b
UNIVERSITÄT
BERN

Flesh Simulation in the Field of Animation

Bachelor Thesis

submitted in fulfilment of the requirements for the degree of
Bachelor of Science (B.Sc.)

at the

University of Bern
Institute of Computer Science

1. Prüfer: Prof. Dr. David Bommes
2. Prüfer: Max Mustermann

Eingereicht von: Corina Danja Masanti
Matrikelnummer: 15-128-655
Datum der Abgabe: 01.01.2001

Vorwort

Dies ist ein Vorwort

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Contents

Chapter 1

Introduction

"Animation offers a medium of story telling and visual entertainment which can bring pleasure and information to people of all ages everywhere in the world."
- Walt Disney

1.1 Motivation

With steadily increasing computational power, the demand of better looking results is constantly growing. Especially in the field of animation and simulation we are no longer happy with mediocre results. In the entertainment sector animation studios like Pixar© brought us movies of highest quality. They have made groundbreaking progress over the years. This can easily be observed, when we compare today's work with that from ten years ago.

As always, we have different requirements for each use. In some cases we want to exaggerate a movement or a reaction in a certain way. We can for example create a massive explosion in a movie that would not be half as spectacular in the real world.

In other scenarios we want to come as close as possible to reality. For instance, we may want an animated character to move and physically interact with its environment as a real human being would. Otherwise, the human brain would immediately recognize that some things do not add up. The goal here is to bring characters quite literally to life. We

can add small details like visible breathing and small wrinkles to have an even more convincing effect. We aim to create the illusion of a character with personality, thought and emotions. In order to achieve this effect, we need the character to move and react physically correct.

In the paper *Stable Neo-Hookean Flesh Simulation* [SGK18], the authors addressed exactly this problem of making an animated movement of a human-like character look as natural as possible. This thesis is based heavily on this paper. In the following, I will abbreviate the name of the paper with *SNH-FS*.
But before diving further into the content of the paper, a fundamental background is needed. In order to animate a physical movement, we first need to understand the physics behind it. This requires some knowledge in the field of continuum mechanics. Unfortunately, for most of us it has yet to be learned. The goal of this thesis is to give the necessary physical and mathematical background for a regular computer science student to understand the field of animation. In addition I will go deeper into the thematics of the paper *SNH-FS*. I aim to get an understanding of their contribution in the field and implement their proposed energy myself.

1.2 Structure

In the following, I will give a brief overview of the necessary mathematical background and deliver an introduction into the field of continuum mechanics. Next, I will go through the ideas mentioned in the paper and include some calculations and visualisations that serve for a better understanding. Lastly, I will give an insight into the process of implementing the energy.

Chapter 2

Background

My goal is the same as in the paper *SNH-FS*, namely to animate human-like characters. More precisely, the focus is on the behaviour of the character's flesh. This chapter serves as an introduction into the mathematical as well as physical background, needed to understand the upcoming calculations and conclusions. At the beginning of this chapter, I will define the notation and convention used throughout this whole thesis. Next, I will give insights in the mathematics used and present some of the concepts used in continuum mechanics. However, I will not include each proof explicitly, as there are already many good resources for an interested reader.

2.1 Notation and Convention

At first, I will declare the notation used in this thesis to avoid misunderstandings. I will use the common notation used in continuum mechanics taken from the book *Continuum Mechanics* [Spe80]. Additionally, I will include some more specific declarations formulated and used by the authors of the paper *SNH-FS*.

2.1.1 General Notation

Scalars are represented by regular, normal-weight variables such as a, whereas tensors and matrices are represented by upper-case bold letters

such as for example $\mathbf{A}$. Vectors will be denoted by bold lower-case variables like $\mathbf{a}$.

2.1.2 Tensor Notation

Furthermore, I will use the tensor notation used in *SNH-FS*. They decided to define the vectorization $\text{vec}(\cdot)$ as column-wise flattening of a matrix into a vector ([SGK18], 12:5) similar to Golub and Van Loan (2012) ([GV12]).

In order to indicate that I am dealing with a vectorized matrix, I will use the symbol $\check{\cdot}$ as shown in the following equation:

$$\mathbf{A} = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \qquad \text{vec}(\mathbf{A}) = \check{\mathbf{a}} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

Additionally, I will have to deal with 4^{th} order tensors in a form of matrix-of-matrices. These matrices are denoted by using blackboard bold:

$$\mathbb{A} = \begin{bmatrix} \begin{bmatrix} a & c \\ b & d \end{bmatrix} & \begin{bmatrix} i & k \\ j & l \end{bmatrix} \\ \begin{bmatrix} e & g \\ f & h \end{bmatrix} & \begin{bmatrix} m & o \\ n & p \end{bmatrix} \end{bmatrix} = \begin{bmatrix} [\mathbf{A}_{00}] & [\mathbf{A}_{01}] \\ [\mathbf{A}_{10}] & [\mathbf{A}_{11}] \end{bmatrix}$$

We can now vectorize $\mathbb{A}$ and get the following result:

$$\text{vec}(\mathbb{A}) = \left[\text{vec}\left(\mathbf{A}_{00}\right) \,\middle|\, \text{vec}\left(\mathbf{A}_{10}\right) \,\middle|\, \text{vec}\left(\mathbf{A}_{01}\right) \,\middle|\, \text{vec}\left(\mathbf{A}_{11}\right) \right] = \check{\mathbf{A}}$$

This term above is equivalent to

$$\check{\mathbf{A}} = \begin{bmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ d & h & l & p \end{bmatrix}.$$

The advantage of this form is that we can write several expressions as a cross product. I will need this property later to simplify complicated expressions and calculations.

2.1.3 Summary

Here is a quick overview of the introduced notation:

a : Scalar

$\mathbf{A}$: Matrix or tensor

$\mathbf{a}$: Vector

$\check{\mathbf{a}}$: Vectorized matrix (also written as vec($\mathbb{A}$))

$\check{\mathbf{A}}$: matrix-of-matrices

2.2 Mathematical Background

Since mathematics play an important role in the field of interest, the first step is to build a solid background before diving further into more complex calculations. This section covers all the important concepts used later in the calculations. A basic understanding of linear algebra is assumed.

2.2.1 Singular Value Decomposition

The Singular Value Decomposition (SVD) will play an important role in the formulation of the deformation gradient. It represents the best

possible approximation of a given matrix by a matrix of low rank. This approximation can be looked at as a compression of the data given ([LM15], p. 295). Firstly we need to define what singular values are.

Definition 1 (**Singular Values**). *The singular values of a matrix* $\mathbf{A} \in \mathbb{R}^{mxn}$ *are the square roots of the eigenvalues of* $\mathbf{A}\mathbf{A}^\intercal$.

The theorem of the singular value decomposition tells us that we can factor every m-by-n matrix into one orthogonal m-by-m, one diagonal m-by-n and one orthogonal n-by-n matrix. More formally:

Theorem 1 (**The SVD Theorem**). *Let* $\mathbf{A} \in \mathbb{R}^{mxn}$ *be a matrix having r positive singular values,* $m \geq n$. *Then there exist orthogonal matrices* $\mathbf{U} \in \mathbb{R}^{mxm}$, $\mathbf{V} \in \mathbb{R}^{nxn}$ *and a diagonal matrix* $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{mxn}$ *such that*

$$\mathbf{A} = \mathbf{U}\tilde{\mathbf{\Sigma}}\mathbf{V}^\intercal$$
$$\tilde{\mathbf{\Sigma}} = \begin{bmatrix} \mathbf{\Sigma} & 0 \\ 0 & 0 \end{bmatrix}$$

where $\mathbf{\Sigma} = diag\ (\sigma_1, \sigma_2, \ldots, \sigma_r)$, *and* $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ *are the positive singular values of* $\mathbf{A}$.

This definition and theorem were taken from *Numerical linear algebra with applications: Using MATLAB* ([For14], p.113, p.300).

2.2.2 Polar Decomposition

Another theorem we will use in the next sections is the Polar Decomposition Theorem.

Theorem 2 (**The Polar Decomposition Theorem**). *Let* $\mathbf{F}$ *be a non-singular square matrix. Then* $\mathbf{F}$ *can be decomposed uniquely into either of the following two products*

$$\mathbf{F} = \mathbf{R}\mathbf{U}, \quad \mathbf{F} = \mathbf{V}\mathbf{R}$$

where $\mathbf{R}$ *is orthogonal and* $\mathbf{U}$ *and* $\mathbf{V}$ *are positive definite symmetric matrices.*

This theorem was taken from *Continuum Mechanics* in which they include the proof for 3×3 matrices ([Spe80], p.12).

2.2.3 Frobenius Norm

Definition 2 (**Frobenius Norm**). *Let* $\boldsymbol{A}$ *be a* $(m \times n)$ *matrix in the real or complex domain. Then the Frobenius norm is defined as*

$$\|\mathbf{A}\|_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}.$$

We can further represent the norm with the trace of the matrix, in which $\mathbf{A}^*$ is the conjugate transpose of $\mathbf{A}$. Going from there, we can use the SVD of $\mathbf{A}$ and write the norm with respect to the singular values of $\mathbf{A}$, denoted by σ_i:

$$\|\mathbf{A}\|_F = \sqrt{\text{trace}(\mathbf{A}\mathbf{A}^*)} = \sqrt{\sum_{i=1}^{\min\{m,n\}}\sigma_i^2} \tag{2.1}$$

2.3 Continuum Mechanics

In this section, I will give a broad introduction into the field of Continuum Mechanics. In Continuum Mechanics, we are less interested in small particles like atoms or molecules of an object but rather in pieces of matter which are in comparison very large. The reason for this is that the calculation for the behaviour of individual atoms is very difficult for larger systems. We are therefore concerned with the mechanical behavior of solids and fluids on the macroscopic scale and we treat material as uniformly distribeted throughout regions of space. With this approach, it is possible to define quantities such as displacement, density, etc. as continuous functions of position ([Spe80], p. 1).

