P. and S. Schorlemmer, *The role of cortical bone and its microstructure in bone strength.* Age Ageing, 2006. **35 Suppl 2**: p. ii27-ii31.

38. Lai, Y.-S., et al., *The effect of graft strength on knee laxity and graft in-situ forces after posterior cruciate ligament reconstruction.* PloS one, 2015. **10**(5): p. e0127293-e0127293.

39. Cross, L.M., et al., *Nanoengineered biomaterials for repair and regeneration of orthopedic tissue interfaces.* Acta Biomaterialia, 2016. **42**: p. 2-17.

40. Lin, E., et al., *3D printed, bio-inspired prototypes and analytical models for structured suture interfaces with geometrically-tuned deformation and failure behavior.* Journal of the Mechanics and Physics of Solids, 2014. **73**: p. 166-182.

41. Ji, B. and H. Gao, *Mechanical properties of nanostructure of biological materials.* Journal of the Mechanics and Physics of Solids, 2004. **52**(9): p. 1963-1990.

42. Zhang, Y., et al., *Bio-inspired interfacial strengthening strategy through geometrically interlocking designs.* J Mech Behav Biomed Mater, 2012. **15**: p. 70-7.

43. Jaslow, C.R., *Mechanical properties of cranial sutures.* Journal of Biomechanics, 1990. **23**(4): p. 313-321.

44. Cao, N. *Fabrication of alginate hydrogel scaffolds and cell viability in calcium-crosslinked alginate hydrogel*. 2011.

45. Torres, J., et al., *Mechanical Property Optimization of FDM PLA in Shear with Multiple Objectives.* JOM, 2015. **67**(5): p. 1183-1193.

46. Tonarelli, G., et al., *3D-printed biomimetic composite structures of the human cortical bone.* 2020.

47. Diana. *32.1.1 Equivalent Von Mises Strain.* 2012; Available from: https://dianafea.com/manuals/d944/Analys/node405.html.

48. Saarai, A., et al., *On the development and characterisation of crosslinked sodium alginate/gelatine hydrogels.* Journal of the Mechanical Behavior of Biomedical Materials, 2013. **18**: p. 152-166.

49. Daemi, H., M. Barikani, and M. Barmar, *Synthesis and characterization of calcium alginate nanoparticles*. 2012.

50. Gonzalez-Fernandez, T., et al., *Pore-forming bioinks to enable spatio-temporally defined gene delivery in bioprinted tissues.* Journal of Controlled Release, 2019. **301**: p. 13-27.

51. Li, H., et al., *3D Bioprinting of Highly Thixotropic Alginate/Methylcellulose Hydrogel with Strong Interface Bonding.* ACS Applied Materials & Interfaces, 2017. **9**(23): p. 20086-20097.

52. Schütz, K., et al., *Three-dimensional plotting of a cell-laden alginate/methylcellulose blend: towards biofabrication of tissue engineering constructs with clinically relevant dimensions.* J Tissue Eng Regen Med, 2017. **11**(5): p. 1574-1587.

53. Surrao, D.C., et al., *Blended, crosslinked alginate-methylcellulose hydrogels for encapsulation and delivery of olfactory ensheathing cells.* Materialia, 2020. **10**: p. 100654.

54. Calejo, I., et al., *A Textile Platform Using Continuous Aligned and Textured Composite Microfibers to Engineer Tendon-to-Bone Interface Gradient Scaffolds.* Adv Healthc Mater, 2019. **8**(15): p. e1900200.

55. Gadjanski, I., *Recent advances on gradient hydrogels in biomimetic cartilage tissue engineering [version 2; peer review: 2 approved].* F1000Research, 2018. **6**(2158).

56. Sant, S., et al., *Biomimetic gradient hydrogels for tissue engineering.* The Canadian journal of chemical engineering, 2010. **88**: p. 899-911.

57. Fratzl, P., et al., *Hindered Crack Propagation in Materials with Periodically Varying Young's Modulus—Lessons from Biological Materials.* Advanced Materials, 2007. **19**(18): p. 2657-2661.

58. Tarafder, S., et al., *In situ tissue engineering of the tendon-to-bone interface by endogenous stem/progenitor cells.* Biofabrication, 2019. **12**(1): p. 015008.

59. Bosia, F., M. Buehler, and N. Pugno, *Hierarchical simulations for the design of supertough nanofibers inspired by spider silk.* Physical review. E, Statistical, nonlinear, and soft matter physics, 2010. **82**: p. 056103.

60. De Blasio, F.V., *The role of suture complexity in diminishing strain and stress in ammonoid phragmocones.* Lethaia, 2008. **41**(1): p. 15-24.

61. Li, Y., C. Ortiz, and M.C. Boyce, *Bioinspired, mechanical, deterministic fractal model for hierarchical suture joints.* Phys Rev E Stat Nonlin Soft Matter Phys, 2012. **85**(3 Pt 1): p. 031901.

62. Yao, H. and H. Gao, *Mechanics of robust and releasable adhesion in biology: Bottom–up designed hierarchical structures of gecko.* Journal of the Mechanics and Physics of Solids, 2006. **54**(6): p. 1120-1146.

63. Dumont, S., et al., *Soft and hard interface models for bonded elements.* Composites Part B: Engineering, 2018. **153**: p. 480-490.

