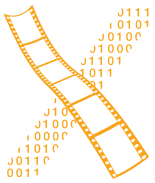

Chair for Computer Science 10 (Media Computing and Human-Computer Interaction)

RWTH AACHEN UNIVERSITY

Evaluating SoRoCAD: Enabling Users to Design Custom Soft Robotics

Master's Thesis
submitted to the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Jakob Strüver

Thesis advisor:
Prof. Dr. Jan Borchers

Second examiner:
Prof. Dr. Jan Bender

Registration date: 20.04.2020
Submission date: 17.10.2020

Zentrales Prüfungsamt/Central Examination Office

Eidesstattliche Versicherung
Statutory Declaration in Lieu of an Oath

Name, Vorname/Last Name, First Name

Matrikelnummer (freiwillige Angabe)
Matriculation No. (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel
I hereby declare in lieu of an oath that I have completed the present paper/Bachelor thesis/Master thesis* entitled

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
independently and without illegitimate assistance from third parties (such as academic ghostwriters). I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

Ort, Datum/City, Date

Unterschrift/Signature

*Nichtzutreffendes bitte streichen

*Please delete as appropriate

Belehrung:
Official Notification:

§ 156 StGB: Falsche Versicherung an Eides Statt
Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
Para. 156 StGB (German Criminal Code): False Statutory Declarations
Whoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.
§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt
(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.
(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.
Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence
(1) If a person commits one of the offences listed in sections 154 through 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.
(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:
I have read and understood the above official notification:

Ort, Datum/City, Date

Unterschrift/Signature

Contents

List of Figures

List of Tables

Abstract

The goal of this thesis is improving the usability and functionality of a simple CAD tool for designing soft pneumatic actuators. To that end we want to integrate a soft-body physics simulation feature into the tool, so that users can see how their actuator performs when inflated, without having to fabricate it. Users should be able to design 3D meshes for both the air channels inside of the actuator, as well as the actuator's silicone exterior. Currently the soft-body physics simulation is provided by a tool called SOFA, which has several useful plugins for soft robotics applications. The focus of this thesis is on finishing the integration of SOFA into our tool, as well as completely redesigning the current workflow inside of the tool. We aim to make working with the tool simple and self-explanatory, by providing a guided workflow that leads users step-by-step through the design process. We also provide an evaluation of the tool with a qualitative study. With the study we aim to assess the effectiveness of the tool and of multiple supportive features that are designed to convey domain knowledge to users.

Überblick

Das Ziel dieser Abschlussarbeit liegt daran ein bestehendes, simples CAD Tool für Soft Robotics zu erweitern und die Benutzbarkeit dessen zu verbessern. Um dies zu erreichen soll ein Soft-Body Physics Simulation Feature in das Tool integriert werden, damit Nutzer das Verhalten ihres Actuators sehen können, bevor sie anfangen, ihn zu fabrizieren. Nutzer sollen in der Lage sein, sowohl das Innenleben als auch das Äußere ihres Actuators als 3D-Mesh zu designen. Die Soft-Body Physics Simulation wird durch ein weiteres Tool namens SOFA bereitgestellt, welches mehrere hilfreiche Plugins for Soft Robotics Anwendungen anbietet. Der Fokus dieser Abschlussarbeit liegt zum einen auf der Integration von SOFA in das SoRoCAD Tool und zum anderen auf einem komplett neuen Design für den Workflow und das User Interface des Tools. Das Ziel ist es, das Arbeiten mit dem Tool so einfach und selbsterklärend wie möglich zu machen, indem wir einen Workflow implementieren, der die Nutzer Schritt für Schritt durch den Design Prozess führt. Außerdem wird das Tool mit einer qualitativen Studie evaluiert. Mit der Study soll die Effektivität des Tools an sich, sowie die Wirkung mehrerer unterstützenden Features beurteilt werden, die dafür eingebaut wurden, um Nutzern möglichst viel Dömanenwissen zu vermitteln.

Acknowledgements

I want to give a big thank you to all the people who participated in the pre-study or in the final study of this thesis.

I also want to thank anybody at the chair who assisted with the thesis either by providing feedback or advice. Special thanks also go to my supervisor Anke Brocker who always gave me helpful advice that improved this thesis immensely.

Conventions

Throughout this thesis we use the following conventions.

Definitions of technical terms or short excursus are set off in coloured boxes.

EXCURSUS:
Excursus are detailed discussions of a particular point in a book, usually in an appendix, or digressions in a written text.

Definition:
Excursus

The whole thesis is written in British English.

Chapter 1

Introduction

The field of Soft Robotics is relatively new, and while there are quite a few artifact contributions that present different soft robots, suitable to different tasks, there are not many specialized, supportive tools available to makers when designing their soft robots. The process of creating soft pneumatic robots includes design work, for which many makers utilize CAD software, molding, which is not a trivial task in many cases, as well as casting and post processing. In this paper we present SoRoCAD, a specialized CAD tool, that is supposed to support makers throughout a large stretch of the creation process.

There already is a variety of CAD tools for general purposes and also plenty of CAD tools for special purposes, as well, but when it comes to soft robotics users are still restricted to general tools like Blender or Fusion360. These tools work fine, but they are not well suited for certain tasks that come up in the fabrication process of actuators, like mold generation to name an example. This is why we have decided to create a specialized CAD tool that explicitly handles domain specific problems.

SoRoCAD should enable users to design their soft robots through constructive solid geometry methods, it should automatically generate molds for users based on the design of their robots and it should provide users with a simulation of their robot's behaviour in order to shorten the Design-

Implement-Analyse cycle for creators. Our hope is that this can reduce the significant barrier of entry that exists for soft robotics.

In this paper, we will first take a look at soft robotics and show some examples of soft pneumatic actuators from literature. We will also look at the fabrication process and at a variety of different molding techniques, employed in literature. To finish up the introduction we will take a look at established tools that are being used in the context of soft robotics. We will also evaluate the tools in terms of if they can be integrated into SoRoCAD to provide specific features.

We will then describe the state of the SoRoCAD tool that it was in at the start of this thesis and provide the results of a pre-study that we conducted in order to gather feedback for a complete redesign of the tool. We will describe the different parts of the new design and why we decided to put them together in the way that we did and to finish it off we will talk about some specific implementation details, namely the mold generation as well as the connection to the SOFA tool, which provides the simulation feature to SoRoCAD.

Finally, we will evaluate the new design with a qualitative user study and analyze the results of the study. Based on the results we will then derive some ideas for the improvement of the tool as future work to finish up the paper.

1.1 Soft Robotics

Soft Robotics is a field dealing with robots that are either partially or completely made of soft materials. Such materials can include, but are not limited to, silicone or other elastomers, as well as textile materials. Parts of a robot can be constructed with soft materials, while the actuators themselves are still made of conventional, hard materials like steel or aluminium.

ACTUATORS:
An Actuator is the part of a robot that creates motion. Traditionally those parts can consist of motors or hydraulic systems.

Definition:
Actuators

Some contributions to the field of Soft Robotics go even further and create robots almost entirely out of soft materials, including large parts of the actuators [1]. When looking at these actuators, there are several kinds of actuation mechanisms that are popularly used.

String-based actuation relies on a string that is embedded into a soft actuator and connected to a motor. The motor can pull on or release the string, causing the actuator to contract or bend, depending on where the string was embedded. These kinds of actuators are simple and easy to understand, but they still require a motor to function. Similarly, the strings can be substituted by shape-memory alloys that change their shape based on their temperature. While these SMA wires don't require a motor to be actuated, they do require a power supply to heat up.

Hydraulic or pneumatic actuation is based on actuators that are usually hollow on the inside, similar to a balloon. A pump can push liquids[2] or gases into the actuator, causing it to inflate. Depending on the properties of the actuator this inflation can cause specific motions. The most common materials to induce a motion are either water or air. Air however is usually the preferred choice, because it is lighter and can be pumped with less effort. When pumping air into an actuator, the time it takes for the air to distribute along the interior of the actuator is negligible, while water distributes slower and weighs down the actuator more. Similarly to string-based actuation, this method requires parts from traditional robotics to function, in this case a pump. This actuation method makes it easier to have multiple actuators relying on a single pump via a system of valves. Such a system however requires dynamic switching of valves, which might necessitate the use of some sort of microcontroller.

Fabricating a soft robot usually includes a molding and casting process, where silicone is poured into a set of molds, which produce multiple silicone parts, that are then glued together. Since molds are usually 3D-printed and since the silicone takes a long time to cure between steps, it can take up to a day to fabricate even a very simple soft robotic actuator.

Chapter 2

Related work

In this chapter we will present some of the related work that is relevant to this thesis. We will start with some interesting applications of soft robotics that we found in the literature. In order to get deeper insights into the established fabrication processes for soft robotics we will then take a closer look at the molding and casting methods that are described in the related literature. We will close this chapter with a deeper look at three different software tool contributions that share some features with SoRoCAD and see how they differ both from SoRoCAD and each other.

2.1 Soft Robotics

Applications of soft robotics are diverse and include, but are not limited to, gripping tasks, user assistance in exoskeletons, as well as movement tasks. Karmakar et al[3] designed and built multiple types of soft robotic grippers, using both hydraulics and SMA approaches, as well as an approach utilizing two electromagnets. They found soft grippers to be very effective in gripping other soft objects.