2.3.1 Deformation

In continuum mechanics the term *strain* is used as a measure of deformation and we denote *stress* as the force per unit area. If we apply a force over an object, the object itself undergoes a deformation. This behavior is intuitively clear. Graphically, we can imagine a deformation with the help of a two dimensional deformation map as shown in Fig. 2.1. The ellipse on the left side represents an object in its rest state. A function ϕ maps this rest state of the ellipse to a deformed state as shown on the right side.

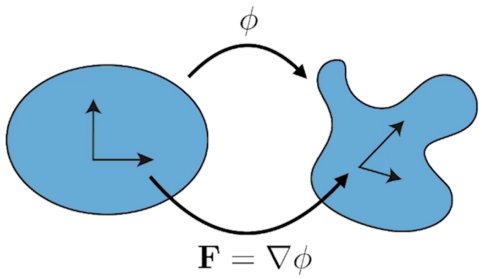


Figure 2.1: Deformation Map [Pix]

We can imagine that we map each particle of a chosen object from its rest state to a deformed one. Each particle X of a body can be characterized by a vector $\mathbf{x}$ containing its positional coordinates at time $t_0 = 0$. This is the reference configuration. If the particles are displaced after a time t, we can describe their new coordinate vector with

$$\check{\mathbf{x}} = \phi(\mathbf{x}, t).$$

If we consider, for example a *Translation*, every particle is displaced by the same distance and direction. Hence, we can calculate $\check{\mathbf{x}}$ with

$$\check{\mathbf{x}} = \mathbf{x} + \mathbf{c}(t)$$

where $\mathbf{c}$ is a vector that only depends on t ([Spe80], p.63).

TODO: Include own image instead of the one here? Other example than translation? Explain strain and stress more?

2.3.2 Deformation Gradient

The next quantity I will use in the upcoming chapters is the deformation gradient **F**. With its help, we can calculate the change in volume and length of an object during a deformation. We can obtain **F** by taking the gradient of the function ϕ discussed in the previous section.

As an example, I am taking a simple deformation in 2D. The coloured area in Fig. 2.2 represents an object in its rest state. We can stretch this area in the following way:

$$\check{x} = 1.5x + 0y$$
$$\check{y} = 0x + 2y$$

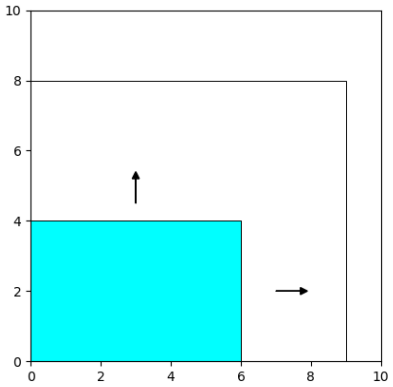


Figure 2.2: Stretching of a rectangle

The resulting deformation gradient can then be obtained by

$$\mathbf{F} = \begin{bmatrix} \frac{\partial[1.5x+0y]}{\partial x} & \frac{\partial[1.5x+0y]}{\partial y} \\ \frac{\partial[0x+2y]}{\partial x} & \frac{\partial[0x+2y]}{\partial y} \end{bmatrix} = \begin{bmatrix} 1.5 & 0.0 \\ 0.0 & 2.0 \end{bmatrix}.$$

If **F** is equal to the identity matrix **I**, there is no deformation present. This would be the case for rigid body displacements. Since we will be handling deformation in the 3D space, the deformation gradient will have the form of a 3×3 matrix:

$$\mathbf{F} = \left[f_0 \middle| f_1 \middle| f_2 \right] = \begin{bmatrix} f_0 & f_3 & f_6 \\ f_1 & f_4 & f_7 \\ f_2 & f_5 & f_8 \end{bmatrix} \tag{2.2}$$

F can be factorized and used to calculate other quantities. The next few sections will explain this.

Singular Value Decomposition of F

Using the SVD theorem shown in Thm.1, **F** can be written in the form of

$$\mathbf{F} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\intercal} \tag{2.3}$$

in which $\mathbf{\Sigma}$ stands for

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_0 & 0 & 0 \\ 0 & \sigma_1 & 0 \\ 0 & 0 & \sigma_2 \end{bmatrix}. \tag{2.4}$$

The σ_i denote the singular values of $\mathbf{\Sigma}$. $\mathbf{U}$ and $\mathbf{V}$ are both orthogonal matrices that represent the rotation of $\mathbf{F}$. $\mathbf{\Sigma}$ on the other hand indicates the scaling of each coordinate x_i by the factor σ_i. Unlike the standard convention, where a possible reflection lies in the rotation variants (meaning $\mathbf{U}$ and $\mathbf{V}$), here the reflections are moved to $\mathbf{\Sigma}$ and it is therefore allowed to have a negative entry. This has the effect that $\det(\mathbf{U}) = \det(\mathbf{V}) = 1$.

Polar Decomposition of the Deformation Gradient

With the help of Thm. 2 we can decompose the deformation gradient into the form

$$\mathbf{F} = \mathbf{R}\mathbf{S}, \tag{2.5}$$

where $\mathbf{R}$ is orthogonal and $\mathbf{S}$ is a positive definite symmetric matrix. $\mathbf{R}$ symbolises the rotation (with possible reflection) that $\mathbf{F}$ undergoes, whereas $\mathbf{S}$ contains the scaling along the orthogonal directions of $\mathbf{F}$.

Relative volume change

A useful information about a deformation is the relative volume change of the deformed object. It can be calculated by the determinate of $\mathbf{F}$

$$J = \det(\mathbf{F}). \tag{2.6}$$

For a normal deformation, J is a positive value. A determinant of zero would mean that the object is being deformed into a zero volume state, e.g. a plane or a point. A negative determinant indicates an inversion.

Cauchy-Green

The right Cauchy-Green tensor $\mathbf{C}$ can be calculated by

$$\mathbf{C} = \mathbf{F}^{\intercal}\mathbf{F}, \tag{2.7}$$

and is a 3×3 matrix for deformations in the 3D domain. Using $\mathbf{C}$, we can calculate the first right Cauchy-Green invariant I_C with

$$I_C = \mathrm{tr}(\mathbf{C}). \tag{2.8}$$

This quantity describes the length change of the object after a deformation.

2.3.3 Deformation Energy

The deformation energy or strain energy Ψ is a scalar that defines the stored energy of a object undergoing a deformation. The strain energy density is the strain energy per unit volume. It equal to the work that has to be done by the stresses in order to alter the strains ([Kor17], p.10). This means that the energy is an indicator of how much force must be applied in order for an object to be deformed in a certain way. Thus, we can use the deformation energy to express the relationship between the stresses and strains. We can illustrate this with a stress-strain curve shown in Fig. 2.3. The area under the stress-strain curve corresponds to the strain energy.

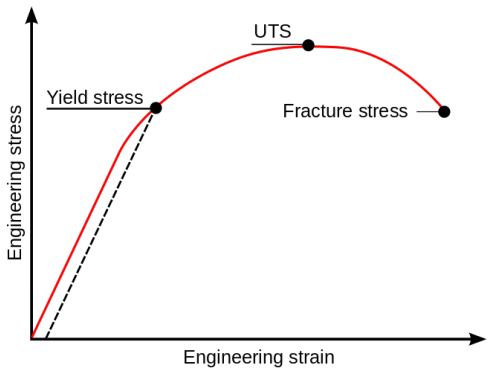


Figure 2.3: Example for a stress-strain curve[1]

This relationship is not always linear. For example if we reach a stress state that produces yielding, we cannot assume that the relationship is linear anymore. If the material reaches the yield point, it can no longer recover into its rest shape due to permanent fractures during the deformation. This point is illustrated in Fig. 2.3 with *Yield stress*. *UTS* in Fig. 2.3 is abbreviated for Ultimate tensile strength and defines the maximal stress an object can bear before breaking.

The behaviour of the object depends heavily on the material it consists of. This will be further explained in the next section concerning *Material Constants*. In order to get a convincing simulation for a specific material, we have to choose an appropriate energy function. The behaviour of human flesh can be put into the category of hyperelastic materials. So we need a hyperelastic energy. The key property of elasticity is, that if all the forces that are applied over an object are removed, the object recovers into its original shape and volume ([Kor17], p. 5). For deformations of elastic materials, the Hooke's equation holds to some extend. Hooke's law describes a linear relationship of the applied force and the distance to the original position of a spring:

$$F = kx$$

F decribes the force, x the distance and k is a constant value. Material for which this equation holds are called linear-elastic or Hookean.

TODO: Explain Hooke better and add source for hookean, Neo-Hookean Model in Hyperelasticity. Not happy with last paragraph, maybe additional subsection?