Supplementary data

Appendix A: Mesh refinement study

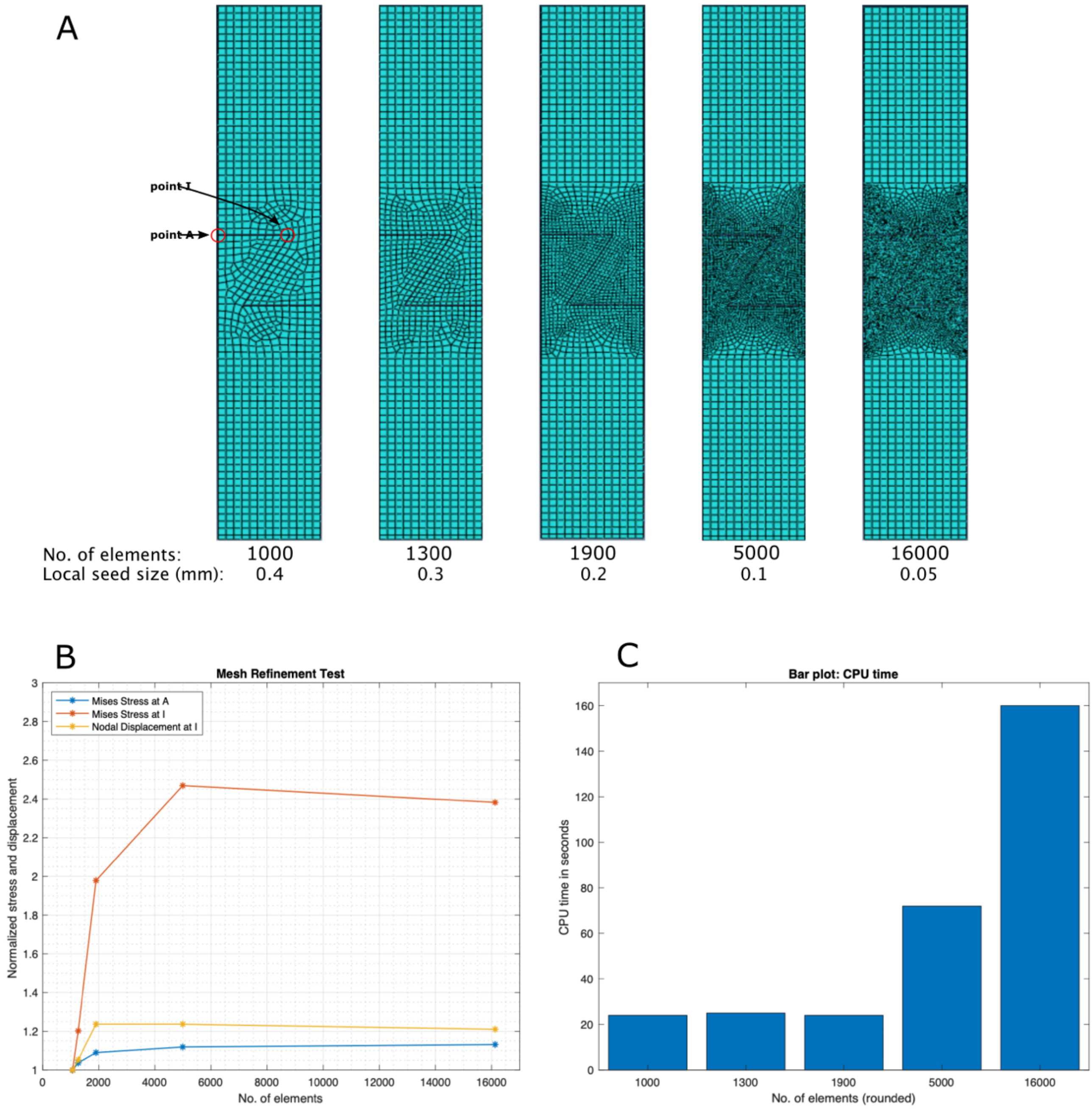


Figure S1. A) Illustration of five mesh models, starting from the coarsest mesh (left: 1000 elements, 0.4 mm local seeds) to the finest mesh (right: 16.000 elements, 0.05 mm local seeds). Left model shows locations of points A and I, which apply for all mesh models. B) Mesh refinement test for three variables, *i.e.*, Mises stresses at point A, Mises stresses at point I and nodal displacement at point I. C) Bar plot numbers of elements against required CPU time per simulation.

Appendix B: Preparation of biomaterial solutions

The formation of a printable bio-ink was accomplished by mixing calcium chloride with alginate. The exact volume and concentration of calcium chloride solution was calculated using the following equation (S.1).

$$M1 \times V1 = M2 \times Vtot \tag{S.1}$$

Hereby, $M1$ is the initial molarity of calcium chloride solution, $V1$ is the initial volume of calcium chloride solution, $M2$ is the final molarity of desired pre-crosslinked solvent, $Vtot$ is the initial volume plus added volume of hydrogel solvent

Which results in equation (S.2) after rearranging the terms.

$$M1 \times V1 = M2\ (V2 + V1)$$

$$V1 = \frac{M2 \times V2}{M1 - M2} \tag{S.2}$$

Hereby, $V2$ is the initial volume of solvent. A 1000 mM calcium chloride solution ($M1$) was pipetted and mixed with alginate. The desired calcium chloride molarity for the pre-crosslinked sodium alginate hydrogel was 20 mM ($M2$). The volume of alginate used per batch was 50 mL ($V2$).

The calcium chloride solution was prepared using the following equation (S.3).

$$m = c \times V \times Mw \tag{S.3}$$

Hereby, m is the mass of calcium chloride in grams, c is the required concentration of calcium chloride in mol per liter, V is the volume of demi water in liter and Mw is the molecular weight of calcium chloride in grams per mol. The molecular weight of calcium chloride is 110,98 g/mol.

Appendix C: Result tables

Table S.1. Stiffness (MPa) computational models for AT 90° angles.

90		Depth
		1 - 5
Width	1 - 5	0

Table S.2. Stiffness (MPa) computational models for AT 75° angles.

75		Depth				
		1	2	3	4	5
Width	1	64	97	113	121	127
	2	48	86	108	122	131
	3	37	74	99	115	126
	4	32	64	90	107	121
	5	29	57	83	102	118

Table S.3. Stiffness (MPa) computational models for AT 60° angles.

60		Depth				
		1	2	3	4	5
Width	1	131	160	172	177	179
	2	109	166	189	200	205
	3	90	157	191	210	224
	4	84	145	190	218	230
	5	78	138	187	212	-

Table S.4. Stiffness (MPa) computational models for AT 45° angles.

45		Depth				
		1	2	3	4	5
Width	1	150	176	184	185	-
	2	136	189	211	-	-
	3	123	188	227	-	-
	4	115	186	-	-	-
	5	111	184	-	-	-

Table S.5. Stiffness (MPa) computational models for AT 30° angles.