Polygerinos et al[2] created an exoskeletal glove with soft robotic fingers, that is supposed to aid patients who survived strokes in rehabilitation. They used fiber-reinforced

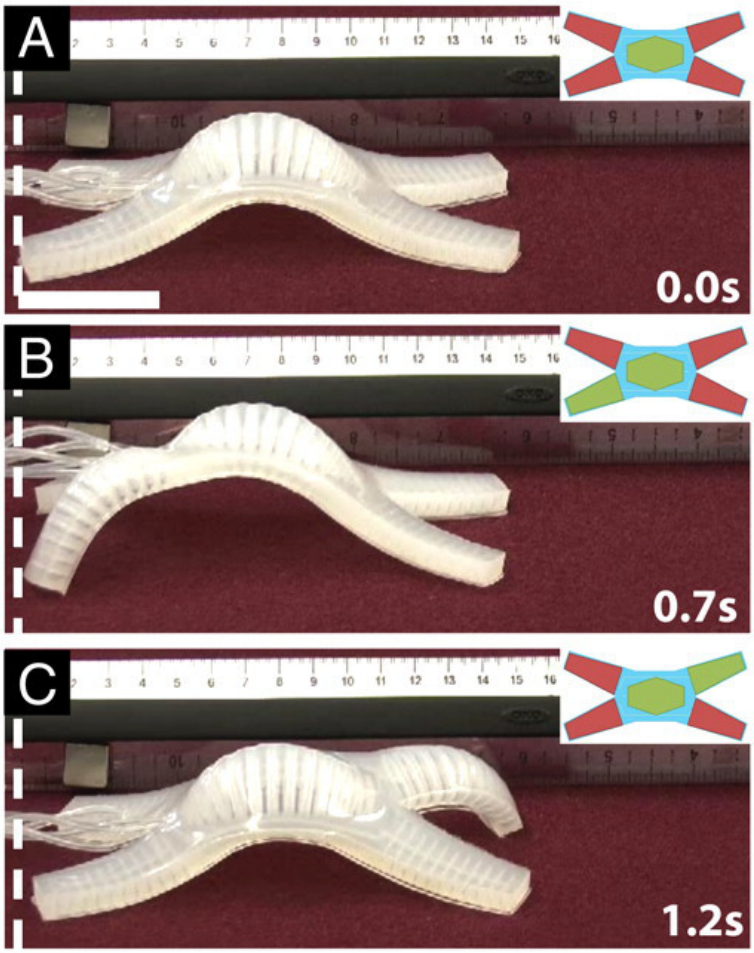

Figure 2.1: A walking, quadrupedal soft robot, that is pneumatically actuated

elastomer actuators combined with less elastic strain limiting layers to create the twisting and bending motions needed to achieve the multiple degrees of freedom they required for finger movement. Notably they used water-based actuation, with a pump as well as a valve controller concealed in several belt pouches.

There is also a variety of research focused on movement tasks and the recreation of different animal movements[4][5]. Shepherd et al[6] created a soft, quadrupedal robot capable of crawling. They designed five air channels, one in each leg as well as an additional one in

the body connecting the legs. By selectively inflating these air channels, their robot can move forward as well as crawl underneath obstacles, all without the need for any internal or external skeleton. For their robot they used soft elastomers and they used air as their actuation material. An example of the robots movement can be seen in Figure 2.1.

Tolley et al[7] created a very similar looking quadrupedal robot, but where the previous iteration relied on an external air supply, their soft robot was completely untethered. The power supply for the pumps as well as the pumps themselves were situated on top of the robot itself, instead of needing to connect the air channels to sources outside of the robot.

Marchese et al[8] built a soft robotic fish with pressured air actuation. The back half of the fish has interior air channels that can be inflated, causing the backside to bend left or right. This action moves the caudal fin of the fish, leading to underwater propulsion very similar to how an actual fish moves in water. Other contributions mimicry the movement behavior of worms[9] or other invertebrate creatures like an octopus[10].

Another example of this is the paper by Lin et al[11] about GoQBot, their soft robot that emulates the behaviour of a special species of caterpillar. The caterpillar is capable of a very unique kind of movement, where in dangerous situations it can roll up into a ring shape and use the momentum of this motion to roll away from danger. Through the use of a soft silicone body, actuated by shape memory alloys, they were able to not only recreate this behaviour but to make the movement extremely fast. Figure 2.2 shows the robots movement compared to the actual movements of the caterpillar.

Onal et al[12] presented a hydraulically actuated silicone robot on wheels, that emulates the movement behavior of a snake. The robot is divided into segments, each with a valve on both sides as well as a pair of wheels to ease the influence of friction on the movement. In order to move forward, individual valves are opened and segment halves are inflated, leading to a slithering, undulating behaviour

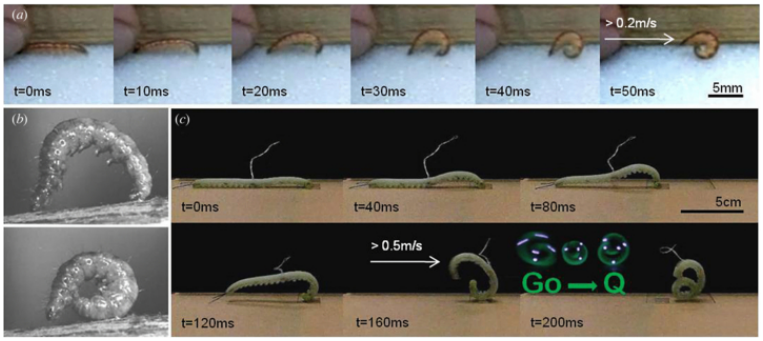

Figure 2.2: A rolling soft robot that emulates the behaviour of a caterpillar

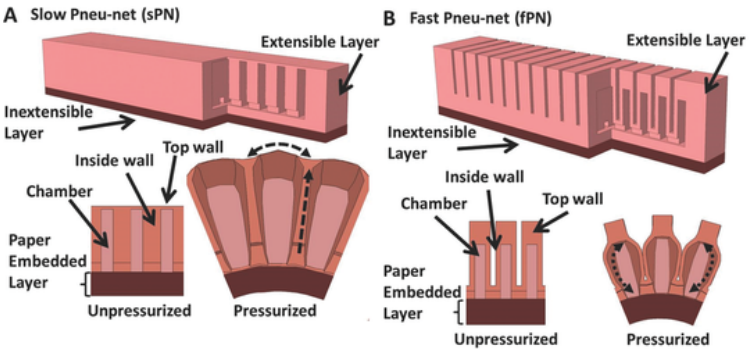

Figure 2.3: Fast Pneu-net construction, that actuate much faster than regular Pneu-nets

that creates forward momentum.

In this paper we focus on hydraulic actuators that are entirely made of silicone. Without the use of additional materials like constraining fibers or less elastic materials, that are embedded into a base layer, the behavior of the actuator is mostly dependent on the size and shape of its air channels, as well as the individual thicknesses of its walls. Having opposing walls of differing thicknesses for example creates a bending motion, as air under pressure will expand into the area where resistance is lowest.

While we focus on actuators that are either cuboids or consist of a series of cuboids strung together, Mosadegh et al [13] showed that soft pneumatic actuators can actuate with

a lot less input pressure when the outer form of the actuator is adapted in a special way. They compared the behaviour of a simple cuboid actuator with an actuator that had an outer form that was adapted to fit the interior air channels, effectively creating a ridge for each air chamber. They found that this kind of actuator is reliable across many different use iterations and that it takes only a fraction of the air pressure for it to actuate in the same way that the simpler actuator would. Needing to pump less air into the actuator for it to move also equates to a faster actuation, compared to the simpler actuator. Figure 2.3 shows the difference between regular pneumatic actuators and their proposed design.

2.2 Molding and Casting

In order to fabricate a soft robotic actuator, the use of molds is required, into which silicone can be poured. Designing molds for this task is not trivial and often requires the use of a CAD tool and a 3D printer or a laser cutter. It is also important to state that fabricating a 3D-Object using molding and casting is very different from fabricating a hollow 3D-object with the same procedures. Creating a defined hollow space inside an object requires that space to be encoded in the mold as a negative. There are several ways to do this and we will describe a few different approaches that are described in literature in this chapter.

You can work with a single mold and create a separate air channel negative that you manually insert inside of the silicone during the casting process. This process is simple but prone to poor precision, and there is also an issue of removing the negative once the object is cured. This might involve destroying the object and gluing it back together or making the negative from a material that is easily extricable in one way or another.

You can also work with multiple partial molds with one or more partial mold encoding the air channel. This process involves some abstract thought and planning and it is not very intuitive. It also requires multiple parts to be 3D-

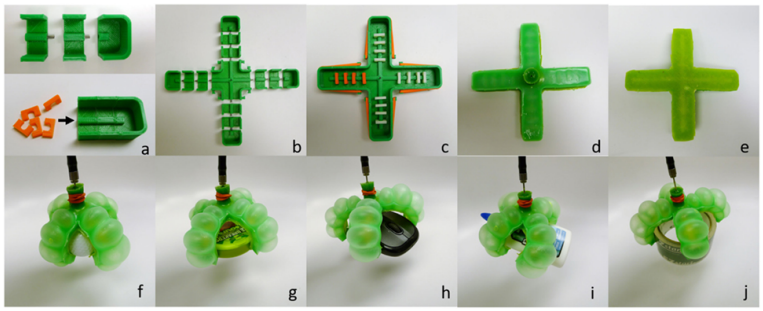

Figure 2.4: A set of modular mold pieces, to let users easily create their own mold and actuator

printed and the multiple partial molds produce multiple partial casts that need to be glued together to assemble the finished object. This process is the process we use for SoRo-CAD and it is explained in more detail in a later chapter.