2.3.4 Material Constants

When we look at a deformation of an object, we need to consider the material the object consists of. A material can be very stiff like steel or easily deformable like rubber. In order to measure the deformation of a specific material, we need the *Poisson's Ratio* of said material. The poisson's ratio is a material constant that is defined as

[1] https://commons.wikimedia.org/wiki/File:Stress-strain_curve.svg

$$\sigma = -\frac{\epsilon_{11}}{\epsilon_{22}} \in [-1, 0.5] \tag{2.9}$$

where ϵ_{11} is the lateral and ϵ_{22} the axial strain. The range in which σ lies in starts at -1 and goes up to 0.5 [MR09]. If we imagine pulling a rubber band on each of its sides, we can observe that the band gets longer and the middle part gets narrower. The poisson's ratio indicates the extend of this process. Incompressible materials such as rubber lead to a higher poisson's ratio.

Usually the poisson's ratio of a material is positive. A negative value would mean that the material becomes wider in the cross section when it is stretched. This behaviour is very uncommon in nature. Examples of materials with a negative poisson's ratio are for instance discussed in *Foam structures with a negative Poisson's ratio* [Lak87] or *Advances in negative Poisson's ratio materials* [Lak93]. In table 2.1 are some examples of the poisson's ratio of various materials.

Material	**Poisson's ratio**
C (graphite)	0.31
Sn (metal)	0.357
Cu	0.355
Zn	0.25
Ag	0.36
Au	0.45
Concrete	0.20–0.37
Titanium (dental alloy)	0.30–0.31
Bronze	0.34
18–8 Stainless steel	0.305
Natural rubber	0.4999
B_2O_3 glass	0.30
GeO_2 glass	0.20

Table 2.1: Different materials with their poisson's ratio ([MR09], p. 3)

In the context of flesh simulations, I am using the poisson's ratio as a characterization for the resistance to volume change of flesh. The poisson's ratio of biological tissues such as flesh, fat and muscles takes on higher values in the range of 0.45 and 0.5 ([SGK18], 12:1).

The calculation of the poisson's ratio defined in Eq. 2.9 is a challenge.

Fortunately, we can make use of the *Lamé Parameters*, the two material specific constants μ and λ. With the help of these two constants we can transform Eq. 2.9 into the form

$$\sigma = \frac{\lambda}{2(\lambda + \mu)}. \tag{2.10}$$

This equation allows to calculate the poisson's ratio much easier ([Ber15], p. 231).

TODO: Include Piola-Kirchhoff Stress?

Chapter 3

Stable Neo-Hookean Flesh Simulation

In this chapter, I will examine further the topic of the paper *SNH-FS*. In the interest of understanding the thought process of the authors, I will include some of their calculations more detailed. In addition, examples and visualisations are presented for a better understanding.

3.1 Deformation Gradient

In the following calculations the properties of the deformation gradient $\mathbf{F}$ covered in chapter 2 will be used. These properties are summarized in table 3.1 for a better overview.

Symbol	**Definition**
$\mathbf{F} = \mathbf{RS}$	Polar decomposition
$J = \det(\mathbf{F})$	Relative volume change
$\mathbf{C} = \mathbf{F}^\intercal\mathbf{F}$	Right Cauchy-Green
$I_C = \mathrm{tr}(\mathbf{C})$	First right Cauchy-Green invariant

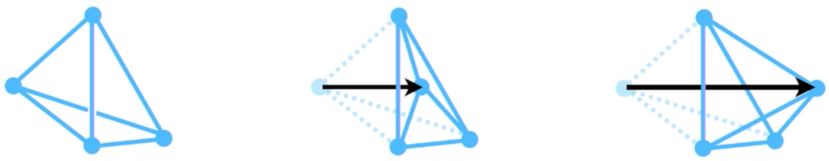
Table 3.1: Quantities derived from the Deformation Gradient

3.2 Energy Formulation

3.2.1 Stability

The core goal of the paper was to model deformations for virtual characters that have human-like features. In order to achieve better results than what has been done in current research, they formulated a new deformation energy. In chapter 2, I concluded that the appropriate energy for animating soft tissues such as flesh has to be hyperelastic. An important property is the stability of the energy. We need an energy that is stable in the following four ways:

1. Inversion Stability: Given some arbitrary object, it is possible that while deforming the object we can arrive at a zero volume state or even an entire inversion. Take for example the tetrahedron shown in Fig. 3.1a. In Fig. 3.1b we see a deformed state of this tetrahedron where the volume is scaled down to zero and we are left with a simple triangle. In Fig. 3.1c the tetrahedron arrives at an inversed stated. The deformation energy has to be able to deal with both cases without creating severe artefacts. That means, that the energy has to be singularity-free, meaning that it is defined for every possible point. This should hold without needing any filters or threshold ([SGK18], 12:3).



(a) Rest state (b) Zero-volume state (c) Inversed state

Figure 3.1: Inversion of a tetrahedron [Pix]

2. Reflection stability: While deforming an object, it can occur that we are dealing with reflections. An example of a reflection in 2D is shown in Fig. 3.2. Here the coloured triangle is reflected over the y-axis. A matrix that represents a reflection is orthogonal with determinant -1. The deformation energy needs to be well behaved under reflections. This has to hold, regardless of the reflection convention used in the SVD of $\mathbf{F}$. The reflection convention is explained in chapter 2.3.2 in the subsection *Singular Value Decomposition of* $\boldsymbol{F}$.

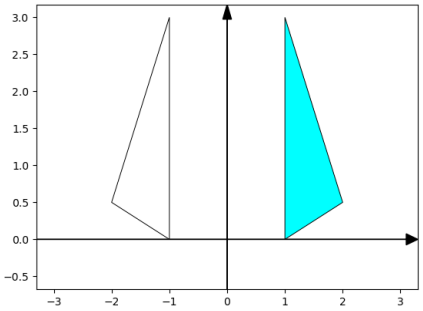


Figure 3.2: Reflection of a triangle over the y-axis

3. Rest stability: When deforming an object in a certain way, we apply one or multiple forces to that object, which influences the deformation. But when we remove all the forces, the object should be back in its rest state.

4. Meta-stability under degeneracy: This is a special case of rest stability. We can crush an object into an arbitrary shape like a plane, line or point. This process is illustrated for a cube in Fig. 3.3. Afterwards, when the applied forces are removed, the cube should still be able to recover from this arbitrary shape to its actual shape.

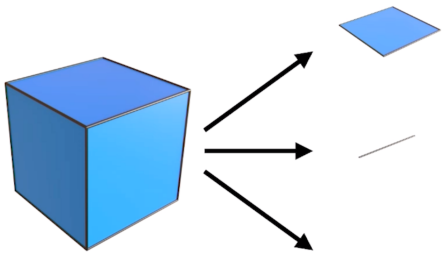


Figure 3.3: Illustration of meta stability [Pix]

Based on these four requirements, we will determine if a deformation energy is suited for our needs.

TODO: Are images ok?

3.2.2 Existing Neo-Hookean Energies

In previous literature, a few energies were proposed that I will analyse in this section. They are listed in table 3.2.

Energy	**Author(s)**
$\Psi_{Neo} = \frac{\mu}{2}(I_C - 3) - \mu \log J + \frac{\lambda}{2}(\log J)^2$	e.g. Bonet and Wood 1997 ([BW97])
$\Psi_{\mathrm{A}} = \frac{\mu}{2}(I_C - 3) - \mu \log J + \frac{\lambda}{2}(J-1)^2$	Odgen 1997 ([Ogd97])
$\Psi_{\mathrm{B}} = \frac{\mu}{2}\left(J^{-2/3} I_C - 3\right) + \frac{\lambda}{2}(J-1)^2$	Bower 2009 ([Bow09])
$\Psi_{\mathrm{C}} = \frac{\mu}{2}\left(J^{-2/3} I_C - 3\right) + \frac{\lambda}{2}(J-1)^2$	Wang and Yang 2016 ([WY16])

Table 3.2: Summary of proposed energies ([SGK18], 12:3)

Each energy formulation can be split up into a 1D length term and a 3D volume term. The 1D length term penalizes the length changes an object undergoes, whereas the 3D volume term is a volume-preserving penalty term.

Energy	**1D length term**	**3D volume term**
Ψ_{Neo}	$\frac{\mu}{2}(I_C - 3)$	$-\mu \log J + \frac{\lambda}{2}(\log J)^2$
Ψ_{A}	$\frac{\mu}{2}(I_C - 3)$	$-\mu \log J + \frac{\lambda}{2}(J-1)^2$
Ψ_{B}	$\frac{\mu}{2}\left(J^{-2/3} I_C - 3\right)$	$\frac{\lambda}{2}(J-1)^2$
Ψ_{C}	$\frac{\mu}{2}\left(J^{-2/3} I_C - 3\right)$	$\frac{\lambda}{2}(J-1)^2$

Table 3.3: Energies split up into their 1D length and 3D volume term

1D Length Term

Mooney ([Moo40]) originally proposed the 1D length term

$$\Psi_M = \frac{\mu}{2}(I_C - 3),$$

which is used in Ψ_{Neo} and Ψ_A. If we expand the energy with the singular values of the deformation gradient $\mathbf{F}$, we get the following term:

$$\Psi_M = \frac{\mu}{2}\left(\sigma_0^2 + \sigma_1^2 + \sigma_2^2 - 3\right)$$

This energy reaches its minimum at a zero volume state, meaning $I_C = 0$. Mooney added the hard constraint that J should be equal to 1, so that

the energy is minimized at the volume preserving configuration that is closest to the stretch space origin. Note that the energy is singularity free and well defined under inversion.