30		Depth				
		1	2	3	4	5
Width	1	121	144	-	-	-
	2	112	154	-	-	-
	3	105	155	-	-	-
	4	102	-	-	-	-
	5	102	-	-	-	-

Appendix D: Anti-trapezoidal python script

```
#   2D Anti-Trapezoidal model

from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
from odbAccess import*
from regionToolset import *
import regionToolset
import math
import mesh
import random
import numpy as np
import os

##########################################

##     VARIABLES TO FILL IN !!

A = math.radians(60)        # A = ANGLE (DEG -> RAD)
B = 5.0                     # B = DEPTH
C = 3.0                     # C = WIDTH OPENING

##########################################

# CREATE PARAMETERS HALF UNIT-CELL

# calculate from variables above
A2 = math.radians(90) - A
C = C / 2

# fixed size: half square
w = 5.0
h = 10.0

# parameters interlocking structure
l1     = w - C
ldia   = B / math.cos(A2)
lov    = B * math.tan(A2)
l2     = lov + C
surf_hard = (l1-lov) * B
surf_soft = C * B
if surf_hard == surf_soft:
    surf_hard = surf_hard + 0.00000000001    # introducing minor error

# parameters half box
h1 = h/2 + 0.5*B
h2 = h/2 - 0.5*B

##########################################

# MAKE LOWER HARD INTERLOCK

P1 = (0.0,    0.0)
P2 = (0.0,        h1)
P3 = (l1,         h1)
P4 = (w-l2,    h2)
P5 = (w,      h2)
P6 = (w,      0.0)
##
P7 = (w,      -h)
P8 = (0.0,    -h)

# MAKE UPPER SOFT INTERLOCK
```

```
U1 = (0.0,        h)
U2 = P2
U3 = P3
U4 = P4
U5 = P5
U6 = (w,      h)
##
U7 = (w,      2*h)
U8 = (0.0,    2*h)

##########################################

# COORDINATE FOR EXTRA UNIT-CELL

diff = abs(surf_hard - surf_soft)
extra_h = diff / w

U7e = (w,     2*h+extra_h)
U8e = (0.0,        2*h+extra_h)
P7e = (w,     -h-extra_h)
P8e = (0.0,    -h-extra_h)

##########################################

# CREATE PARTS (with extra unit-cell)

if surf_hard > surf_soft:
    # PART 1
    mdb.Model(name='Model-1', modelType=STANDARD_EXPLICIT)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P2, point2=P3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P3, point2=P4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P4, point2=P5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P5, point2=P6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P6, point2=P7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P7, point2=P8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P8, point2=P1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-1', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-1'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    # PART 2 modified (added extra unit-cell)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U2, point2=U3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U3, point2=U4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U4, point2=U5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U5, point2=U6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U6, point2=U7e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U7e, point2=U8e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U8e, point2=U1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-2', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-2'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
else:
    # PART 1 modified (added extra unit-cell)
    mdb.Model(name='Model-1', modelType=STANDARD_EXPLICIT)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P2, point2=P3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P3, point2=P4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P4, point2=P5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P5, point2=P6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P6, point2=P7e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P7e, point2=P8e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P8e, point2=P1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-1', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-1'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    # PART 2
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U2, point2=U3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U3, point2=U4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U4, point2=U5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U5, point2=U6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U6, point2=U7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U7, point2=U8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U8, point2=U1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-2', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-2'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']

##########################################

# CREATE PARTITIONING (with extra unit-cell)

if surf_hard > surf_soft:
    #Part 1
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=25.0, transform=
        mdb.models['Model-1'].parts['Part-1'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-1'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-1'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P6)
    #
    mdb.models['Model-1'].parts['Part-1'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    #Part 2
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=50.0, transform=
        mdb.models['Model-1'].parts['Part-2'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-2'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U6)
    # EXTRA SECOND LINE
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U8, point2=U7)
    #
    mdb.models['Model-1'].parts['Part-2'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
else:
    #Part 1
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=25.0, transform=
        mdb.models['Model-1'].parts['Part-1'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-1'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-1'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P6)
    # EXTRA SECOND LINE
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P8, point2=P7)
    #
    mdb.models['Model-1'].parts['Part-1'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    #Part 2
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=50.0, transform=
        mdb.models['Model-1'].parts['Part-2'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-2'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U6)
    #
    mdb.models['Model-1'].parts['Part-2'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']

##########################################

# MAKE MATERIALS AND ASSIGN MATERIAL SECTIONS

# MAKE MATERIALS
mdb.models['Model-1'].Material(name='Hydrogel')
mdb.models['Model-1'].materials['Hydrogel'].Elastic(table=((190.0, 0.495), ))
mdb.models['Model-1'].Material(name='PLA')