In the literature there exist quite a few different approaches to the molding and casting process for soft robotics and in this chapter we will take a closer look at a few interesting contributions.

Zhang et al[14] created a set of modules that can be assembled into different mold shapes for soft robotics. Figure 2.4 shows their toolkit of different mold pieces. As creating a silicone exterior with a completely enclosed air channel is not possible, without destroying the mold during fabrication, their fabrication process involves silicone as well as other inelastic materials that serve as a base layer to close up the air channels. While this keeps the fabrication process simple, by only needing one mold, it also requires the use of a different material and some post-processing steps like cutting that material into shape and applying it to the actuator. The shape of the air channels is also somewhat limited to a simple line or to rectangular air chambers connected to a line. The resulting actuators are also limited to simple bending motions. However the outer shape and to a lesser extent the shape of the air channels can be easily customized without the use of any hardware and without the end-user having to use a 3D-printer.

Sun et al[15] evaluate two different types of soft actuators in terms of how much force or torque they can provide dependent on how much pressure they put in and on the wall thickness. One type of actuator they evaluate is a bending soft pneumatic actuator, very similar to what SoRoCAD is supposed to create. They describe a fabrication process very similar to Zhang et al, requiring only one mold that fully encodes the air channel within it. They also close up the actuator with a strain constraining fabric coated in silicone which serves as the inflexible layer. In these kinds of actuators the bending motion is not achieved by differing wall thicknesses but by that inflexible layer that makes inflation on that side much harder.

Finio et al[16] describe a program, where students can build a soft pneumatic gripper. The process just involves casting the actuator in a pre-fabricated mold. However, their fabrication process relies only on silicone as a material. Much like Zhang et al they have only a single mold, in which the air channels are encoded, but contrary to them they close up the actuator with a small layer of silicone instead of an inelastic material. This closing up of the actuator is not achieved through a second mold, but by pouring out silicone over a flat surface and letting it cure. After curing they pour some more uncured silicone on the previous pour and then they put the unclosed actuator on the silicone to close it up. After the last pour of silicone is cured, they only have to cut away any excess silicone. This process is very simple and it requires only one mold but there's also very little control over the wall thickness of the last wall that closes up the actuator.

Moradi et al[17] performed an exploration into working with silicone and provided design tools for creating silicone bladders. In their fabrication process they use some interesting techniques for casting hollow silicone objects. They create molds that have two different fill heights for the the two halves of the bladder, above and below the air channel. They fill the mold with silicone up to the first fill height, and let it cure. Once it is cured, they add a very thin separator on top of it, made of a material called PVA, that they 3D printed, just like the mold itself. They then fill it up with silicone up to the second fill height and let it cure again. Now,

since PVA is water soluble they need to do another post-processing step of filling the air channel with water to wash out the PVA, leaving them with a hollow silicone bladder. This process only requires one mold to be 3D printed, but it requires multiple materials other than PLA and silicone and it requires additional post-processing steps to finish up the actuator. Since the mold and the separator are different 3D-printed parts, they can also reuse molds to make different bladders with the same shape, which can remove the enormous time sink that is 3D-printing large molds.

In the supplemental materials of Mosadegh et al's paper about fast pneumatic networks[13] they provide their molding and casting details. This is interesting because their actuator has a unique outer form that is not just a cuboid. Instead there are empty spaces between the different air chambers, which are only connected at the base of the actuator. While they also use a flexible and an inflexible layer to achieve a bending motion they actually require three molds to fabricate their somewhat complex actuator. The simplest mold is the one for the inflexible layer, that only produces a cast with the shape of a cuboid, in which an inflexible material is embedded. The second mold encodes the shape of the air chambers as a negative and the third encodes the shape of the omissions of the outer form as a negative. The third mold gets filled with silicone and the second mold gets inserted into the silicone from the top, resulting in the characteristic ridges of the fast pneumatic network they describe in their paper. The resulting part is then glued to the base layer to complete the actuator.

2.3 Design Tools

The one thing all of the previously mentioned fabrication workflows have in common is their reliance on molds and on 3D-printing to provide the molds. Most users create those molds with general CAD tools[18], which can be a complicated task. General CAD tools usually don't support mold generation for 3D-objects, so users have to design molds all by themselves. This requires knowledge about how molding works and how the rest of the fabrication

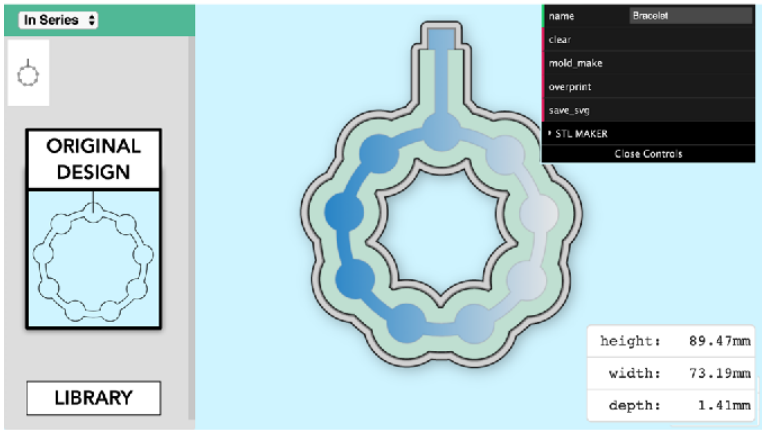

Figure 2.5: SiloSeam Interface that creates an SVG image for mold generation

workflow should work. There are different software tools that are already capable of assisting users through both the design process as well as the mold creation process, that we will take a lot at in this chapter.

Moradi et al[17] provide a workflow for fabricating silicone bladders in their paper, that includes the designing of molds by the user. Their workflow is based on a 2.5D approach, that starts with the silhouette of the bladder provided as a .svg file, that can be created in tools like Photoshop. This .svg file also needs a line for the air channels connection to the air supply. The picture is then loaded into another tool, that they designed, where a bigger border is applied around every object in the picture. The image that results from this process can be seen in Figure 2.5. The user can then increase the height of individual parts of the the picture to create a mold with different fill heights as described in the earlier chapter. Creating a mold with this workflow still requires some mold-specific input from the user, and the degree to which they can customize the bladder is limited to its silhouette.

While Siloseam was mostly about helping users generate molds, there are other tools that assist users in designing soft robots by providing them with a simulation feature.

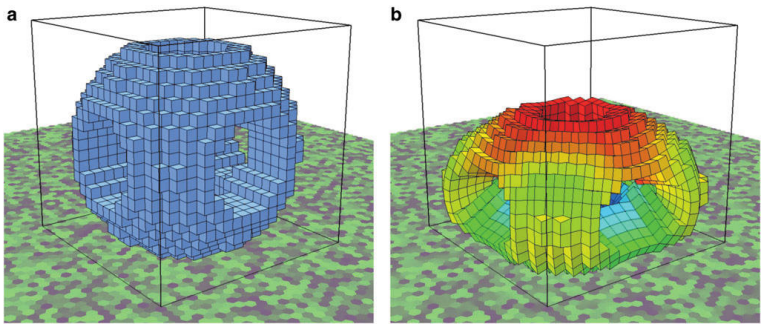

Figure 2.6: Example of a VoxCAD simulation of a soft object

VoxCAD is one such tool, provided by Hiller et al[19]. The tool is based on voxels, which basically means dividing the coordinate space into a discrete 3-dimensional grid. Each discrete position in the grid can be occupied by a single block. VoxCAD offers multiple kinds of blocks with different simulation parameters that the user can set. Such parameters include their stiffness in MPa, their possions ratio as well as their density in Kg/m^3. Now, while users can design soft robots, especially ones that move on actuation, the actuators that they focus on differ from the ones that we deal with in this thesis. The actuators in VoxCAD seem to be temperature actuated, so the individual blocks shrink or expand based on the temperature, which fluctuates during the simulation. Since they are not pneumatically actuated, there is no need for a system of air channels within the actuator, as can be seen in the example in Figure 2.6. And while it is possible to create actuators with VoxCAD that behave very similar to pneumatic actuators, the movements rely on the different properties of the materials, instead of the thickness of the material around an air channel. Also while the design of actuators is possible with the VoxCAD tool, it is not possible to output molds. Even though that is a task, that SoRoCAD could handle, the tool is not suitable for integration into SoRoCAD because of the stark difference in domains.

SOFA or Simulation Open Framework Architecture[20] is a tool that handles general physics simulations with a focus on medical applications. This focus on the medical domain means it is equipped for soft-body physics and de-

formation. It is also an open framework where anyone can write their own plugins to expand the capabilities of the tool. One of those plugins specifically deals with Soft Robotics[21] and how they deform when actuated. The plugin provides functions for string-based actuation as well as for actuation through pneumatic air channels. They also provide some examples with hollow objects, whose stiffness the user can set in a scene file, that get inflated, and they even have an example of a soft pneumatically actuated gripper, which exactly fits the use-case for SoRoCAD. SOFA is a general physics simulation tool, as such it expects a 3D model of an already designed actuator and it outputs a view of the deformed actuator for each time step of 0.01 seconds. Features that it does not provide are actuator design, as well as mold generation, which are the parts of the fabrication process that SoRoCAD can provide on its own. This lead us to ultimately choosing SOFA and its soft robotics plugin as the framework to handle simulations in SoRoCAD.