The second term is

$$\Psi_R = \frac{\mu}{2}\left(J^{-2/3}I_C - 3\right)$$

and used in Ψ_B and Ψ_C and was introduced by Rivlin ([Riv48]). Using the singular values of $\mathbf{F}$, we get the following term:

$$\Psi_R = \frac{\mu}{2}\left(\frac{\sigma_0^2 + \sigma_1^2 + \sigma_2^2}{(\sigma_0\sigma_1\sigma_2)^{\frac{2}{3}}} - 3\right)$$

Unfortunately, this term is not singularity free. If either σ_0, σ_1 or σ_2 is equal to zero the result is not defined anymore.

3D Volume Term

The volume term of Ψ_{Neo}, meaning

$$\Psi_{Neo,volume} = -\mu \log J + \frac{\lambda}{2}(\log J)^2,$$

results in numerical problems, since the logarithmic function is not defined for $J < 0$ and grows unbounded for $J \to 0$. In conclusion, $\Psi_{Neo,volume}$ is not singularity free. The same applies for the 3D volume term of Ψ_A, namely

$$\Psi_{A,volume} = -\mu \log J + \frac{\lambda}{2}(J-1)^2.$$

The term of Ψ_A and Ψ_B, which is of the form

$$\Psi_M = \frac{\lambda}{2}(J-1)^2,$$

does not have these problems. It is bounded, well defined and invertible. After these observations we combine the robust length with the robust volume term and receive Ψ_D, which is defined as

$$\Psi_D = \frac{\mu}{2}(I_C - 3) + \frac{\lambda}{2}(J-1)^2.$$

Ψ_D is singularity free and well defined under inversion. Unfortunately, it does not satisfy the requirement of being rest stable, which will be discussed in the next section.

TODO: explain how to get to equation with singular values?

3.2.3 Rest Stabilization

Although Ψ_D meets almost all stated requirements, it is not rest stable. This can be shown with the first Piola-Kirchhoff (PK1) stress tensor, defined as Ψ_D derived by the deformation gradient $\mathbf{F}$:

$$\begin{aligned} P_D(\mathbf{I}) &= \frac{\partial \Psi_D}{\partial \mathbf{F}}(\mathbf{I}) = \frac{\partial \Psi_D}{\partial \mathbf{F}} \left[\frac{\mu}{2} (I_C - 3) + \frac{\lambda}{2}(J-1)^2 \right] \\ &= \frac{\partial \Psi_D}{\partial \mathbf{F}} \frac{\mu}{2} (\mathrm{tr}(\mathbf{I}^\intercal \mathbf{I}) - 3) + \frac{\partial \Psi_D}{\partial \mathbf{F}} \frac{\lambda}{2} (\mathrm{tr}(\mathbf{I}) - 1)^2 \\ &= \mu \mathbf{I} + \lambda(\det(\mathbf{I}) - 1) \frac{\partial}{\partial \mathbf{F}} \det(\mathbf{I}) = \mu \mathbf{I} \neq 0 \end{aligned}$$

If the energy had rest stability, $P_D(\mathbf{I})$ would resolve to zero. Unfortunately, this is not the case here. In order to solve this problem, the authors modified $(J-1)^2$ to $(J-\alpha)^2$. Using this modification, the energy can be written as

$$\Psi_E = \frac{\mu}{2} (I_C - 3) + \frac{\lambda}{2}(J-\alpha)^2.$$

Inserting Ψ_E into the PK1 equation from before, we get

$$\begin{aligned} P_E(\mathbf{F}) &= \frac{\partial \Psi_E}{\partial \mathbf{F}} \left[\frac{\mu}{2} (I_C - 3) + \frac{\lambda}{2}(J-\alpha)^2 \right] \\ &= \mu \mathbf{F} + \lambda(\det(\mathbf{F}) - \alpha) \frac{\partial}{\partial \mathbf{F}} \det(\mathbf{F}). \end{aligned}$$

Solving for an alpha that satisfies $P_E(\mathbf{I}) = 0$ gives us $\alpha = 1 + \frac{\mu}{\lambda}$. Now Ψ_E has to be changed accordingly:

$$\begin{aligned} \Psi_E &= \frac{\mu}{2} (I_C - 3) + \frac{\lambda}{2}(J - 1 - \frac{\mu}{\lambda})^2 \\ &= \frac{\mu}{2} (I_C - 3) - \mu (J-1) + \frac{\lambda}{2}(J-1)^2 + \left(\frac{\mu}{\lambda}\right)^2. \end{aligned}$$

Since constants disappear under differentiation, this expression is functionally equivalent to

$$\Psi_E = \frac{\mu}{2}(I_C - 3) - \mu(J-1) + \frac{\lambda}{2}(J-1)^2.$$

Note that Ψ_E looks very similar to Ψ_{Neo}. The difference is that $log(J)$ is replaced with $(J-1)$ in Ψ_E. Keep in mind that $(J-1)$ is the first term in the taylor approximation of $\log(J)$ at $J = 1$:

$$\begin{aligned}\sum_{n=0}^{\infty} &= \frac{f^{(n)}(1)}{n!}(J-1)^n \\ &= \boldsymbol{(J-1)} - \frac{1}{2}(J-1)^2 + \frac{1}{3}(J-1)^3 - \frac{1}{4}(J-1)^4 + ...\end{aligned}$$

Thus Ψ_E is an approximation of Ψ_{Neo}.

TODO: Include all calculations? Check consistency of calculations (when trace when I_C etc.).

3.2.4 Meta-Stability under Degeneracy

We have seen that he energy Ψ_E has inversion, reflection and rest stability. Now we are interested in how it behaves under degeneracy. The authors of the paper *SNH-FS* viewed this as a Drucker stability analysis (see [Bow09]). By Drucker stability, we mean a set of criteria that a material can satisfy or not. This is a measurement of the stability of the material. The calculations can be found in the supplemental material of *SNH-FS*, I will only present the results here.

The energy remains stable when crushing the object into a *plane*. The object is able to return to its rest state. Thus, no further adjustments need to be done. When crushing the object into a *line*, the energy is meta-stable. A meta-stable state is a weaker state of stability that is resistant to small changes but not to large ones. The alternative would be a state that yields singularities, which worsens the situation. Hence, we can leave the energy as it is. After crushing the object to a *point*, the material cannot recover into its rest state. This effect is small for a

higher poisson's ratio. For completeness the authors nevertheless added a regularized origin barrier:

$$\Psi_{origin} = \log(I_C + \delta)$$

This term eliminates the point meta-stability for a positive poisson's ratio, without interfering with the other requirements for the energy. It can be shown that the value δ should be set to 1. This is also included in the supplemental material of the paper ([SGK18], 12:4).

The final energy is then

$$\Psi_{new} = \frac{\mu}{2}(I_C - 3) + \frac{\lambda}{2}(J - \alpha)^2 - \frac{\mu}{2}\log(I_C + 1). \tag{3.1}$$

With that adjustment the rest stability term can be written as $\alpha = 1 + \frac{\mu}{\lambda} - \left(\frac{\mu}{4}\right)\lambda$.

3.2.5 Lamé Reparametrization

In order to stay consistent with Hooke's law, we need to do a reparametrization of μ and λ. For achieving this, we set

$$\mathbf{P}(\mathbf{F}) = 2\mu_{Lam}\epsilon + \lambda_{Lam}\,\mathrm{tr}(\epsilon)\mathbf{I}, \tag{3.2}$$

where $\epsilon = \frac{1}{2}(\mathbf{F} + \mathbf{F}^\intercal) - \mathbf{I}$ is the linearized strain tensor. We then set $\mu = \frac{4}{3}\mu_{Lam}$ and $\lambda = \lambda_{Lam} + \frac{5}{6}\mu_{Lam}$. The equation for the poisson's ratio then shifts to

$$v = \frac{\lambda - \left(\frac{5}{8}\right)\mu}{2\left(\lambda + \left(\frac{1}{8}\mu\right)\right)}.$$

TODO: Do a better job here.

3.3 Energy Analysis

The goal of this chapter is to show that a complete eigenanalysis can be performed on Eq. 3.1. This helps us understand the properties and

limitations of the new energy.

3.3.1 First Piola-Kirchhoff Stress (PK1)

In order to analyse the energy, PK1 can be used for Eq. 3.1 with $\alpha = 1 + \frac{\mu}{\lambda} - \left(\frac{\mu}{4}\right)\lambda$:

$$\begin{aligned} P(\mathbf{F}) &= \frac{\partial \Psi_D}{\partial \mathbf{F}} \left[\frac{\mu}{2}(I_C - 3) + \frac{\lambda}{2}(J-\alpha)^2 - \frac{\mu}{2}(I_C + 1) \right] \\ &= \mu\mathbf{F} + \lambda(\det(\mathbf{F}) - \alpha)\frac{\partial}{\partial \mathbf{F}}\det(\mathbf{F}) - \frac{\partial}{\partial \mathbf{F}}\left[\log\left(\mathrm{tr}(\mathbf{F}^\intercal\mathbf{F}) + 1\right)\right] \\ &= \mu\mathbf{F} + \lambda(\det(\mathbf{F}) - \alpha)\frac{\partial}{\partial \mathbf{F}}\det(\mathbf{F}) - \mu\mathbf{F}\frac{1}{\mathrm{tr}(\mathbf{F}^\intercal\mathbf{F}) + 1} \\ &= \mu\left(1 - \frac{1}{\mathrm{tr}(\mathbf{F}^\intercal\mathbf{F}) + 1}\right)\mathbf{F} + \lambda(J-\alpha)\frac{\partial J}{\partial \mathbf{F}} \end{aligned}$$

For further use it can be more practical to write $\frac{\partial J}{\partial \mathbf{F}}$ as a result of cross products:

$$\frac{\partial J}{\partial \mathbf{F}} = \left[f_1 \times f_2 \,\middle|\, f_2 \times f_0 \,\middle|\, f_0 \times f_1 \right]. \tag{3.3}$$

TODO: Is it easy to follow? Maybe add some explanations of some steps?