mdb.models['Model-1'].materials['PLA'].Elastic(table=((3500.0, 0.36), ))

# MAKE SECTIONS
mdb.models['Model-1'].HomogeneousSolidSection(material='PLA', name=
    'HardSection', thickness=1.0)
mdb.models['Model-1'].HomogeneousSolidSection(material='Hydrogel', name=
    'SoftSection', thickness=1.0)

# ASSIGN MATERIALS TO SECTIONS
## #3 selects the upper and lower partition, without the middle section
## #7 selects all three partitions

if surf_hard > surf_soft:
    mdb.models['Model-1'].parts['Part-1'].Set(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#3 ]',
        ), ), name='HardPart_Set')
    mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-1'].sets['HardPart_Set'], sectionName=
        'HardSection', thicknessAssignment=FROM_SECTION)
    mdb.models['Model-1'].parts['Part-2'].Set(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#7 ]',
        ), ), name='SoftPart_Set')
    mdb.models['Model-1'].parts['Part-2'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-2'].sets['SoftPart_Set'], sectionName=
        'SoftSection', thicknessAssignment=FROM_SECTION)
else:
    mdb.models['Model-1'].parts['Part-1'].Set(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#7 ]',
        ), ), name='HardPart_Set')
    mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-1'].sets['HardPart_Set'], sectionName=
        'HardSection', thicknessAssignment=FROM_SECTION)
    mdb.models['Model-1'].parts['Part-2'].Set(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#3 ]',
        ), ), name='SoftPart_Set')
    mdb.models['Model-1'].parts['Part-2'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-2'].sets['SoftPart_Set'], sectionName=
        'SoftSection', thicknessAssignment=FROM_SECTION)

##########################################

# ASSEMBLY: MAKE INSTANCE

mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=OFF, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=OFF, name='Part-2-1',
    part=mdb.models['Model-1'].parts['Part-2'])

##########################################

# CREATE INTERACTION AT INTERFACE
```

```
mdb.models['Model-1'].ContactProperty('IntProp-1')
mdb.models['Model-1'].interactionProperties['IntProp-1'].TangentialBehavior(
    dependencies=0, directionality=ISOTROPIC, elasticSlipStiffness=None,
    formulation=PENALTY, fraction=0.005, maximumElasticSlip=FRACTION,
    pressureDependency=OFF, shearStressLimit=None, slipRateDependency=OFF,
    table=((0.05, ), ), temperatureDependency=OFF)

if surf_hard > surf_soft:
    mdb.models['Model-1'].rootAssembly.Surface(name='m_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        ('[#e0 ]', ), ))
    mdb.models['Model-1'].rootAssembly.Surface(name='s_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        ('[#1c0 ]', ), ))
else:
    mdb.models['Model-1'].rootAssembly.Surface(name='m_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        ('[#700 ]', ), ))
    mdb.models['Model-1'].rootAssembly.Surface(name='s_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        ('[#1c ]', ), ))

mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
    clearanceRegion=None, createStepName='Initial', datumAxis=None,
    initialClearance=OMIT, interactionProperty='IntProp-1', master=
    mdb.models['Model-1'].rootAssembly.surfaces['m_Surf-1'], name=
    'Int-1_S2Scontact', slave=
    mdb.models['Model-1'].rootAssembly.surfaces['s_Surf-1'], sliding=FINITE,
    thickness=ON)

##########################################

# CREATE MESH

# Assign global mesh
mdb.models['Model-1'].rootAssembly.seedPartInstance(deviationFactor=0.1,
    minSizeFactor=0.1, regions=(
    mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1']), size=0.4)

if surf_hard > surf_soft:
    ### WORKS FOR UPPER EXTRA CELL
    mdb.models['Model-1'].rootAssembly.seedEdgeBySize(constraint=FINER,
        deviationFactor=0.1, edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        mask=('[#1f0 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        mask=('[#3e0 ]', ), ), minSizeFactor=0.1, size=0.1)
    mdb.models['Model-1'].rootAssembly.setMeshControls(elemShape=QUAD, regions=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), ))
    mdb.models['Model-1'].rootAssembly.setElementType(elemTypes=(ElemType(
        elemCode=CPE4R, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
        hourglassControl=DEFAULT, distortionControl=DEFAULT), ElemType(
        elemCode=CPE3, elemLibrary=STANDARD)), regions=(
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), ), ))
else:
    ### WORKS FOR LOWER EXTRA CELL
    mdb.models['Model-1'].rootAssembly.seedEdgeBySize(constraint=FINER,
        deviationFactor=0.1, edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        mask=('[#f80 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        mask=('[#3e ]', ), ), minSizeFactor=0.1, size=0.1)
    mdb.models['Model-1'].rootAssembly.setMeshControls(elemShape=QUAD, regions=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), ))
    mdb.models['Model-1'].rootAssembly.setElementType(elemTypes=(ElemType(
        elemCode=CPE4R, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
        hourglassControl=DEFAULT, distortionControl=DEFAULT), ElemType(
        elemCode=CPE3, elemLibrary=STANDARD)), regions=(
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), ), ))

mdb.models['Model-1'].rootAssembly.generateMesh(regions=(
    mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1']))

##########################################

# CREATE PERIODIC BOUNDARY CONDITION

##################################################################################
#INPUTS FOR ADDING STEP, SYMMETRIC AND PERIODIC CONDITIONS IN Y DIRECTION
##################################################################################

Instance1 = 'Part-1-1'
Instance2 = 'Part-2-1'

HH = 3*h + extra_h          #model height
WW = 5.0                    #model width

# My origin is at P1, so adjust HH for RefPoints and finding all nodes
# my origin is 10 mm higher than bottom left corner => offset
if surf_hard > surf_soft:
    yTop = 2*h + extra_h
    yBot = -h
    offset = -10.0
else:
    yTop = 2*h
    yBot = -h - extra_h
    offset = -10.0 - extra_h

tol = 0.001
strainperc = 25.0 #strain in percentage you wish to deform

##################################################################################
#CREATE PERIODIC BOUNDARY CONDITION SETS AND REFERENCE POINTS
##################################################################################