Chapter 3

SoRoCAD

3.1 Motivation and Goals

Soft Robotics is a field that has a lot of potential for hobbyists and the maker scene, but there are also some significant barriers to entry for newcomers to the field. In order to create an actuator for a task, one needs to have access to sophisticated and expensive machinery like a 3D printer or construct a mold by hand which takes time and precision to do right. For most actuators that you can find in the literature, multiple materials are needed. Silicone or other elastomers are the most common but many of these actuators require some parts with differing elasticity. This is usually realized by using a different silicone with a higher shore value or embedding inelastic materials like constraining strings or paper into or around the actuator, further complicating the fabrication process.

Another barrier to entry is the process of designing molds. A maker has to not only figure out the dimensions and interior design of his actuator, he also has to go through the process of designing one or multiple molds for his actuator, which depending on the complexity of his desired product, can be quite complicated to do. This is also the step where CAD work meets real world complications. Not everything that can be designed in a CAD tool is also

able to be fabricated. While floating interior geometry is not an issue in a virtual tool, it requires significant support structures when being printed, which destroy the purpose of the mold entirely.

Another problem in the process is the long time it takes for a single actuator to be fabricated. Creating the design can take a while depending on the complexity of the actuator, while 3D printing the mold can take multiple hours depending on the size of the mold and the type of printer. Finally with the time spent on waiting for the silicone to cure, as well as gluing the individual parts together, the process can take 1-2 days from idea to finished product. With a single cycle taking this long, it is especially frustrating to do prototyping and to try out different ideas. Mistakes are also especially costly in time when you can only assess the behavior of an actuator once it is entirely finished.

While the duration of the print or the curing of the rubber is hard to be reduced, we can make the process more efficient by assisting users during the mold creation process and by giving them the ability to assess the behavior of actuators at design time. The goal of SoRoCAD is to enable users to design their own actuators, without having to worry about creating a mold on their own. In addition a simulation feature would give users the ability to evaluate the behavior of their actuators without having to go through the entire fabrication process first, saving them both time and material.

In order to reach those goals with the SoRoCAD tool, we need to impose some restrictions on the soft robotics design space. For one, we only look at hydraulic actuators, although the molds that the tool generates would also work with water-based actuation. In terms of the simulation feature we also only look at actuators that are made entirely of the same kind of silicone. If we impose these restrictions, the simulation outcome only depends on a couple of factors: The shore value of the silicone used, the construction of the actuator (i.e. the individual wall thicknesses) and the shape and size of the interior air channels. For this

thesis we decided to iterate on the existing SoRoCAD tool, to make the design workflow more effective for users. To achieve this, a redesign of the UI and workflow of the tool was needed. In order to get feedback on the current state of the tool and on areas of the tool that should be changed we conducted a pre-study. The addition of a simulation feature was another goal of this thesis that was somewhat independent of the UI redesign. In the following chapters we will go through the features of the tool and how they changed throughout the thesis, including the actuator design features, the mold generation and the simulation feature. We will also present the design and results of the pre-study and how the feedback from it impacted the new workflow.

3.2 Previous Design of SoRoCAD

At the start of the work for this thesis, the tool was already capable of actuator design and it already somewhat handled the generation of molds for actuators. The UI at the time was divided into three steps called ”Design”, ”Construct” and ”Refine”, that users could iterate through. The actuator design process was based on Constructive Solid Geometry (CSG), where users could select simple shapes from a predefined library, in order to combine them to form the air channels of their actuator.

CONSTRUCTIVE SOLID GEOMETRY:
Constructive Solid Geometry or CSG is the technique to create complex 3D objects by combining several simpler or smaller 3D objects using boolean operations, like union or subtraction.

Definition: *Constructive Solid Geometry*

The first step, labeled ”Design”, allowed users to select a base shape from the predefined library. Users could then apply transformations to this shape, for example changing the height or width of the shape. The selected shape would then serve as the default shape for the CSG operations in the following steps. However the user could still use different shapes from the library in the later steps, by replacing

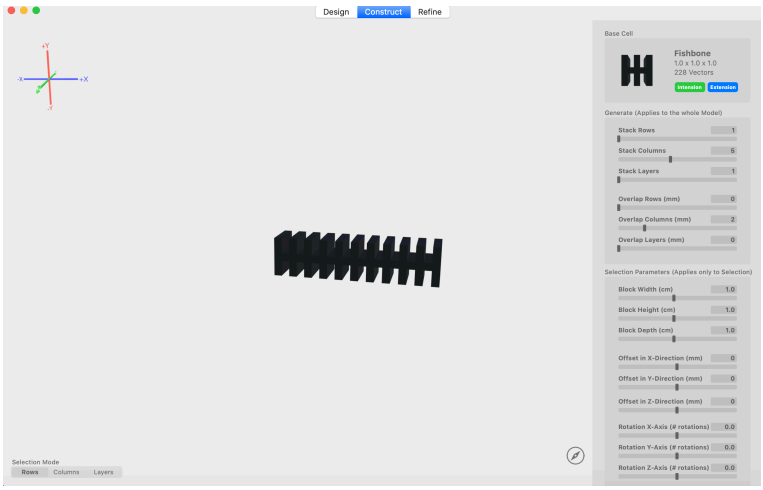

Figure 3.1: The original user interface of SoRoCAD

individual blocks.

The second step, labeled ”Construct”, was where users could stack the base shape in the x, y or z-dimension, creating a bigger 3D shape. Users could also apply transformations like rotations, scaling or translations to entire rows of blocks simultaneously. Through the use of offsets it was also possible for blocks to partially overlap. The idea behind this step was to provide users the opportunity to create a rough draft of their design by giving them the tools to transform several blocks at once. Figure 3.1 shows this workflow step, as well as the general look of the SoRoCAD tool at that iteration of the tool.

The third and final step, labeled ”Refine”, included multiple things. For one, users could apply transformations to individual blocks as well as replace individual blocks with other shapes from the library. The mold generation and export features were also located in this final tab. The idea behind the workflow was, that users could refine their rough draft by manipulating individual blocks in contrast to the entire rows of blocks they could manipulate in the second step. Almost the entire 3-step workflow focused on designing the air channels of the actuator with little focus

on the exterior apart from 3 sliders in the final steps that enabled users to set the wall thicknesses of the actuator exterior. However the sliders grouped multiple walls together, i.e. users could not set either the top or the bottom wall thicknesses individually, which prevented the creation of bending motions.

Before conducting the pre-study we added some features to the existing tool. To put more focus on the exterior of the actuator and to make it more apparent, that the user was mostly designing the air channels of the actuator, a button was added to toggle the display of a partially transparent model of the actuator exterior. This transparent model was laid over the air channels in the "Refine" step and reflected the values of the wall thickness sliders. Another feature that was added was a first version of the simulation feature. To provide simulation capabilities to SoRoCAD, we integrated the SOFA tool, which will be described further in a later chapter.

3.3 Pre-study

In order to evaluate the current state of the tool and to gather feedback and ideas on a potential redesign as well as on future features, a pre-study was conducted. The study consisted of 5 participants, 4 male and 1 female. 3 participants reported prior experience with CAD tools, while 2 reported none or very little prior experience with any CAD tool.

The pre-study was planned as an explorative, qualitative study and as such no quantitative measurements were observered. Participants were given a brief introduction to soft robotics with some examples as well as the presentation of a finished silicone actuator and its molds. They were then given a couple of minutes with the tool to get acquainted with the workflow. After this they were asked a series of about 25 questions in a semi-structured interview. The questions were grouped in a way to orient the interview along the different parts of the tool. There were groups of questions about each of the individual parts of

the workflow as well as about the implementation of the simulation feature at the time. Participants were also asked about how they would envision using such a feature and at which stage of the workflow they would expect it to be.
There were several interesting findings during the course of the study. Participants reported several points of confusion when using the tool, spanning each of the workflow steps, starting with the first tab, labeled "Define". Serving as the first part of the tool that the user sees and interacts with, several users expressed uncertainty when presented with it. While most quickly started fiddling with the sliders and seeing the results of their interaction in the viewport, the purpose of creating a base block was not apparent until switching to the next tab. Depending on the changes they made to the base block, or the lack thereof, some remained confused by the purpose of the first tab until further exploration of the tool and some backtracking.

The second tab, labeled "Construct" seemed to work well to an extent. Participants started to play with the sliders and most understood the purpose of the first tab, once they started playing around with the "Stacking"-sliders, spawning more instances of the base shape into the scene. One participant with CAD experience interestingly expressed that he found it unintuitive to create nodes in the scene via slider input and that he would prefer some kind of button that spawns nodes on click.

Another interesting feedback at this stage revolved around the viewport. While all participants had figured out, that they could control the camera to rotate around the object, one participant remarked that he would prefer to apply transformations to objects via the viewport itself instead of selecting them and using sliders. Viewport interactions like that are common in other CAD tools and 3D tools like Blender. This tab is also where another problem started manifesting. Selection of individual nodes in this tab was not possible for participants. Instead they could select entire rows, columns or layers of nodes by clicking on an individual node. The participants could switch between selection modes via a menu on the bottom left, situated above the viewport. For example if the selection mode "Rows" was selected, then all of the nodes that had the same x-

coordinate to the selected individual node, were selected as well. Changing this selection mode was essential to successfully interacting with this tab, but none of the users interacted with the selection mode switcher, many of them later reporting that they had not even noticed the switcher on the bottom left at all.