3.3.2 The Energy Hessian Terms

Using the scalar notation for $\mathbf{F}$, the Hessian of the energy can be written as a fourth-order matrix-of-matrices:

$$\frac{\partial^2 \psi}{\partial \mathbf{F}^2} = \frac{\partial P(\mathbf{F})}{\partial \mathbf{F}} = \begin{bmatrix} \left[\frac{\partial P(\mathbf{F})}{\partial f_0}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_3}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_6}\right] \\ \left[\frac{\partial P(\mathbf{F})}{\partial f_1}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_4}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_7}\right] \\ \left[\frac{\partial P(\mathbf{F})}{\partial f_2}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_5}\right] & \left[\frac{\partial P(\mathbf{F})}{\partial f_8}\right] \end{bmatrix}$$

Each entry of the Hessian is defined as

$$\frac{\partial P(\mathbf{F})}{\partial f_i} = \frac{\partial}{\partial f_i}\left[\mu\left(1 - \frac{1}{\mathrm{tr}(\mathbf{F}^\intercal\mathbf{F}) + 1}\right)\mathbf{F} + \lambda(J-\alpha)\frac{\partial J}{\partial \mathbf{F}}\right]$$

$$\overset{\text{prod.rule}}{=} \underbrace{\frac{\partial \mathbf{F}}{\partial f_i}\mu\left(1-\frac{1}{\text{tr}(\mathbf{F}^\intercal\mathbf{F})+1}\right)}_{\text{T}_i} + \underbrace{\mathbf{F}\mu\frac{2}{(\text{tr}(\mathbf{F}^\intercal\mathbf{F})+1)^2}f_i}_{\text{M}_i}$$

$$r + \underbrace{\lambda\frac{\partial J}{\partial f_i}\frac{\partial J}{\partial \mathbf{F}}}_{\text{G}_i} + \underbrace{\lambda(J-\alpha)\frac{\partial^2 J}{\partial \mathbf{F}\partial f_i}}_{\text{H}_i} \in \mathbb{R}^3.$$

We can split up this final equation into these four term: T_i (Tikohonov), M_i (Mu), G_i (Volume Gradient), H_i (Volume Hessian). I will examine them separately in the next section.

3.3.3 The Tikhonov, Mu, and Gradient Terms

Tikhonov

The Tikhonov term is a fourth-order matrix-of-matrices. It is calculated by $\frac{\partial \mathbf{F}}{\partial f_i}$ and takes on the form

$$\mathbb{T} = \left[\begin{bmatrix}\begin{bmatrix}1&0&0\\0&0&0\\0&0&0\end{bmatrix} & \begin{bmatrix}0&1&0\\0&0&0\\0&0&0\end{bmatrix} & \begin{bmatrix}0&0&1\\0&0&0\\0&0&0\end{bmatrix}\\ \begin{bmatrix}0&0&0\\1&0&0\\0&0&0\end{bmatrix} & \begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix} & \begin{bmatrix}0&0&0\\0&0&1\\0&0&0\end{bmatrix}\\ \begin{bmatrix}0&0&0\\0&0&0\\1&0&0\end{bmatrix} & \begin{bmatrix}0&0&0\\0&0&0\\0&1&0\end{bmatrix} & \begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix}\end{bmatrix}\right].$$

If we vectorize $\mathbb{T}$, we get the identity matrix $\mathbf{I} \in \mathbb{R}^{9x9}$ which is of full rank, positive definite and independent of the values in $\mathbf{F}$:

$$\text{vec}(\mathbb{T}) = \check{\mathbf{T}} = \mathbf{I} = \in \mathbb{R}^{9x9}$$

The Tikhonov term serves as a regularizer for the rest of the energy.

Mu

The Mu term has the same structure with different entries. It can be obtained by $\mathbf{F}f_i$ and has the form

$$\mathbb{M} = \begin{bmatrix} \begin{bmatrix} f_0^2 & f_0f_3 & f_0f_6 \\ f_0f_1 & f_0f_4 & f_0f_7 \\ f_0f_2 & f_0f_5 & f_0f_8 \end{bmatrix} & \begin{bmatrix} f_3f_0 & f_3^2 & f_3f_6 \\ f_3f_1 & f_3f_4 & f_3f_7 \\ f_3f_2 & f_3f_5 & f_3f_8 \end{bmatrix} & \begin{bmatrix} f_6f_0 & f_6f_3 & f_6f_6 \\ f_6f_1 & f_6f_4 & f_6f_7 \\ f_6f_2 & f_6f_5 & f_6f_8 \end{bmatrix} \\ \begin{bmatrix} f_1f_0 & f_1f_3 & f_1f_6 \\ f_1^2 & f_1f_4 & f_1f_7 \\ f_1f_2 & f_1f_5 & f_1f_8 \end{bmatrix} & \begin{bmatrix} f_4f_0 & f_4f_3 & f_4f_6 \\ f_4f_1 & f_4^2 & f_4f_7 \\ f_4f_2 & f_4f_5 & f_4f_8 \end{bmatrix} & \begin{bmatrix} f_7f_0 & f_7f_3 & f_7f_6 \\ f_7f_1 & f_7f_4 & f_7^2 \\ f_7f_2 & f_7f_5 & f_7f_8 \end{bmatrix} \\ \begin{bmatrix} f_2f_0 & f_2f_3 & f_2f_6 \\ f_2f_1 & f_2f_4 & f_2f_7 \\ f_2^2 & f_2f_5 & f_2f_8 \end{bmatrix} & \begin{bmatrix} f_5f_0 & f_5f_3 & f_5f_6 \\ f_5f_1 & f_5f_4 & f_5f_7 \\ f_5f_2 & f_5^2 & f_5f_8 \end{bmatrix} & \begin{bmatrix} f_8f_0 & f_8f_3 & f_8f_6 \\ f_8f_1 & f_8f_4 & f_8f_7 \\ f_8f_2 & f_8f_5 & f_8^2 \end{bmatrix} \end{bmatrix}.$$

When vectorizing $\mathbb{M}$, the resulting matrix $\check{\mathbf{M}}$ has the squared values of f_i placed on the diagonal:

$$\mathrm{vec}(\mathbb{M}) = \check{\mathbf{M}} = \begin{bmatrix} f_0^2 & f_1f_0 & f_2f_0 & f_3f_0 & f_4f_0 & f_5f_0 & f_6f_0 & f_7f_0 & f_8f_0 \\ f_0f_1 & f_1^2 & f_2f_1 & f_3f_1 & f_4f_1 & f_5f_1 & f_6f_1 & f_7f_1 & f_8f_1 \\ f_0f_2 & f_1f_2 & f_2^2 & f_3f_2 & f_4f_2 & f_5f_2 & f_6f_2 & f_7f_2 & f_8f_2 \\ f_0f_3 & f_1f_3 & f_2f_3 & f_3^2 & f_4f_3 & f_5f_3 & f_6f_3 & f_7f_3 & f_8f_3 \\ f_0f_4 & f_1f_4 & f_2f_4 & f_3f_4 & f_4^2 & f_5f_4 & f_6f_4 & f_7f_4 & f_8f_4 \\ f_0f_5 & f_1f_5 & f_2f_5 & f_3f_5 & f_4f_5 & f_5^2 & f_6f_5 & f_7f_5 & f_8f_5 \\ f_0f_6 & f_1f_6 & f_2f_6 & f_3f_6 & f_4f_6 & f_5f_6 & f_6^2 & f_7f_6 & f_8f_6 \\ f_0f_7 & f_1f_7 & f_2f_7 & f_3f_7 & f_4f_7 & f_5f_7 & f_6f_7 & f_7^2 & f_8f_7 \\ f_0f_8 & f_1f_8 & f_2f_8 & f_3f_8 & f_4f_8 & f_5f_8 & f_6f_8 & f_7f_8 & f_8^2 \end{bmatrix}$$

This structure makes it possible to write $\check{\mathbf{M}}$ as an outer product of $\mathrm{vec}(\mathbf{F})$:

$$\check{\mathbf{M}} = \mathrm{vec}(\mathbf{F})\,\mathrm{vec}(\mathbf{F})^\intercal = \check{\mathbf{f}}\check{\mathbf{f}}^\intercal$$

This matrix has rank one and has a single non-zero eigenvalue. In order to examine the eigenvalues, we can calculate

$$\|\check{\mathbf{f}}\|_2^2 = \sum_{n=0}^{8} |f_n|^2 = \|\mathbf{F}\|_F^2 = \sum_{n=0}^{3} \sigma_i^2 = \left(\sigma_0^2 + \sigma_1^2 + \sigma_2^2\right),$$

in which $\|\cdot\|_F$ stands for the Frobenius norm introduced in Def. 2 and σ_i for the singular values from $\mathbf{\Sigma}$ in the SVD of $\mathbf{F}$ stated in Eq. 2.4. The equation can be obtained by using Eq. 2.1, explained in chapter 2. The eigenvector is $\check{\mathbf{f}}/\|\check{\mathbf{f}}\|$. The eigenvalue is always non-negative and large if $\mathbf{F}$ contains a large strech.