# Getting the nodes from the edges of instances
mdb.models['Model-1'].rootAssembly.Set(name='Top',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yTop-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Bottom',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yBot+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Right',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=WW-tol,
yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=WW-tol,
yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Left',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=0.0+tol, yMax=yTop+tol, zMax=0.0+tol),
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=0.0+tol, yMax=yTop+tol, zMax=0.0+tol),],)

# Assign Ref.points
NameRef1='RefPoint-0';
NameRef2='RefPoint-1';
mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name=NameRef1, type=  DEFORMABLE_BODY)
mdb.models['Model-1'].parts[NameRef1].ReferencePoint(point=(WW, HH/2+offset, 0))
mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name=NameRef2, type=  DEFORMABLE_BODY)
mdb.models['Model-1'].parts[NameRef2].ReferencePoint(point=(WW/2, HH+offset, 0))
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=NameRef1,
part=mdb.models['Model-1'].parts[NameRef1])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=NameRef2,
part=mdb.models['Model-1'].parts[NameRef2])
mdb.models['Model-1'].rootAssembly.Set(name=NameRef1, referencePoints=( mdb.models['Model-
1'].rootAssembly.instances[NameRef1].referencePoints[1],))
```

```
mdb.models['Model-1'].rootAssembly.Set(name=NameRef2, referencePoints=( mdb.models['Model-
1'].rootAssembly.instances[NameRef2].referencePoints[1],))

###################################################################################
#CREATE STEP AND HISTORY OUTPUTS
###################################################################################

#mdb.models['Model-1'].StaticStep(name='Step-1', nlgeom=OFF,
previous='Initial',maxNumInc=100000, initialInc=0.1,minInc=1E-015,)#maxInc=0.01,)
#mdb.models['Model-1'].steps['Step-1'].setValues(stabilizationMagnitude=0.0002,
stabilizationMethod=DISSIPATED_ENERGY_FRACTION, continueDampingFactors=False,
adaptiveDampingRatio=0.05)

mdb.models['Model-1'].StaticStep(adaptiveDampingRatio=0.05,
    continueDampingFactors=False, description='', initialInc=0.001, maxNumInc=
    100000, name='Step-1', nlgeom=ON, noStop=ON, previous='Initial',
    stabilizationMagnitude=0.0002, stabilizationMethod=
    DISSIPATED_ENERGY_FRACTION, timeIncrementationMethod=FIXED)


mdb.models['Model-1'].steps['Step-1'].control.setValues(
    allowPropagation=OFF, resetDefaultValues=OFF, timeIncrementation=(4.0,
    10.0, 9.0, 16.0, 10.0, 4.0, 12.0, 10.0, 6.0, 3.0, 50.0))

#OLD values general solution controls
#mdb.models['Model-1'].steps['Step-1'].control.setValues(
#    discontinuous=ON, timeIncrementation=(8.0, 10.0, 9.0, 16.0, 10.0, 4.0, 12.0, 12.0, 6.0,
3.0, 50.0))

# NEW values gen.sol.cntrl Ir 12 & Ia 15
mdb.models['Model-1'].steps['Step-1'].control.setValues(discontinuous=OFF,
    timeIncrementation=(8.0, 12.0, 9.0, 16.0, 10.0, 4.0, 12.0, 15.0, 6.0, 3.0,
    50.0))

regionDef=mdb.models['Model-1'].rootAssembly.sets['RefPoint-1']
#mdb.models['Model-1'].historyOutputRequests['H-Output-1'].setValues(
#    variables=('U1','U2','RF1','RF2'), region=regionDef, sectionPoints=DEFAULT,
rebar=EXCLUDE, frequency=1,)

mdb.models['Model-1'].HistoryOutputRequest(name='H-Output-T', createStepName='Step-1',
variables=('U2','RF2' ),
    region=mdb.models['Model-1'].rootAssembly.sets[NameRef2], sectionPoints=DEFAULT,
rebar=EXCLUDE)
mdb.models['Model-1'].HistoryOutputRequest(name='H-Output-L', createStepName='Step-1',
variables=('U1','RF1' ),
    region=mdb.models['Model-1'].rootAssembly.sets[NameRef1], sectionPoints=DEFAULT,
rebar=EXCLUDE)
del mdb.models['Model-1'].historyOutputRequests['H-Output-1']

###################################################################################
#CREATE  BC  IN Y DIRECTION
###################################################################################

#alpha = HH * strainperc /100.0
alpha = 10
DefMat=[(UNSET,0.0,0.0),(0.0,alpha,0.0), (0.0,0.0,UNSET)]

mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Step-1',
    distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name=
    'BC-REF-1', region=Region(referencePoints=(
    mdb.models['Model-1'].rootAssembly.instances[NameRef1].referencePoints[1],
    )), u1=DefMat[0][0], u2=DefMat[0][1], u3=DefMat[0][2], ur1=UNSET,ur2=UNSET,ur3=UNSET)
mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Step-1',
    distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name=
    'BC-REF-2', region=Region(referencePoints=(
    mdb.models['Model-1'].rootAssembly.instances[NameRef2].referencePoints[1],
    )), u1=DefMat[1][0], u2=DefMat[1][1], u3=DefMat[1][2], ur1=UNSET,ur2=UNSET,ur3=UNSET)

mdb.models['Model-1'].YsymmBC(name='BC-Bottom',
    createStepName='Step-1', region=mdb.models['Model-1'].rootAssembly.sets['Bottom'],
localCsys=None)
#mdb.models['Model-1'].EncastreBC(createStepName='Step-1', localCsys=None, name=
#    'BC-BottomEncastre', region=mdb.models['Model-1'].rootAssembly.sets['Bottom'])
mdb.models['Model-1'].XsymmBC(name='BC-Left',
    createStepName='Step-1', region=mdb.models['Model-1'].rootAssembly.sets['Left'],
localCsys=None)
```

```
###################################################################################
#CREATE EQUATIONS FOR PERIODIC TOP AND RIGHT
###################################################################################

NameSet = ['Top','Right']
comb = ['1','0']
dimm = [2,1]
for turnn in range(0,len(NameSet)):
    nodesAll=mdb.models['Model-1'].rootAssembly.sets[NameSet[turnn]].nodes
    ranNodes=range(0,len(nodesAll))
    repConst=0
    for repnod1 in range(0,len(nodesAll)):
        mdb.models['Model-1'].rootAssembly.Set(name='Node-'+str(repConst)+str(NameSet[turnn]),
nodes=
           mdb.models['Model-1'].rootAssembly.sets[NameSet[turnn]].nodes[repnod1:repnod1+1])
        mdb.models['Model-1'].Equation(name=NameSet[turnn]+str(dimm[turnn])+'-'+str(repConst),
    terms=((1.0,'Node-'+str(repConst)+str(NameSet[turnn]), dimm[turnn]),(-1.0, 'RefPoint-
'+str(comb[turnn]), dimm[turnn])))
        repConst=repConst+1        #Increase integer for naming equation constraint


##########################################

# CREATE JOB

mdb.Job(atTime=None, contactPrint=OFF, description=
    'Half UnitCell displacement SIM', echoPrint=OFF, explicitPrecision=SINGLE,
    getMemoryFromAnalysis=True, historyPrint=OFF, memory=90, memoryUnits=
    PERCENTAGE, model='Model-1', modelPrint=OFF, name='Job-HalfUnitCell'
    , nodalOutputPrecision=SINGLE, queue=None, resultsFormat=ODB, scratch='',
    type=ANALYSIS, userSubroutine='', waitHours=0, waitMinutes=0)

mdb.jobs['Job-HalfUnitCell'].submit(consistencyChecking=OFF)
```