The third tab, labeled ”Refine” is where many issues seemed to materialize. Most participants voiced confusion on reaching this final tab. A lot of that confusion seemed to stem from the fact that the ”Refine” tab was very similar in features to the ”Construct” tab. Some participants played with the sliders in this tab and were confused why they were able to apply all of these transformations to nodes again, when they had already done so. Adding to that confusion, some users had created long and thin air channels, where no two nodes had the same x-coordinate. Coupled with them never discovering the selection mode switcher and row selection being the default mode, they only ever selected one node at a time in the ”Construct” tab. For these users, most of the sliders in this final tab were virtually identical to the ones in the previous tab.

This is also where the aforementioned problem of finding UI elements that are situated atop of the viewport occurred again. In order to display the entirety of the actuator by displaying a semi-transparent box on top of the air channels, participants had to click a checkbox labeled ”Show whole actuator” situated at the top right of the viewport, just left of the toolbar on the right. Again most participants were not able to discover this checkbox until they were made aware of it by the conductor of the study. Some of the participants voiced confusion after fiddling with the wall thickness sliders. They expected to see the results of their input in the viewport as they had been able to throughout the tool. However without activating the checkbox, they were unable to do so. The participants had been trained by using the tool to assume that all interactive elements that could cause change in the viewport would be in the toolbar on the right side of the UI. As such they didn’t attempt to search anywhere else for interactivity, which ultimately led to confusion in this step.

Another more general issue was with the sliders and their behavior. Manually entering a value into the text field associated with the slider required a double click. One participant tried multiple times to manually enter a value in the text field by single clicking, later on even giving feedback that he would like to be able to set a value to the slider in that way. Even though he tried to use the feature and came very close to finding it, the barrier of double clicking the text field prevented him from using it. Another user enthusiastically tried out the sliders and after finding out that he could double click the text field, started entering all kinds of values into it. He was however confused by the selection color, which in this case was red. He had assumed that, because the text field became red upon selection, that he had entered an illegal value into the field and that the color was there to alert him of that fact, which was not the intent of using that color at all.

The biggest problem that became apparent by the ”Refine” tab however, was that participants seemed confused about what exactly the 3D object, that they were creating, was supposed to be. Several participants referred to the object they made as a ”soft robot” or as an ”actuator” when in fact they were merely designing the air channels of an actuator. When participants were made aware of the ”Show whole actuator” checkbox and the corresponding visualization of the rest of the actuator, most of them expressed surprise or had a moment of sudden insight when they realized what they had been creating. This lead some of them to go back to the other tabs, inspecting them again in the newfound context they had now acquired. When asked about the simulation and where they would expect such a feature in the workflow, most participants answered that they would like to design and test their actuators in an iterative way, making small changes to the actuator and evaluating the results of these changes via simulation. This exploratory way of using the tool could also help users understand the effects of certain parameters, like wall thicknesses, on the behaviour of the entire actuator.

Also while at that time, the simulation returned only an image of the inflated actuator after a specified amount of timesteps, most users said they would prefer to see results

in an animated way, if at all possible. Presenting the results as an image also puts further restrictions on interaction, like users only being able to view one side of the actuator and not being able to zoom or rotate the camera. Users were also unable to see the process of inflation as they were only presented with the final state of the inflation process.

Any redesign of the tool would have to solve this problem first and foremost, by offering clearer visualizations, that are easier to access. The workflow would need to undergo changes as well, to accommodate the placements of these visualizations. To help users understand the different components of an actuator, namely its air channels and its silicone walls, the tool would have to put focus on it.

3.4 SoRoCAD Redesign

Based on the feedback we received from the pre-study, we decided on a variety of changes for the tool. The most significant of these changes was the complete redesign of the workflow and user interface. The previous workflow went from designing a single block, to stacking blocks in 3 possible dimensions and transforming groups of them, to finally transforming individual blocks. This entire workflow was focused on designing an air channel for an actuator, with the silicone walls as well as the resulting molds receiving less focus.

There was still a problem apart from the workflow, that had to be addressed. The space of designable actuators was quite large and didn't fully overlap with the space of actuators that we can generate molds for with the currently used algorithm. Due to being able to stack air channel blocks in all 3 dimensions, users were able to create a wide variety of objects that might have very complex air channels with a lot of verticality, which makes them harder and harder to be fabricated by a simple generalized process, using one or two molds.

Another issue was the use of 3D-translation when dealing with air channel blocks. Users could set offsets between

blocks that could be both negative or positive, meaning that blocks could either partially overlap or have empty space between them. While partial overlap is a standard technique used in Constructive Solid Geometry, this can be confusing for novice users. Also a positive offset meant that users could create multiple air channels in the same actuator, which can be useful for certain use-cases but is a pretty advanced concept.

Because of these issues it was easy for users to add a lot of complexity to their actuators very quickly, leaving them confused and overwhelming novice users. To remedy this, the decision was made to limit the design space somewhat, by focusing on a certain type of actuator. Users can no longer stack blocks in each direction, but only along the x-Axis, leading to tentacle-like actuators. Actuators like theses are commonly used in grippers and are capable of multiple types of motions, like bending or elongation. 3D-translation was also removed as an option to further reinforce the ”building blocks”-metaphor. While it is still possible to create gaps between air channel blocks through the use of rotation, it is now at least harder to arrive in such a state.

The first question when designing a new workflow was ”who are the users for this tool?”. We decided to gear the tool for novice users, people who have little to no experience with soft robotics and who either want to learn the basics or already have a simple idea that they want to try out. In order to support such a user, the tool had to be simple and encode a lot of domain knowledge in the UI in the form of helpful tips or by handling complicated, automatable tasks such as mold generation for the user.

The transitions between workflow steps should be small and easy to comprehend and each step should correspond to a specific, physical part of the actuator or the mold. As such we decided that the first step should be designing the air channels. This still works by stringing together different kinds of base shapes, that can be scaled or rotated. However, as stated previously, they can only be stacked along the X-axis to limit the space of designable actuators. The user interface for this step can be seen in Figure 3.2.

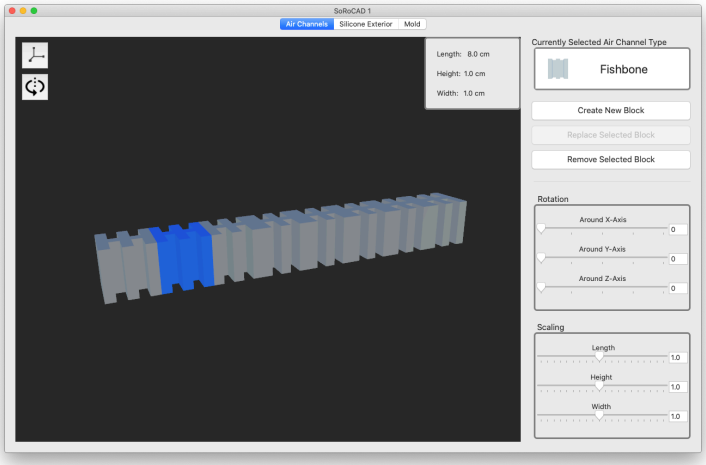

Figure 3.2: The first tab of the revised workflow, where users can design air channels

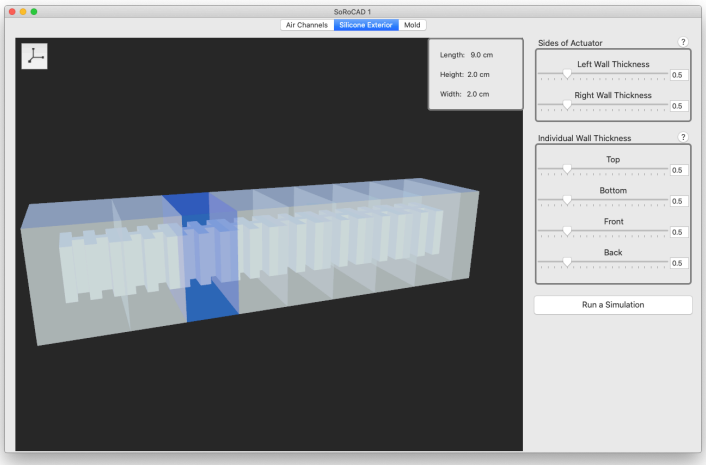

Figure 3.3: The second tab of the revised workflow, where users can design the exterior of their actuator

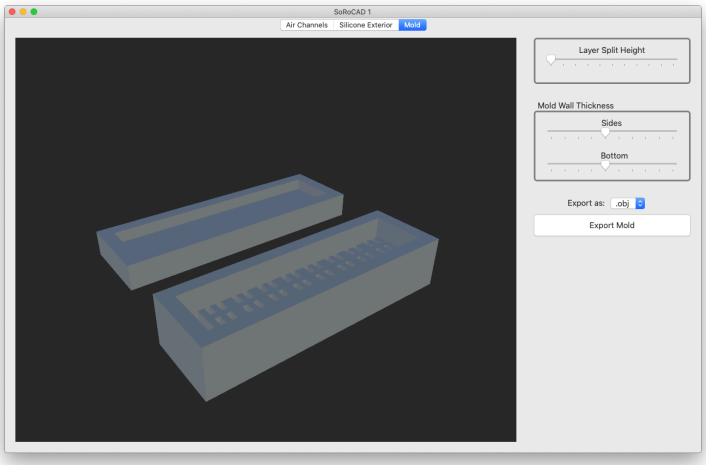

Figure 3.4: The third tab of the revised workflow, where users can view and export molds