Gradient

The gradient term also has the same structure as the two terms before with different entries defined by

$$\mathbb{G}(\mathbf{F}) = \frac{\partial J}{\partial \mathbf{F}} \frac{\partial J}{\partial fi}.$$

The vectorized matrix $\check{\mathbf{G}}$ can be written in a similar form as for the Mu term. With the help of the cross product of $\partial J/\partial \mathbf{F}$ stated in Eq. 3.3, $\check{\mathbf{G}}$ can be written as an outer product in the following way:

$$\text{vec}(\mathbb{G}(\mathbf{F})) = \check{\mathbf{G}} = \text{vec}\left(\frac{\partial J}{\partial \mathbf{F}}\right) \text{vec}\left(\frac{\partial J}{\partial \mathbf{F}}\right)^{\intercal} = \check{\mathbf{g}}\check{\mathbf{g}}^{\intercal}.$$

As in the Mu term, we again have a single non-zero, non-negative eigenvalue calculated by

$$\|\check{\mathbf{g}}\|_2^2 = \left\|\frac{\partial J}{\partial \mathbf{F}}\right\|_F^2 = \left[(\sigma_0\sigma_1)^2 + (\sigma_0\sigma_2)^2 + (\sigma_1\sigma_2)^2\right].$$

The eigenvector is $\check{\mathbf{g}}/\|\check{\mathbf{g}}\|$.

TODO: Not all proofs and calculations included. Add or leave out some if too confusing.

3.3.4 The Volume Hessian

To make an analysis of the volume Hessian term is a bit trickier. It is of the form

$$\mathbb{H} = \frac{\partial^2 J}{\partial \mathbf{F} \partial f_i} = \begin{bmatrix} \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_0}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_3}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_6}\right] \\ \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_1}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_4}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_7}\right] \\ \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_2}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_5}\right] & \frac{\partial}{\partial \mathbf{F}}\left[\frac{\partial J}{\partial f_8}\right] \end{bmatrix}.$$

Vectorizing $\mathbb{H}$ reveals the structure

$$\mathrm{vec}(\mathbb{H}) = \check{\mathbf{H}} = \begin{bmatrix} 0 & 0 & 0 & 0 & f_8 & 0 & 0 & -f_5 & f_4 \\ 0 & 0 & 0 & -f_8 & 0 & f_6 & f_5 & 0 & -f_3 \\ 0 & 0 & 0 & f_7 & -f_6 & 0 & -f_4 & f_3 & 0 \\ 0 & -f_8 & f_7 & 0 & 0 & 0 & 0 & f_2 & -f_1 \\ f_8 & 0 & -f_6 & 0 & 0 & 0 & -f_2 & 0 & f_0 \\ -f_7 & f_6 & 0 & 0 & 0 & 0 & f_1 & -f_0 & 0 \\ 0 & f_5 & -f_4 & 0 & -f_2 & f_1 & 0 & 0 & 0 \\ -f_5 & 0 & f_3 & f_2 & 0 & -f_0 & 0 & 0 & 0 \\ f_4 & -f_3 & 0 & -f_1 & f_0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

We can write $\check{\mathbf{H}}$ as a cross-product matrix in the form

$$\check{\mathbf{H}} = \begin{bmatrix} 0 & -\widehat{\mathbf{f_2}} & \widehat{\mathbf{f_1}} \\ \widehat{\mathbf{f_2}} & 0 & -\widehat{\mathbf{f_0}} \\ -\widehat{\mathbf{f_1}} & \widehat{\mathbf{f_0}} & 0 \end{bmatrix}.$$

In this form, $\widehat{\mathbf{f_1}}$ stands for the cross-product matrix

$$\widehat{\mathbf{x}} = \begin{bmatrix} 0 & -x_2 & x_1 \\ x_2 & 0 & -x_0 \\ -x_1 & x_0 & 0 \end{bmatrix}.$$

We can easily observe that $\check{\mathbf{H}}$ is a *self-similar* cross-product matrix. This means that the macro structure of the matrix is the same as the micro structure.

Volume Hessian Eigenvalues

For determining the eigenvalues of $\check{\mathbf{H}}$, we need to examine the two characteristic polynomials

$$\epsilon^3 - \operatorname{tr}(\mathbf{C})\epsilon - 2J = 0 \tag{3.4}$$

$$\epsilon^3 - \operatorname{tr}(\mathbf{C})\epsilon^2 + \frac{1}{2}\left(\operatorname{tr}^2(\mathbf{C}) - \operatorname{tr}(\mathbf{C}^2)\right)\epsilon - \det(\mathbf{C}) = 0 \tag{3.5}$$

where ϵ denotes the eigenvalues of $\check{\mathbf{H}}$ and $\mathbf{C}$ is taken from Table 3.1. Eq. 3.4 is easier to solve and corresponds to the characteristic polynomial of $\mathbf{C}$. Given its roots ϵ_α, ϵ_β, ϵ_γ, the eigenvalues of $\check{\mathbf{H}}$ are: $\pm\sqrt{\epsilon_\alpha}$, $\pm\sqrt{\epsilon_\beta}$, $\pm\sqrt{\epsilon_\gamma}$. Using the singular values of $\mathbf{F}$, the eigenvalues can be written nicely in the following way:

$$\begin{aligned} \epsilon_3 &= \sigma_0 & \epsilon_6 &= -\sigma_0 \\ \epsilon_4 &= \sigma_1 & \epsilon_7 &= -\sigma_1 \\ \epsilon_5 &= \sigma_2 & \epsilon_8 &= -\sigma_2 \end{aligned}$$

The remaining eigenvalues can be obtained by using Eq. 3.5. This equation represents a depressed cube. A depressed cube is a cubic of the form

$$t^3 + pt + q.$$

Eq. 3.5 can be written in this form with $p = -\operatorname{tr}(\mathbf{C})$ and $q = -2$. Using this knowledge, the roots of Eq. 3.5 can be obtained by

$$\epsilon_k = 2\sqrt{\frac{I_C}{3}}\cos\left[\frac{1}{3}\left(\arccos\left(\frac{3J}{I_C}\sqrt{\frac{3}{I_C}}\right) + 2\pi k\right)\right] \quad k = 0, 1, 2.$$

For further reading about how to solve depressed cubic equations, see e.g. *How to Solve a Cubic Equation, Part 4: The 111 Case* [Bli07].

These are all of the eigenvalues of $\check{\mathbf{H}}$. We know that three of the six eigenvalues (ϵ_3, ..., ϵ_8) have to be negative or equal to zero since the root function yields a positive and a negative value. In addition, the cosine function for calculating ϵ_0, ϵ_1 and ϵ_2 ensures that one or two of these eigenvalues are also negative. As we have seen in the previous sections, the other terms of $\frac{\partial^2\psi}{\partial\mathbf{F}^2}$ all only have nonnegative eigenvalues. Thus,

the volume Hessian is the only possible source of negative eigenvalues ([SGK18], 12:6).

In order to investigate the behaviour of the volume Hessian term further, let us look at $J = \det(\mathbf{F})$ a bit more in detail. J is not convex which is problematic, because we will use the Newton method for the optimization process. In order for the Newton method to work correctly, it needs a convex function. Fortunately, the other terms of $\frac{\partial^2 \psi}{\partial \mathbf{F}^2}$ serve as an additional regularization as already stated for the Tikhonov term.

TODO: Regularisation?

Volume Hessian Eigenvectors

$\check{\mathbf{H}}$ can be factorized as $\check{\mathbf{H}} = \check{\mathbf{Q}}\mathbf{\Lambda}\check{\mathbf{H}}^\intercal$ according to its eigendecomposition. The eigenvectors can then be obtained by $\check{\mathbf{Q}}$. In the tensor form signified by $\mathbb{Q}$ each eigenvector is a 3 $\times$3 matrix:

$$\mathbb{Q} = \begin{bmatrix} [\mathbf{Q}_0] & [\mathbf{Q}_3] & [\mathbf{Q}_6] \\ [\mathbf{Q}_1] & [\mathbf{Q}_4] & [\mathbf{Q}_7] \\ [\mathbf{Q}_2] & [\mathbf{Q}_5] & [\mathbf{Q}_8] \end{bmatrix}.$$

The eigenvector corresponding to ϵ_3 can be written as

$$\mathbf{Q}_3 = \frac{1}{\sqrt{2}}\mathbf{U}\mathbf{D}_3\mathbf{V}_\intercal$$

where $\mathbf{U}$ and $\mathbf{V}$ are taken from the SVD of $\mathbf{F}$, $\frac{1}{\sqrt{\acute{\mathrm{L}}}}$

3.3.5 The Complete Eigensystem

With these expressions for each individual term, we can now analyse the complete system, which will be called $\check{\mathbf{A}}$.

$$\check{\mathbf{A}} = \mu\left(1 - \frac{1}{I_C + 1}\right)\mathbf{I} + \mu\frac{2}{(I_C+1)^2}\check{\mathbf{f}}\check{\mathbf{f}}^\intercal + \lambda\check{\mathbf{g}}\check{\mathbf{g}}^\intercal + \lambda(J-\alpha)\check{\mathbf{H}}$$

Deriving the eigenvalues from $\check{\mathbf{A}}$ is nontrivial. Therefore, I will not include the calculations here and will only present the results. An interested reader can have a look at the paper *SNH-FS* on p.12:7 and 12:8.