Appendix E: Double hook python script

```
#   TEST FOR A SIMPLE 2D Model-1

from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
from odbAccess import*
from regionToolset import *
import regionToolset
import math
import mesh
import random
import numpy as np
import os

##########################################

##     VARIABLES TO FILL IN !!

# Warnings
# A   !=   B       (!= is not equal to)
# A   <    B
# A   <<   C
# D   <    B

A =     3.0     # WIDTH OPENING AND THICKNESS
B =     6.0     # DEPTH
C =     6.0     # WIDTH DEEP

D =     3.0

##########################################

# CREATE PARAMETERS HALF UNIT-CELL

# calculate from variables above
A_half = A / 2.0
B = B
C_half = C / 2.0

# size half square
w = 5.0
h = 10.0

# parameters interlocking structure
l1= w - A_half
l2= B - D
l3= C_half - A_half
l4= D
l5= C_half
l6= (h - B) / 2.0

# calculate surface around interlock
surf1 = l2 * A_half;
surf2 = l4 * l5;
surf3 = l2 * l1;
surf4 = l4 * (w - l5);
surf_soft = surf1 + surf2;
surf_hard = surf3 + surf4;
if surf_hard == surf_soft:
    surf_hard = surf_hard + 0.00000000001    # introducing minor error

# parameters large scale
h1 = h/2 + 0.5*B
h2 = h/2 - 0.5*B
```

```
h3 = h1 - l2

##########################################

# MAKE LOWER HARD INTERLOCK

P1 = (0.0,    0.0)
P2 = (0.0,        h1)
P3 = (l1,      h1)
P4 = (l1,      h3)
P5 = (w-C_half, h3)
P6 = (w-C_half, h2)
P7 = (w,      h2)
P8 = (w,      0.0)
##
P9 = (w,      -h)
P10= (0.0,    -h)


# MAKE UPPER SOFT INTERLOCK

U1 = (0.0,        h)
U2 = P2
U3 = P3
U4 = P4
U5 = P5
U6 = P6
U7 = P7
U8 = (w,      h)
##
U9 = (w,      2*h)
U10= (0.0,    2*h)

##########################################

# COORD FOR EXTRA UNIT-CELL

diff = abs(surf_hard - surf_soft)
extra_h = diff / w

U9e = (w,     2*h+extra_h)
U10e = (0.0,   2*h+extra_h)
P9e = (w,     -h-extra_h)
P10e = (0.0,   -h-extra_h)

##########################################


##########################################

# CREATE PARTS (with extra unit-cell)

if surf_hard > surf_soft:
    # PART 1
    mdb.Model(name='Model-1', modelType=STANDARD_EXPLICIT)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P2, point2=P3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P3, point2=P4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P4, point2=P5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P5, point2=P6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P6, point2=P7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P7, point2=P8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P8, point2=P9)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P9, point2=P10)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P10, point2=P1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-1', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-1'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    # PART 2 modified (added extra unit-cell)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U2, point2=U3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U3, point2=U4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U4, point2=U5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U5, point2=U6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U6, point2=U7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U7, point2=U8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U8, point2=U9e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U9e, point2=U10e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U10e, point2=U1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-2', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-2'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
else:
    # PART 1 modified (added extra unit-cell)
    mdb.Model(name='Model-1', modelType=STANDARD_EXPLICIT)
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P2, point2=P3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P3, point2=P4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P4, point2=P5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P5, point2=P6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P6, point2=P7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P7, point2=P8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P8, point2=P9e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P9e, point2=P10e)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P10e, point2=P1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-1', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-1'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    # PART 2
    mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=200.0)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U2)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U2, point2=U3)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U3, point2=U4)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U4, point2=U5)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U5, point2=U6)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U6, point2=U7)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U7, point2=U8)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U8, point2=U9)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U9, point2=U10)
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U10, point2=U1)
    mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name='Part-2', type=
        DEFORMABLE_BODY)
    mdb.models['Model-1'].parts['Part-2'].BaseShell(sketch=
        mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']

#########################################

# CREATE PARTITIONING (with extra unit-cell)

if surf_hard > surf_soft:
    #Part 1
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=25.0, transform=
        mdb.models['Model-1'].parts['Part-1'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-1'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-1'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P8)
    #
    mdb.models['Model-1'].parts['Part-1'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    #Part 2
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=50.0, transform=
        mdb.models['Model-1'].parts['Part-2'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-2'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U8)
    # EXTRA SECOND LINE
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U10, point2=U9)
    #
    mdb.models['Model-1'].parts['Part-2'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
else:
    #Part 1
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=25.0, transform=
        mdb.models['Model-1'].parts['Part-1'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-1'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-1'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P1, point2=P8)
    # EXTRA SECOND LINE
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=P10, point2=P9)
    #
    mdb.models['Model-1'].parts['Part-1'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']
    #Part 2
    mdb.models['Model-1'].ConstrainedSketch(gridSpacing=0.55, name='__profile__',
        sheetSize=50.0, transform=
        mdb.models['Model-1'].parts['Part-2'].MakeSketchTransform(
        sketchPlane=mdb.models['Model-1'].parts['Part-2'].faces[0],
        sketchPlaneSide=SIDE1, sketchOrientation=RIGHT, origin=(0.0, 0.0, 0.0)))
    #
    mdb.models['Model-1'].parts['Part-2'].projectReferencesOntoSketch(filter=
        COPLANAR_EDGES, sketch=mdb.models['Model-1'].sketches['__profile__'])
    mdb.models['Model-1'].sketches['__profile__'].Line(point1=U1, point2=U8)