The second step handles the exterior of the actuator, in other words everything that will be cast in silicone later. This means four individual wall thicknesses for each air channel block, at the top, bottom, front and back of the block, as well as the thicknesses of the left and right end of the actuator. This step also features a button to start a simulation of the constructed actuator. Since the wall thicknesses are the biggest factor in the behaviour of an actuator, the simulation interface was put here to enable an iterative design process. Users can run a simulation, check the results, and then quickly make changes to the actuator based on the results, without having to change the tab. An image of this workflow step can be seen in Figure 3.3

The third and final step is not as interactive as the previous steps and concerns the mold. A two-part mold is generated and displayed to the user through the viewport. There are some minuscule modifications of the mold that the user can manage here, like the thicknesses of the mold for fabrication or the height, at which the mold is split between the two layers. The latter slightly increases the space of actuators we can generate molds for, which will be more thoroughly discussed in a later chapter. This step of the work-

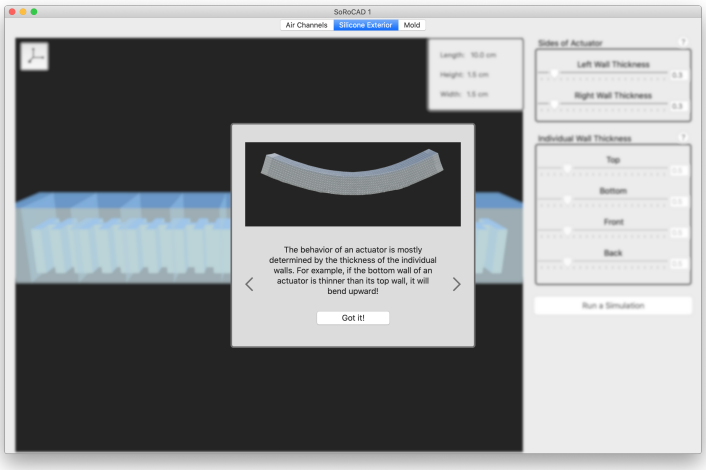

Figure 3.5: The help popup that shows whenever the user enters the second tab for the first time

flow also features the file export options as seen in Figure 3.4.

Through this 3-step workflow we already convey domain knowledge to the user, by showing them the important parts of an actuator and as well as the parameters, that they can set for each of these parts, namely the shape of the air channel and the wall thicknesses around the air channel. We also handle the mold generation process for the user, which, although it only works under certain assumptions, enables the user to focus on the actuator and not on abstract fabrication details.

In order to provide more assistance to users, a popup was also implemented that displays the first time the user opens the second tab. It contains information about how to make an actuator enlarge, bend or elongate, presented both through text and an animated GIF, that displays the motion, as can be seen in Figure 3.5. Users can reopen this popup after closing it, by clicking on ?-Buttons that are positioned throughout the second tab.

An early idea to help novice users was to reverse the workflow. Instead of designing an actuator and then seeing

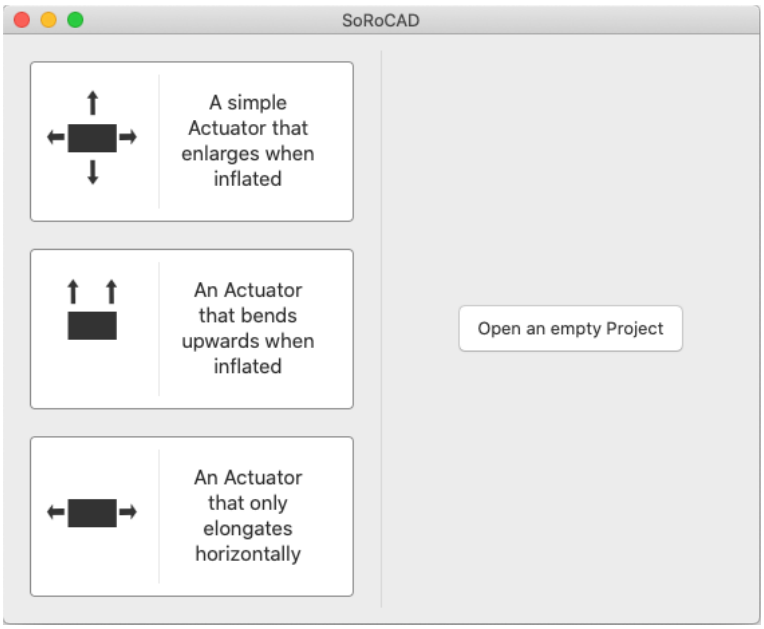

Figure 3.6: Template Selection Window where users can select a template or start with an empty project

how it behaves, users would only give size measurements and the desired behavior as parameters to the tool, which would then generate an actuator based on their requirements. For example a user could specify a 10x2x2cm actuator that bends upwards by 90 degrees when inflated.

This idea evolved into a template system for the tool. We created a small library of size-adjustable prebuilts, that users can utilize. These prebuilts can be accessed via a window that opens up whenever the tool is started. As is portrayed in Figure 3.6 users have the choice between selecting a specific template or just opening an empty project. Using a template basically pre-populates the tool with data and instantly sends the users to the ”Silicone Exterior” tab, where the shape of the air channel as well as any wall thicknesses are already set by the tool. This feature can trivialize the creation of simple actuators, as well as serve as a starting point for more complex designs. In addition to the help pop-up this feature can also serve to further the understanding of the effect of wall thicknesses on the behavior as users can check out concrete examples of actuators that be-

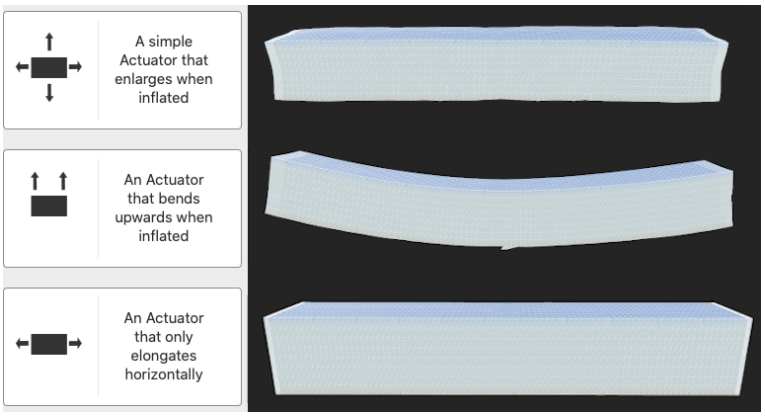

Figure 3.7: The different templates with their corresponding simulation behaviour

have in a certain way.

At the moment we offer three different templates to users, that can be seen in Figure 3.7. The first template labeled ”Enlargement” has thin walls all around, leading to expansion in every direction on actuation. The second template labeled ”Bending” has thicker walls and a very thin wall on the bottom, leading to a upward bending motion. The third and final template labeled ”Elongation” has thick walls all around expect for the ends of the actuator, leading to a small elongation.

However both the help pop-up as well as the template library are mostly front-loaded tutorials for users. While these sorts of tutorials can be helpful for users, they can ignore the templates and click away the tips. Because of this, it would be helpful to give feedback to users, while they use the tool. This was implemented using a variety of small GIFs, depicting various kinds of movements. Every time the user changes a wall thickness, the tool checks the left and right half of the actuator, compares wall thicknesses, determines the dominant behavior and displays the fitting GIF for that half of the actuator. This behaviour can be seen in Figure 3.8, where the GIF is marked in red. For example, if all wall thicknesses on the left half of the actuator are the same, then the ”Enlargement” GIF plays for the left half. If the top wall thickness on the right side is smaller

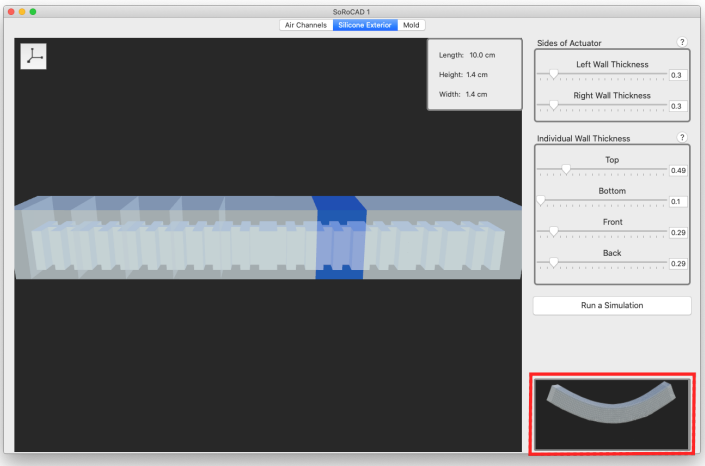

Figure 3.8: A GIF that pops up whenever the user changes a design parameter. The GIF is marked in red for this image

than any other wall thickness, then the ”Downward Bending” GIF for the right side is displayed.

These GIFs are meant to give a broad approximation of the actuator’s behavior and they are only somewhat accurate for the most simple of actuators. Doing some sort of live simulation would be preferable to this solution but as the simulation itself is computationally somewhat expensive and takes 10-15 seconds to finish, it doesn’t seem feasible to implement such a feature for now.