The final eigenvalues are

$$\epsilon_k = \lambda(J-\alpha)\bar{\epsilon}_k + \mu_T \qquad\qquad k = 0, 1, 2$$

and

$$\begin{aligned}
\epsilon_3 &= \lambda(J-\alpha)\sigma_0 + \mu_T & \epsilon_6 &= -\lambda(J-\alpha)\sigma_0 + \mu_T \\
\epsilon_4 &= \lambda(J-\alpha)\sigma_1 + \mu_T & \epsilon_7 &= -\lambda(J-\alpha)\sigma_1 + \mu_T \\
\epsilon_5 &= \lambda(J-\alpha)\sigma_2 + \mu_T & \epsilon_8 &= -\lambda(J-\alpha)\sigma_2 + \mu_T.
\end{aligned}$$

The variable μ_T is equal to $\mu(1 - \frac{1}{I_C+1})$ and serves as a regularization term. $\bar{\epsilon}_k$ are the roots of the equation

$$\bar{\epsilon}^3 + c_2\bar{\epsilon}^2 + c_1\bar{\epsilon} + c_0 = 0.$$

The variables c_0, c_1 and c_2 are defined as

$$\begin{aligned}
c_2 &= -\|\check{\mathbf{g}}\|_2^2\rho - I_C\eta \\
c_1 &= -(1+2J\rho)I_C - 6J\eta + \left(\|\check{\mathbf{g}}\|_2^2 I_C - 9J^2\right)\rho\eta \\
c_0 &= -(2+3J\rho)J + \left(I_C^2 - 4\|\check{\mathbf{g}}\|_2^2\right)\eta + 2J\left(I_C^2 - 3\|\check{\mathbf{g}}\|_2^2\right)\rho\eta
\end{aligned}$$

with

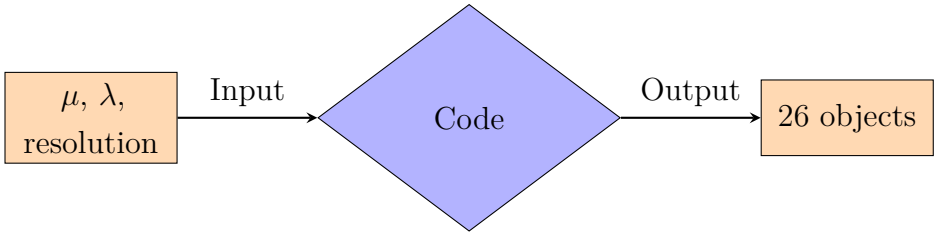
$$\eta = \frac{2\mu}{(I_C+1)^2\left(\lambda(J-1) - \frac{3}{4}\mu\right)} \qquad\qquad \rho = \frac{\lambda}{\lambda(J-1) - \frac{3}{4}\mu}.$$

TODO: Not happy with this section.

3.4 Experiments with the Code

The authors of the paper *SNH-FS* provided the implementation for an application of their formulated energy in C++[1]. The code performs a stretch test on a cube with either a tetrahedral or hexahedral mesh. Their implementation demands a string for defining the directory into which the output files should be saved, a value for each of the two lamé parameters μ and λ and a value for defining the desired resolution as input data.

The algorithm progressively calculates the deformation in 25 steps. The outputs are 26 static objects in the format .obj that show the object in its rest state and then illustrate the 25 steps of deformation.



3.4.1 Example

I will explain how the implementation works based on an example. For this I am taking the input variables $\mu = 1.0$, $\lambda = 10.0$ and a resolution of 10.0. Thus, for the poisson's ratio we get the value

$$\sigma = \frac{10.0}{2(10.0 + 1.0)} = 0.4545.$$

The command to start the execution for a tetrahedral mesh with these input variables is shown in code snippet 3.1.

```
$ ./tetcli 10 stable_neo_hookean 1.0 10.0 output
```

Code snippet 3.1: Bash command for executing the code

[1] http://graphics.pixar.com/library/StableElasticity/snh_code.tar.bz2

The code then creates a cube with 10 edges (corresponds to the chosen resolution) and 11 vertices (calculated by resolution + 1) on each side. The cube therefore consists of 1′331 vertices in total. The amount of hexahedra respectively tetrahedra are calculated by

$$\text{Tet count} = 6 * 10 * 10 * 10 = 6'000$$
$$\text{Cube count} = 10 * 10 * 10 = 1'000.$$

Afterwards the simulation with the 25 steps of the deformation starts. The code sets two faces of the cube as a boundary condition. The stretching is done by modifying these two boundaries further away from each other in the y-axis. This is done by updating the y-value of the vertices with a new value `newNegativeBoundary` or `newPositiveBoundary` depending on whether the vertex should move in the positive or negative direction of the y-axis. The definition of these two values is shown in code snippet 3.2. The variable `stepDelta` is equal to 0.1. It is not a fixed value but modifying it which would influence the extend of the deformation. The variable `stepNum` is the value of the current step from the 25 we are in right now. The position of the remaining vertices are then kinematically scripted.

```
const double newNegativeBoundary = -1.0 - stepNum * ↩
    stepDelta;
const double newPositiveBoundary = 1.0 + stepNum * stepDelta;
```

Code snippet 3.2: Updating y-coordinates of vertices

After the new positions are declared, the `TetNewtonSolver` respectively `CubeNewtonSolver` are used to minimize the strain energy over the meshes. The final mesh is then saved in an .obj file in the directory *output*.

Table 3.4 and 3.5 show, how many Newton iteration had to be done for solving the system. The amount of iterations increase with increasing step size. This is not surprising, since the deformation is increased with each step resulting in more complex calculations.

The images in Fig. 3.4 show four of the resulting objects of this example.

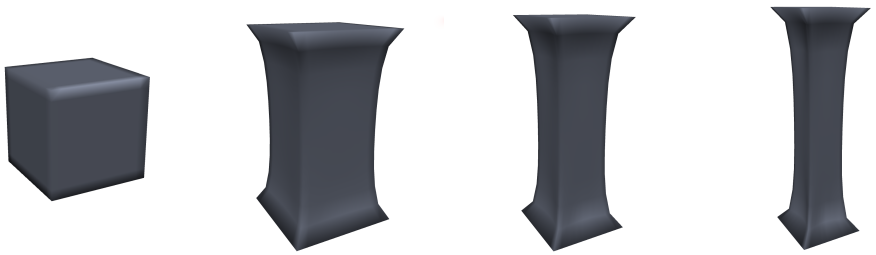
Step size	Iterations
1-10	2
11-21	3
22-25	4

Table 3.4: Newton iterations for a tetrahedral mesh

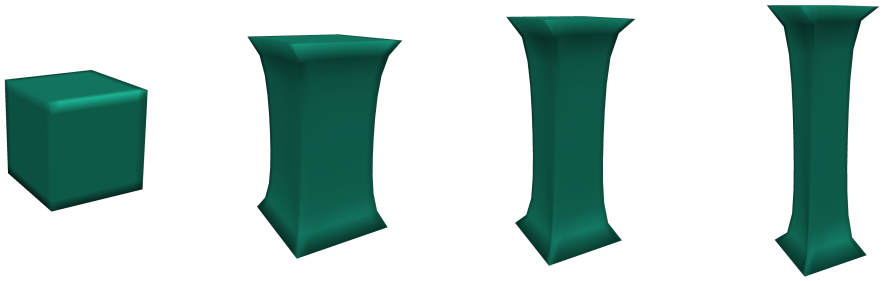
Step size	Iterations
1-7	3
8-14	4
15-18	5
19-22	6
23-25	7

Table 3.5: Newton iterations for a hexahedral mesh

They show step 0 (initial cube), 8, 16 and 24 for a tetrahedral and hexahedral mesh. The result is good, no artefacts can be seen. There is no obvious difference recognizable whether we took a hexahedral or a tetrahedral mesh. All snapshots in this chapter are taken with the help of the software OpenFlipper[2].



(a) Stretch test on a hexahedral mesh



(b) Stretch test on a tetrahedral mesh

Figure 3.4: Stretch test performed on a cube with (a) a hexahedral mesh and (b) a tetrahedral mesh

[2]https://www.graphics.rwth-aachen.de/software/openflipper/

3.4.2 Comparison of different input variables

For examining the influence of the input variable, I experimented with different values. The importance of the resolution is straight forward. For a higher resolution, the cube consists of more vertices and edges. This results in smoother results but also a longer runtime. If we choose a resolution of 30, the cube consists of 29′791 vertices and 162′000 tetrahedra respectively 27′000 hexahedra. Table 3.6 and 3.7 show the completed Newton iterations. The amount of iterations is slightly more than in the example in section 3.4.1.

Step size	**Iterations**
1-7	2
8-13	3
14-20	4
21-25	5

Table 3.6: Newton iterations for a tetrahedral mesh (res=30)

Step size	**Iterations**
1-4	3
5-10	4
11-15	5
16-19	6
20-23	7
24-25	8

Table 3.7: Newton iterations for a hexahedral mesh (res=30)

Fig. 3.5 shows the results with a higher resolution of 30 compared to a lower resolution of 10. All other input variables are the same. The figure shows the results of the deformation step 25. We can see that a higher resolution clearly gives us much nicer results.