    #
    mdb.models['Model-1'].parts['Part-2'].PartitionFaceBySketch(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#1 ]',
        ), ), sketch=mdb.models['Model-1'].sketches['__profile__'])
    del mdb.models['Model-1'].sketches['__profile__']

##########################################

# MAKE MATERIALS AND ASSIGN MATERIAL SECTIONS

# MAKE MATERIALS
mdb.models['Model-1'].Material(name='Hydrogel')
mdb.models['Model-1'].materials['Hydrogel'].Elastic(table=((190.0, 0.45), ))
#mdb.models['Model-1'].materials['Hydrogel'].MaxpsDamageInitiation(table=((10.0,
#    ), ))
#mdb.models['Model-1'].materials['Hydrogel'].maxpsDamageInitiation.DamageEvolution(
#    table=((0.3, ), ), type=DISPLACEMENT)

mdb.models['Model-1'].Material(name='PLA')
mdb.models['Model-1'].materials['PLA'].Elastic(table=((3500.0, 0.36), ))
#mdb.models['Model-1'].materials['PLA'].MaxpsDamageInitiation(table=((200.0, ),
#    ))
#mdb.models['Model-1'].materials['PLA'].maxpsDamageInitiation.DamageEvolution(
#    table=((0.3, ), ), type=DISPLACEMENT)

# MAKE SECTIONS
mdb.models['Model-1'].HomogeneousSolidSection(material='PLA', name=
    'HardSection', thickness=1.0)
mdb.models['Model-1'].HomogeneousSolidSection(material='Hydrogel', name=
    'SoftSection', thickness=1.0)

# ASSIGN MATERIALS TO SECTIONS
## #3 selects the upper and lower partition, without the middle section
## #7 selects all three partitions

if surf_hard > surf_soft:
    mdb.models['Model-1'].parts['Part-1'].Set(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#3 ]',
        ), ), name='HardPart_Set')
    mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-1'].sets['HardPart_Set'], sectionName=
        'HardSection', thicknessAssignment=FROM_SECTION)
    mdb.models['Model-1'].parts['Part-2'].Set(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#7 ]',
        ), ), name='SoftPart_Set')
    mdb.models['Model-1'].parts['Part-2'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-2'].sets['SoftPart_Set'], sectionName=
        'SoftSection', thicknessAssignment=FROM_SECTION)
else:
    mdb.models['Model-1'].parts['Part-1'].Set(faces=
        mdb.models['Model-1'].parts['Part-1'].faces.getSequenceFromMask(('[#7 ]',
        ), ), name='HardPart_Set')
    mdb.models['Model-1'].parts['Part-1'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-1'].sets['HardPart_Set'], sectionName=
        'HardSection', thicknessAssignment=FROM_SECTION)
    mdb.models['Model-1'].parts['Part-2'].Set(faces=
        mdb.models['Model-1'].parts['Part-2'].faces.getSequenceFromMask(('[#3 ]',
        ), ), name='SoftPart_Set')
    mdb.models['Model-1'].parts['Part-2'].SectionAssignment(offset=0.0,
        offsetField='', offsetType=MIDDLE_SURFACE, region=
        mdb.models['Model-1'].parts['Part-2'].sets['SoftPart_Set'], sectionName=
        'SoftSection', thicknessAssignment=FROM_SECTION)

##########################################

# ASSEMBLY: MAKE INSTANCE

mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=OFF, name='Part-1-1',
    part=mdb.models['Model-1'].parts['Part-1'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=OFF, name='Part-2-1',
    part=mdb.models['Model-1'].parts['Part-2'])

##########################################

# CREATE INTERACTION AT INTERFACE

mdb.models['Model-1'].ContactProperty('IntProp-1')
mdb.models['Model-1'].interactionProperties['IntProp-1'].TangentialBehavior(
    dependencies=0, directionality=ISOTROPIC, elasticSlipStiffness=None,
    formulation=PENALTY, fraction=0.005, maximumElasticSlip=FRACTION,
    pressureDependency=OFF, shearStressLimit=None, slipRateDependency=OFF,
    table=((0.05, ), ), temperatureDependency=OFF)

if surf_hard > surf_soft:
    mdb.models['Model-1'].rootAssembly.Surface(name='m_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        ('[#e0 ]', ), ))
    mdb.models['Model-1'].rootAssembly.Surface(name='s_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        ('[#1c0 ]', ), ))
else:
    mdb.models['Model-1'].rootAssembly.Surface(name='m_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        ('[#700 ]', ), ))
    mdb.models['Model-1'].rootAssembly.Surface(name='s_Surf-1', side1Edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        ('[#1c ]', ), ))

mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
    clearanceRegion=None, createStepName='Initial', datumAxis=None,
    initialClearance=OMIT, interactionProperty='IntProp-1', master=
    mdb.models['Model-1'].rootAssembly.surfaces['m_Surf-1'], name=
    'Int-1_S2Scontact', slave=
    mdb.models['Model-1'].rootAssembly.surfaces['s_Surf-1'], sliding=FINITE,
    thickness=ON)

##########################################

# CREATE MESH
# Assign global mesh
mdb.models['Model-1'].rootAssembly.seedPartInstance(deviationFactor=0.1,
    minSizeFactor=0.1, regions=(
    mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1']), size=0.4)

if surf_hard > surf_soft:
    ### WORKS FOR UPPER EXTRA CELL
    mdb.models['Model-1'].rootAssembly.seedEdgeBySize(constraint=FINER,
        deviationFactor=0.1, edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        mask=('[#7f0 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        mask=('[#fe0 ]', ), ), minSizeFactor=0.1, size=0.1)
    mdb.models['Model-1'].rootAssembly.setMeshControls(elemShape=QUAD, regions=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), ))
    mdb.models['Model-1'].rootAssembly.setElementType(elemTypes=(ElemType(
        elemCode=CPE4R, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
        hourglassControl=DEFAULT, distortionControl=DEFAULT), ElemType(
        elemCode=CPE3, elemLibrary=STANDARD)), regions=(
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), ), ))
else:
    ### WORKS FOR LOWER EXTRA CELL
    mdb.models['Model-1'].rootAssembly.seedEdgeBySize(constraint=FINER,
        deviationFactor=0.1, edges=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].edges.getSequenceFromMask(
        mask=('[#3f80 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].edges.getSequenceFromMask(
        mask=('[#fe ]', ), ), minSizeFactor=0.1, size=0.1)
    mdb.models['Model-1'].rootAssembly.setMeshControls(elemShape=QUAD, regions=
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), ))
    mdb.models['Model-1'].rootAssembly.setElementType(elemTypes=(ElemType(
        elemCode=CPE4R, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
        hourglassControl=DEFAULT, distortionControl=DEFAULT), ElemType(
        elemCode=CPE3, elemLibrary=STANDARD)), regions=(
        mdb.models['Model-1'].rootAssembly.instances['Part-1-1'].faces.getSequenceFromMask(
        mask=('[#7 ]', ), )+\
        mdb.models['Model-1'].rootAssembly.instances['Part-2-1'].faces.getSequenceFromMask(
        mask=('[#3 ]', ), ), ))

mdb.models['Model-1'].rootAssembly.generateMesh(regions=(
    mdb.models['Model-1'].rootAssembly.instances['Part-1-1'],
    mdb.models['Model-1'].rootAssembly.instances['Part-2-1']))

##########################################

# CREATE PERIODIC BOUNDARY CONDITION

##################################################################################
#INPUTS FOR ADDING STEP, SYMMETRIC AND PERIODIC CONDITIONS IN Y DIRECTION
##################################################################################

Instance1 = 'Part-1-1'
Instance2 = 'Part-2-1'

HH = 3*h + extra_h          #model height
WW = 5.0                    #model width

# My origin is at P1, so adjust HH for RefPoints and finding all nodes
# my origin is 10 mm higher than bottom left corner => offset
if surf_hard > surf_soft:
    yTop = 2*h + extra_h
    yBot = -h
    offset = -10.0
else:
    yTop = 2*h
    yBot = -h - extra_h
    offset = -10.0 - extra_h
```

```
tol = 0.001
strainperc = 25.0 #strain in percentage you wish to deform

##################################################################################
#CREATE PERIODIC BOUNDARY CONDITION SETS AND REFERENCE POINTS
##################################################################################

# Getting the nodes from the edges of instances
mdb.models['Model-1'].rootAssembly.Set(name='Top',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yTop-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Bottom',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yBot+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Right',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=WW-tol,
yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=WW-tol,
yMin=yBot-tol, zMin=0.0-tol, xMax=WW+tol, yMax=yTop+tol, zMax=0.0+tol),],)
mdb.models['Model-1'].rootAssembly.Set(name='Left',
    nodes=[
   mdb.models['Model-1'].rootAssembly.instances[Instance1].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=0.0+tol, yMax=yTop+tol, zMax=0.0+tol),
   mdb.models['Model-1'].rootAssembly.instances[Instance2].nodes.getByBoundingBox(xMin=0.0-
tol, yMin=yBot-tol, zMin=0.0-tol, xMax=0.0+tol, yMax=yTop+tol, zMax=0.0+tol),],)

# Assign Ref.points
NameRef1='RefPoint-0';
NameRef2='RefPoint-1';
mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name=NameRef1, type=  DEFORMABLE_BODY)
mdb.models['Model-1'].parts[NameRef1].ReferencePoint(point=(WW, HH/2+offset, 0))
mdb.models['Model-1'].Part(dimensionality=TWO_D_PLANAR, name=NameRef2, type=  DEFORMABLE_BODY)
mdb.models['Model-1'].parts[NameRef2].ReferencePoint(point=(WW/2, HH+offset, 0))
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=NameRef1,
part=mdb.models['Model-1'].parts[NameRef1])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name=NameRef2,
part=mdb.models['Model-1'].parts[NameRef2])
mdb.models['Model-1'].rootAssembly.Set(name=NameRef1, referencePoints=( mdb.models['Model-
1'].rootAssembly.instances[NameRef1].referencePoints[1],))
mdb.models['Model-1'].rootAssembly.Set(name=NameRef2, referencePoints=( mdb.models['Model-
1'].rootAssembly.instances[NameRef2].referencePoints[1],))

##################################################################################
#CREATE STEP AND HISTORY OUTPUTS
##################################################################################

#mdb.models['Model-1'].StaticStep(name='Step-1', nlgeom=OFF,
previous='Initial',maxNumInc=100000, initialInc=0.1,minInc=1E-015,)#maxInc=0.01,)
#mdb.models['Model-1'].steps['Step-1'].setValues(stabilizationMagnitude=0.0002,
stabilizationMethod=DISSIPATED_ENERGY_FRACTION, continueDampingFactors=False,
adaptiveDampingRatio=0.05)

mdb.models['Model-1'].StaticStep(adaptiveDampingRatio=0.05,
    continueDampingFactors=False, description='', initialInc=0.001, maxNumInc=
    100000, name='Step-1', nlgeom=ON, noStop=ON, previous='Initial',
    stabilizationMagnitude=0.0002, stabilizationMethod=
    DISSIPATED_ENERGY_FRACTION, timeIncrementationMethod=FIXED)


mdb.models['Model-1'].steps['Step-1'].control.setValues(
    allowPropagation=OFF, resetDefaultValues=OFF, timeIncrementation=(4.0,
    10.0, 9.0, 16.0, 10.0, 4.0, 12.0, 10.0, 6.0, 3.0, 50.0))

#OLD-values general solution controls
#mdb.models['Model-1'].steps['Step-1'].control.setValues(
#    discontinuous=ON, timeIncrementation=(8.0, 10.0, 9.0, 16.0, 10.0, 4.0, 12.0, 12.0, 6.0,
3.0, 50.0))

# NEW values gen.sol.cntrl Ir 12 & Ia 15
mdb.models['Model-1'].steps['Step-1'].control.setValues(discontinuous=OFF,
    timeIncrementation=(8.0, 12.0, 9.0, 16.0, 10.0, 4.0, 12.0, 15.0, 6.0, 3.0,
    50.0))
```

```
regionDef=mdb.models['Model-1'].rootAssembly.sets['RefPoint-1']
#mdb.models['Model-1'].historyOutputRequests['H-Output-1'].setValues(
#    variables=('U1','U2','RF1','RF2'), region=regionDef, sectionPoints=DEFAULT,
rebar=EXCLUDE, frequency=1,)

mdb.models['Model-1'].HistoryOutputRequest(name='H-Output-T', createStepName='Step-1',
variables=('U2','RF2' ),
    region=mdb.models['Model-1'].rootAssembly.sets[NameRef2], sectionPoints=DEFAULT,
rebar=EXCLUDE)
mdb.models['Model-1'].HistoryOutputRequest(name='H-Output-L', createStepName='Step-1',
variables=('U1','RF1' ),
    region=mdb.models['Model-1'].rootAssembly.sets[NameRef1], sectionPoints=DEFAULT,
rebar=EXCLUDE)
del mdb.models['Model-1'].historyOutputRequests['H-Output-1']

###################################################################################
#CREATE  BC  IN Y DIRECTION
###################################################################################

#alpha = HH * strainperc /100.0
alpha = 10
DefMat=[(UNSET,0.0,0.0),(0.0,alpha,0.0), (0.0,0.0,UNSET)]

mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Step-1',
    distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name=
    'BC-REF-1', region=Region(referencePoints=(
    mdb.models['Model-1'].rootAssembly.instances[NameRef1].referencePoints[1],
    )), u1=DefMat[0][0], u2=DefMat[0][1], u3=DefMat[0][2], ur1=UNSET,ur2=UNSET,ur3=UNSET)
mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Step-1',
    distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name=
    'BC-REF-2', region=Region(referencePoints=(
    mdb.models['Model-1'].rootAssembly.instances[NameRef2].referencePoints[1],
    )), u1=DefMat[1][0], u2=DefMat[1][1], u3=DefMat[1][2], ur1=UNSET,ur2=UNSET,ur3=UNSET)

mdb.models['Model-1'].YsymmBC(name='BC-Bottom',
    createStepName='Step-1', region=mdb.models['Model-1'].rootAssembly.sets['Bottom'],
localCsys=None)
#mdb.models['Model-1'].EncastreBC(createStepName='Step-1', localCsys=None, name=
#    'BC-BottomEncastre', region=mdb.models['Model-1'].rootAssembly.sets['Bottom'])
mdb.models['Model-1'].XsymmBC(name='BC-Left',
    createStepName='Step-1', region=mdb.models['Model-1'].rootAssembly.sets['Left'],
localCsys=None)

###################################################################################
#CREATE EQUATIONS FOR PERIODIC TOP AND RIGHT
###################################################################################

NameSet = ['Top','Right']
comb = ['1','0']
dimm = [2,1]
for turnn in range(0,len(NameSet)):
    nodesAll=mdb.models['Model-1'].rootAssembly.sets[NameSet[turnn]].nodes
    ranNodes=range(0,len(nodesAll))
    repConst=0
    for repnod1 in range(0,len(nodesAll)):
        mdb.models['Model-1'].rootAssembly.Set(name='Node-'+str(repConst)+str(NameSet[turnn]),
nodes=
           mdb.models['Model-1'].rootAssembly.sets[NameSet[turnn]].nodes[repnod1:repnod1+1])
        mdb.models['Model-1'].Equation(name=NameSet[turnn]+str(dimm[turnn])+'-'+str(repConst),
    terms=((1.0,'Node-'+str(repConst)+str(NameSet[turnn]), dimm[turnn]),(-1.0, 'RefPoint-
'+str(comb[turnn]), dimm[turnn])))
        repConst=repConst+1        #Increase integer for naming equation constraint


##########################################

# CREATE JOB
mdb.Job(atTime=None, contactPrint=OFF, description=
    'Half UnitCell displacement SIM', echoPrint=OFF, explicitPrecision=SINGLE,
    getMemoryFromAnalysis=True, historyPrint=OFF, memory=90, memoryUnits=
    PERCENTAGE, model='Model-1', modelPrint=OFF, name='Job-HalfUnitCell'
    , nodalOutputPrecision=SINGLE, queue=None, resultsFormat=ODB, scratch='',
    type=ANALYSIS, userSubroutine='', waitHours=0, waitMinutes=0)

mdb.jobs['Job-HalfUnitCell'].submit(consistencyChecking=OFF)
```