Another feature that was requested in the pre-study was viewport interaction, i.e. the ability to manipulate the parameters of the actuator through the viewport instead of through sliders or textfields. This is a standard feature among many 3D-tools like Blender, and as such there is no need to reinvent the wheel here. The idea was to mirror any interaction that is possible through the use of sliders, to the viewport. That means the scaling and the rotation of air channel segments, as well as the scaling of any wall thicknesses. While the scaling is provided by 3D-handles along each axis that users can pull to scale the object up or down along the corresponding axis, the rotation is provided by 3 rings around the object, that can be pulled to

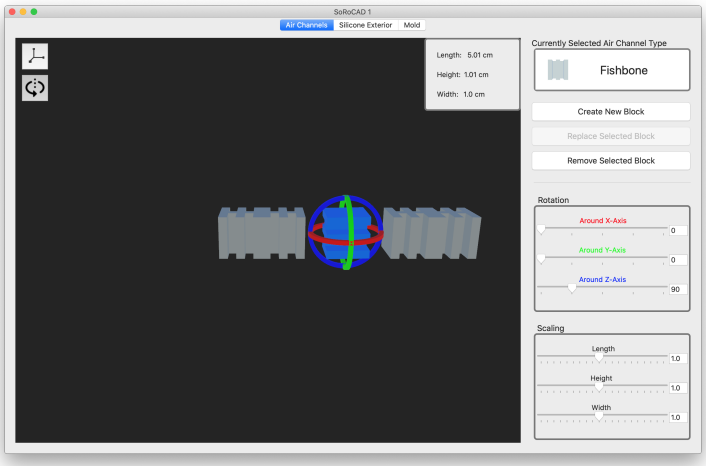

Figure 3.9: Rotation of an air channel segment via the viewport

rotate it around the corresponding axis. One example for viewport interaction is provided in Figure 3.9, where an air channel segment was rotated using the viewport controls.

There were however some problems that needed to be solved for viewport interaction with air channel segments. Since air channel segments are arranged in a straightforward fashion along the X-axis, there is no space between them. You can however scale them along their X-axis, which isn't a problem when it is done through the use of a slider. But when you want to scale an object along its X-Axis through the viewport directly, there is no space to put the scaling handle, except for maybe the first and the last segment in the air channel. This becomes even more of a problem for rotating air channel segments, because rings are put all around the object as handles. Both of these issues are solved by "popping out" the selected air channel segment, as can be seen in Figure 3.9. By adding spaces between the selected segment and any of their neighboring segments, enough space is created to fit the necessary handles, whenever they are needed. This issue doesn't come up in the "Silicone Exterior"-tab, because the only time you need to scale in the x-direction is one the very ends of the actuator, when setting the left and right wall thickness.

The ”Silicone Exterior”-tab also presented some unique challenges, because there are multiple ways to present it to the user. You could view the setting of wall thicknesses as a translation on the air channels. In this case, you just move the air channel around within the confines of the silicone actuator to indirectly set wall thicknesses. For example if you have an actuator with a top and bottom wall thickness of 0.5cm, you could move the air channels up by 0.25cm to create an actuator with a top wall thickness of 0.25cm and a bottom wall thickness of 0.75cm. This works exactly the same way in all of the other directions as well. This also requires the overall size of the actuator to be a separate set of parameters. So a user would set the size of the actuator and then move the air channels to the desired position within these constraints. For the sake of referencing, we call this way of doing things ”Translation-based”-interaction.

Another way to present the ”Silicone Exterior”-tab would be to fix the air channel in place and let the user scale the wall thicknesses directly. For example if you have an actuator with a top and bottom thickness of 0.5cm, you could scale the top wall thickness down to 0.25cm and the bottom wall thickness up to 0.75cm. This would achieve the exact same result as before, but you could also make any of the walls bigger to increase the overall size of the actuator. For example the height of the actuator would add up to the sum of the top wall thickness, the bottom wall thickness and the height of the air channels. We call this way of doing things ”Scaling-based”-interaction.

While scaling-based interaction is more direct and intuitive, it also requires more mental work from a user, because they have to plan out their wall thicknesses to achieve a desired size for their actuator. In translation-based interaction, the user can set the size as a separate set of parameters, but it’s also less intuitive to move the air channel within the actuator, especially since with each movement at least 2 parameters change. For example when you move the air channel along the y-Axis, both the top wall thickness and the bottom wall thickness have to change in accordance with the size constraints.

Both of these interaction methods can produce the same set

of actuators under the assumption that the transformations apply to the whole actuator. However, with that assumption you can only create simple actuators that can do one kind of motion at a time. So for example you can create an actuator that bends upwards but not an actuator that bends upwards for one half and downwards for the other half. To create more complicated actuators we need to be able to transform individual blocks. But this is also where the two interaction methods stop being equivalent in terms of the set of actuators they can create.

When we look at actuators that are capable of mixed motions, for example an actuator that bends upwards on the left side and downwards on the right side, then the capabilities of the two interaction methods are quite different. Bending motions are created by a difference in opposing wall thicknesses, so to do an upward bending motion, the bottom wall thickness has to be lower than the top wall thickness and vice versa for a downward bending motion. While both interaction methods can create an actuator with this behavior, they will look different. In scaling-based interaction, the air channel is fixed, which means every single air channel segment is fixed in place as well. So having a large bottom wall thickness on one half and a small bottom wall thickness on the other half, will create a step in the middle. The finished actuator would consist of two fused cuboids that are offset along the y-direction. In translation-based interaction you move the individual air channel segments instead, so the resulting actuator can look like a simple cuboid. This is not possible with scaling-based interaction, but it comes with some drawbacks as well.

As already discussed before, it seems less intuitive to move the air channels than to scale wall thicknesses, but there is also the issue of connection of air channels. Any rotation or translation of individual air channel segments carries the risk of inadvertently creating two separate air channels that are no longer connected, but are instead separated by silicone. While the simulation can handle this case just fine and would simply inflate both air channels, in reality you would need multiple actuation points for the air to enter into each air channel. To prevent this from happening, it would be necessary to automatically check each air channel

segment to make sure they are touching their neighboring segments. On any failed check, the user would have to be notified about which segments are causing the problem, in order to fix it. While such a system would be possible and not hard to implement, it would cause more complexity on the user interaction side.

In the tool we started with a global-only translation-based interaction, but based on early feedback, we switched to a scaling-based interaction that only works on individual blocks. The hope was, that this would be a good mix of possible actuator complexity and ease of use. Even though all the disadvantages, that were discussed earlier, obviously still apply to this approach, it could be reasonable to sacrifice some possible complexity, when we assume that our users are people which are new to Soft Robotics anyway, who would benefit from more intuitive interaction techniques.

3.5 Mold Generation

The generation of Molds for the third tab of the tool is a topic that seems simple at first but has quite a few pitfalls on closer inspection. Creating an actuator from an air channel is the first step and can be seen as a boolean subtraction in Constructive Solid Geometry. You take a cuboid that has the desired size and shape and you subtract the air channel from it, leaving a hollow space inside the actuator in the shape of the air channel. Mold generation for such a 3D-Object can be seen as just another subtraction. You take a slightly larger cuboid and subtract the actuator from it, which means whatever was solid before is now hollow and vice versa, leading to what is essentially a hollow box with a solid air channel on the inside. This method works fine in a virtual setting but issues start to arise when we translate this into the real world, starting with the process of 3D printing an object.

While floating geometry, like the solid air channel in our mold, is entirely possible in a virtual environment, it is impossible to print without supports. Having supports inside

of the mold makes them unusable for casting. Either you keep the supports, which leads to holes in the actuator, or you remove them after printing, which causes the solid air channel to fall down. Potentially, one could fill the mold up halfway, place the solid air channel inside and then fill up the rest, but that method hardly leads to a precisely fabricated actuator. It follows from this, that we need at least two separate molds to cast a hollow 3D-Object like a soft actuator.

When casting a single object with two molds, two separate components need to be glued together after casting to form the finished actuator. Molding methods that involve two molds are often found in the literature, sometimes referring to the two components as the ”base layer” and the ”flexible layer”, because they achieved a bending motion by either casting these two layers in different materials or by embedding inflexible materials like paper into the base layer.

The first molding approach we used, was based on this approach found in literature. Since the solid air channel needs to be on the bottom of the mold in order to accurately set it in place, the top of the air channel could be the height at which we cut the 3D-Object into two pieces. So we take the actuator from the previous example and we cut it into two pieces along the Y-Axis. The top piece ends right above the start of the air channels and the bottom piece starts at the exact same point. We then take two cuboids that are slightly larger than the actuator in the X and Z direction and slightly larger than their corresponding piece of the actuator in the Y direction. The top of the upper cuboid lines up with the top of its corresponding actuator piece, and the bottom of the cuboid lies a bit below the piece. The lower cuboid is the same but mirrored, so its top lies a bit above its corresponding piece and the bottom lines up with it.