(a) Step 25 with a resolution of 10 (b) Step 25 with a resolution of 30

Figure 3.5: Results with a different resolution on a tetrahedral mesh

In addition, we can increase or decrease μ and λ which changes the poisson's ratio. If we decrease λ, the poisson's ratio also decreases. We

can set λ equal to 1.0 and let the other variables stay the same as in the last example ($\mu = 1.0$, resolution = 30.0). Fig. 3.6 shows how the deformation is influenced by changing λ. With these inputs the cube gets wider in the middle part, as the poisson's ratio indicates that the material is more resistant to the stretching.

(a) Step 25 with $\lambda = 10.0$ (b) Step 25 with $\lambda = 1.0$

Figure 3.6: Results with a different values for λ on a hexahedral mesh

Table 3.8 and 3.9 show the amount of Newton iteration needed for the chosen input values. The amount of iterations has decreased for both meshes. We need less iterations because the extend of the deformation is smaller.

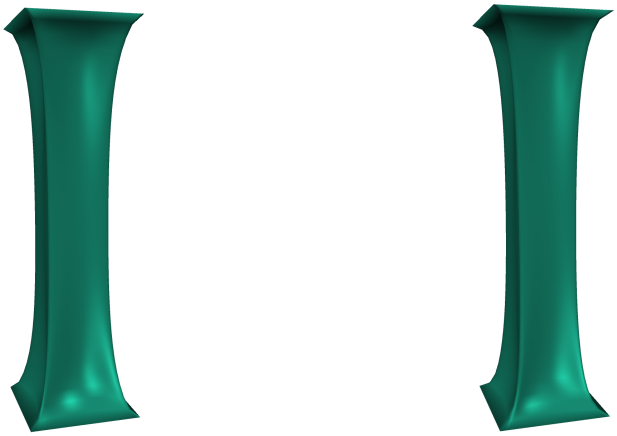
Step size	**Iterations**
1-25	2

Table 3.8: Newton iterations for a tetrahedral mesh ($\lambda = 1.0$)

Step size	**Iterations**
1-25	3

Table 3.9: Newton iterations for a hexahedral mesh ($\lambda = 1.0$)

Now instead of decreasing the poisson's ratio, let's increase it. We can achieve that by choosing $\mu = 0.1$ and letting the other variables stay the same as in the first example of this section ($\lambda = 10.0$, res = 10). This is an extreme case because the poisson's ratio is very close to 0.5. A higher poisson's value makes the cube more susceptible for stretching. We can see this effect a bit by comparing the two images in Fig. 3.7, although the difference is not extreme. The model still behaves well and no artefacts can be seen.



(a) Step 25 with $\mu = 1.0$ (b) Step 25 with $\mu = 0.1$

Figure 3.7: Results with a different values for μ on a tetrahedral mesh

For solving this system, we need more Newton iterations as we can see in Table 3.10 and 3.11. This is caused by the more complex deformation. The cube gets narrower in the middle part which increases the difficulty of reaching a valid solution.

Step size	**Iterations**
1	2
2-3	3
4-6	4
7	5
8-11	6
12	7
13-20	8
21-25	10

Table 3.10: Newton iterations for a tetrahedral mesh ($\mu = 0.1$)

Step size	**Iterations**
1	5
2	6
3	7
4	8
5-6	9
7	10
8-9	11
10-11	12
12	13
13-14	14
15	15
16	16
17-19	17
20-25	19

Table 3.11: Newton iterations for a hexahedral mesh ($\mu = 0.1$)

Extreme values

We can test the energy with some extreme values. For getting a very low poisson's value near 0, I choose $\lambda = 0.1$ and $\mu = 10.0$ with a resolution of 30. This takes about 100 to 200 Newton iterations. The finished result gives us a strange looking deformations and little artefacts as shown in Fig. 3.8. This does not mean that it is a bad model, but rather that is only suited for higher poisson's ratio like it is the case for simulating flesh.

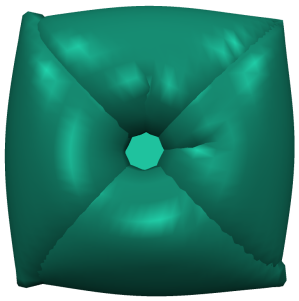


Figure 3.8: Step 4 with $\lambda = 0.1$ and $\mu = 10.0$ on a tetrahedral mesh shown from above

3.5 Discussion

Stuff, Taylor approx.

TODO: Explain difference in iteration with meshes. Explain artefacts.

List of Figures

List of Tables

Code

List of abbreviations

SNH-FS *Stable Neo-Hookean Flesh Simulation* 2–4, 14, 28

SVD *Singular Value Decomposition* 5–7, 9, 15, 24, 27

Bibliography

[Ber15] Jörgen Bergström. "5 - Elasticity/Hyperelasticity". In: *Mechanics of Solid Polymers*. Ed. by Jörgen Bergström. William Andrew Publishing, 2015, pp. 209–307. ISBN: 978-0-323-31150-2. DOI: https://doi.org/10.1016/B978-0-323-31150-2.00005-4. URL: http://www.sciencedirect.com/science/article/pii/B9780323311502000054.

[Bli07] J. F. Blinn. "How to Solve a Cubic Equation, Part 4: The 111 Case". In: *IEEE Computer Graphics and Applications* 27.1 (Jan. 2007), pp. 100–103. ISSN: 1558-1756. DOI: 10.1109/MCG.2007.10.

[Bow09] Allan F Bower. *Applied mechanics of solids*. CRC press, 2009.

[BW97] Javier Bonet and Richard D Wood. *Nonlinear continuum mechanics for finite element analysis*. Cambridge university press, 1997.

[For14] William Ford. *Numerical linear algebra with applications: Using MATLAB*. Academic Press, 2014.

[GV12] H Golub Gene and F Van Loan Charles. *Matrix computations. Vol. 3*. 2012.

[Kor17] Alexander M. Korsunsky. "Chapter 2 - Elastic and Inelastic Deformation and Residual Stress". In: *A Teaching Essay on Residual Stresses and Eigenstrains*. Ed. by Alexander M. Korsunsky. Butterworth-Heinemann, 2017, pp. 5–20. ISBN: 978-0-12-810990-8. DOI: https://doi.org/10.1016/B978-0-12-810990-8.00002-1. URL: http://www.sciencedirect.com/science/article/pii/B9780128109908000021.

[Lak87] Roderic Lakes. "Foam structures with a negative Poisson's ratio". In: *Science* 235 (1987), pp. 1038–1041.

[Lak93] Roderic Lakes. "Advances in negative Poisson's ratio materials". In: *Advanced Materials* 5.4 (1993), pp. 293–296.

[LM15] Joerg Liesen and Volker Mehrmann. *Linear algebra*. 1st ed. 2015. Springer International Publishing, Switzerland 2015: Springer, Cham, 2015. ISBN: 978-3-319-24344-3.

[Moo40] Melvin Mooney. "A theory of large elastic deformation". In: *Journal of applied physics* 11.9 (1940), pp. 582–592.

[MR09] P. H. Mott and C. M. Roland. "Limits to Poisson's ratio in isotropic materials". In: *Phys. Rev. B* 80 (13 Oct. 2009), p. 132104. DOI: 10.1103/PhysRevB.80.132104. URL: https://link.aps.org/doi/10.1103/PhysRevB.80.132104.

[Ogd97] Raymond W Ogden. *Non-linear elastic deformations*. Courier Corporation, 1997.

[Riv48] RS Rivlin. "Large elastic deformations of isotropic materials IV. Further developments of the general theory". In: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 241.835 (1948), pp. 379–397.

[SGK18] Breannan Smith, Fernando De Goes, and Theodore Kim. "Stable Neo-Hookean Flesh Simulation". In: *ACM Trans. Graph.* 37.2 (Mar. 2018), 12:1–12:15. ISSN: 0730-0301. DOI: 10.1145/3180491. URL: http://doi.acm.org/10.1145/3180491.

[Spe80] A. J. M. Spencer. *Continuum Mechanics*. 2004th ed. 31 East 2nd Street, Mineola, N.Y. 11501: Dover Publications, Inc., 1980. ISBN: 0-486-43594-6 (pbk.)

[WY16] Huamin Wang and Yin Yang. "Descent methods for elastic body simulation on the GPU". In: *ACM Transactions on Graphics (TOG)* 35.6 (2016), pp. 1–10.

Online Sources

[Pix] Pixar. *Deformation Map.* URL: https://dl.acm.org/ft_gateway.cfm?id=3180491&ftid=2009597.

Figure Sources

[Pix] Pixar. *Deformation Map.* URL: https://dl.acm.org/ft_gateway.cfm?id=3180491&ftid=2009597.

Erklärung

gemäss Art. 28 Abs. 2 RSL 05

Name/Vorname: .....

Matrikelnummer: .....

Studiengang: ……

Bachelor ☐ Master ☐ Dissertation ☐

Titel der Arbeit: .....

.....

.....

LeiterIn der Arbeit: .....

.....

Ich erkläre hiermit, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe o des Gesetztes vom 5. September 1996 über die Universität zum Entzug des auf Grund dieser Arbeit verliehenen Titels berechtigt ist.

.....
Ort/Datum

.....
Unterschrift