Then we subtract each actuator piece from its corresponding cuboid and receive our two molds. The top mold is much smaller and the solid air channel lies directly on the bottom of the lower mold. This method works for a subset of possible actuators, but there are still some cases where it produces unusable molds, depending on the shape of the air channel. For example if you use a fishbone shape as an

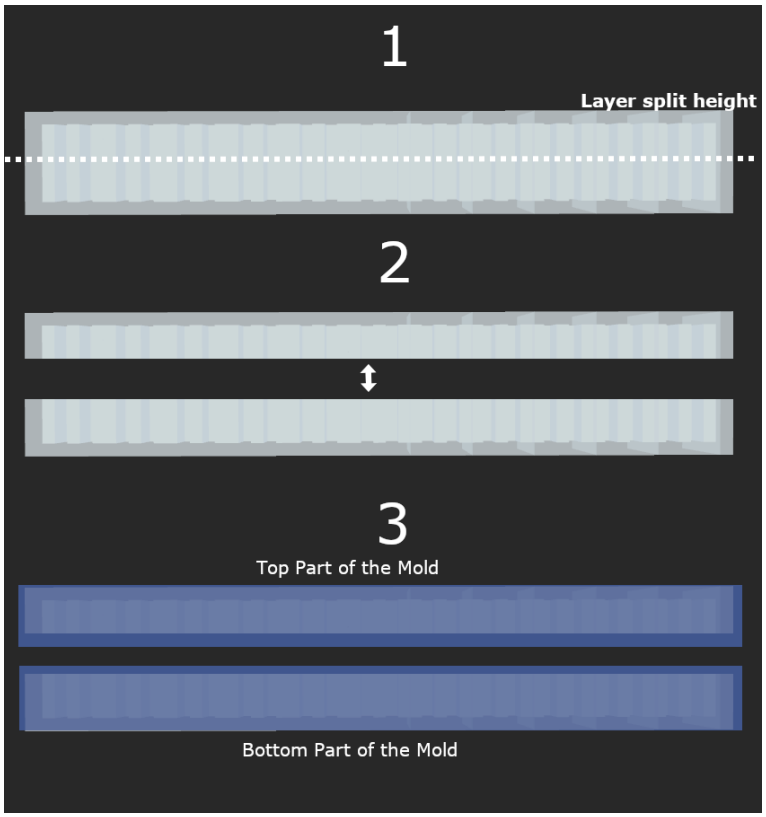

Figure 3.10: Visual explanation of the process for mold generation used in SoRoCAD

air channel segment and you rotate it in a certain way, effectively creating little ”tunnels” on the bottom of the mold, the mold can be fabricatable, potentially even without supports, but you can not remove the silicone from the mold after casting without breaking the silicone that went under the ”tunnels”.

To solve issues like this, we added a slider to the final tab of the tool, that lets the user decide at which height to split the actuator apart. This way, any issues with the mold can be solved by the user. Figure 3.10 shows a visual represantation of this process. The blue boxes in the third step represent the final cuboids that are placed around each actuator part to create the molds. This process however only works with the current scaling-based interaction for actuator designing. If translation of individual air channel segments

were possible, and air channel segments could have differing Y-coordinates, there could be cases, where the new slider would not help.

This method for mold generation was sufficient for the global-only translation-based interaction we had implemented before, but since we switched to scaling-based interaction where you can interact with individual air channel segments, another big issue became apparent. Since it was now possible to increase and decrease wall thicknesses around individual air channel segments, users could create step-like forms. Actuator were no longer only perfect cuboids but they could look like a series of different-sized cuboids glued together.

This created an issue with mold generation, because, in regards to the implementation described earlier, the upper and lower walls of the actuator correspond to the upper part of the respective molds. There was no issue with this before, because those upper and lower walls were always perfectly flat, and the pouring process resulted in silicone that was level. But now that those walls consist of different segments with differing heights, we need to make sure that those wall thicknesses can be properly fabricated.

The current mold generation algorithm already produces a solution, that works in theory. It simply creates 3D-printable walls on top of the mold to make sure the actuator has the desired form. But this is not a great solution in practice for multiple reasons. First off, these extra walls might need supports to print, which would again destroy the whole point of having a mold. And even if the printing process would work without supports, the extra walls would make it hard to extract the actuator from the mold without breaking the mold or damaging the actuator in the process.

As future work, this issue could be addressed by expanding the algorithm to produce three molds instead of two. To do this, one would have the split the actuator into three parts along the y-axis. The upper part starts right above the air channel, the lower part starts right below the air channel and the middle part is the air channel. Although because of

the possible issues with fabricating hollow objects with air channels that were described earlier, there might be cases where the air channel requires two molds by itself. This would bring the total up to four molds for a single actuator, which is a lot of effort for just one actuator. It might be easier to switch to a translation-based interaction or to put some constraints on the outer actuator shapes to address this problem.

3.6 Simulation/SOFA

The addition of a simulation feature to the SoRoCAD tool was a big priority right from the start. The SOFA framework and the corresponding Soft Robotics Plugin were quickly identified as potential candidates to provide this feature. However, in order to use SOFA features in SoRoCAD, there would have to be some connection between the two tools. Since SOFA is written in C++ and SoRoCAD is written in Swift, fully integrating SOFA would take considerable effort. A simpler solution would be to start SOFA through the command line, giving over parameters to the simulation via command line arguments.

As described in an earlier chapter, SOFA is a standalone tool all by itself, with its own Graphical User Interface. However, by providing the command line argument "-g batch" to SOFA on startup, we can start it without a GUI. Now, to run a simulation, several inputs have to be provided to SOFA, including the number of timesteps of simulation that the SOFA tool is supposed to perform, two meshes that describe the actuator and its air channel as well as a python file that describes the scene, where we can set several parameters.

The python file describes a scene for the simulation and places the actuator into that scene. It defines how strong gravity affects it and which coordinate spaces are exempt from any forces. This file is also where file paths to the actuator meshes are placed, so that the SOFA tool can find them. This is also where simulation exporting happens. In an earlier implementation, SOFA was instructed to output

images of the current state of the actuator after a specified amount of timesteps, but in the current build it outputs the full mesh of the actuator at each timestep as an obj file. Some simulation parameters like the amount of pressure or the young modulus of the material are also set in this tool.

YOUNG MODULUS:
The Young Modulus is a mechanical property that describes the stiffness of a material, very similar to a Shore Value. The user sets a Shore Value in SoRoCAD and the tool converts it to a Young Modulus value to pass on to SOFA.

Definition:
Young Modulus

For the FEM Simulation, SOFA needs several meshes of different kinds that represent parts of the actuator. The first mesh that is required is a volumetric mesh of the actuator that is hollow on the inside, just like a finished, fabricated actuator would be. The other required mesh is a surface mesh of the air channel. In SoRoCAD we only work with surface meshes, so providing the air channel mesh is a simple matter of exporting it from the tool and providing the corresponding file path to SOFA.

For the first required mesh, we already have a surface mesh of the finished actuator, that we use for mold generation. The only thing we need to do, is to convert this mesh to a volumetric mesh. This conversion is not trivial and thus we need a framework or plugin to handle this. The Computional Geometry Algorithms Library or CGAL for short, is a C++ library that can handle multiple geometry related tasks. Again, since SoRoCAD is written in Swift, it is not trivial to directly integrate this library. However, there exists a SOFA plugin that enables us to use some CGAL features directly from SOFA. We use this plugin in the python scene file to convert the provided surface mesh to a volumetric mesh, that the simulation can then use.

Definition:
Volumetric Mesh

VOLUMETRIC MESH:
Regular surface meshes only describe the shape of an object through its surface, but some simulation methods require a description of the inner parts of an object. A volumetric mesh describes an object not be a network of polygons that make up its surface, but as a set of tetrahedrons that make up its volume.

We pipe the required meshes to SOFA, convert the surface mesh of the actuator to a volumetric mesh using the CGAL plugin. The simulation is then executed and the resulting mesh is output as .obj files at each simulation time step. But the way the output is handled causes some issues. We have two options when it comes to creating the meshes that we pipe back to SoRoCAD. We can either let SOFA generate a surface mesh from the volumetric mesh used in the simulation, or we can map the changes of the volumetric mesh back to the originally provided surface mesh. Generating a new surface mesh from the deformed volumetric mesh seems like the straightforward solution, but the surface mesh that is produced in this way by SOFA often looks bumpy because it is created from a volumetric mesh made up of individual tetrahedrons, often with space in between them. However, mapping the changes back to the original surface mesh brings its own challenges, as well.

While SOFAs mapping function maps the changes of the volumetric mesh onto the surface mesh, it doesn't alter the structure of the mesh itself, meaning that it doesn't add or remove vertices from the mesh. The meshes that are provided by SoRoCAD have a very low resolution, since all actuators are basically a series of cuboids and each cuboid only requires eight vertices. Now, when we have an actuator whose outer shape is defined by eight vertices, there is only so much change that can be mapped onto it. Combined with this, the CSG functions can sometimes cause artifacts in the mesh topology right around the areas where two cuboids merged. This can lead to some areas of the mesh having a large resolution and other areas having barely any vertices. To really visualize the changes for the user, a more uniform, higher resolution mesh is required from SoRoCAD. The shape of the actuator should still stay

the same, but there should be more vertices in the surface mesh, that SOFA can map to.

This can be achieved by a process called mesh subdivision. There are again several tools and frameworks that can do this. One of them is CGAL, which, as described earlier is mostly a C++ framework. There are also some Python libraries that can provide mesh subdivision, like PyMesh or trimesh. However since the algorithm is not that complicated and the results we wanted were not that complex, we decided that it would be simpler to just implement our own mesh subdivision function, rather than to integrate a library.

The idea of subdivision is simple, but first we have to make sure that our mesh consists only of triangles. During implementation we realized that the SCG functions can sometimes result in some four-sided polygons appearing in the mesh. Splitting a four sided polygon into two triangles seems like a simple tasks, but can be tricky when the polygon is concave. Thankfully Euclid, the CSG library we used, already provides a triangulate function for polygons, that we can use.

Now that we know that our mesh consists of only triangles, we can iterate over the mesh. For each triangle in the mesh, we take the coordinates of all three vertices and use them to create the edges between them. We cut each edge in half and put a new vertex in the middle point between each pair of vertices. So for each triangle that we iterate over, we create three new vertices. These three new vertices together with the original three are connected with each other in a way that creates four triangles out of a single triangle. So each run of this function quadruples our triangle count and doubles our vertices count, while keeping the shape of the mesh exactly as it was. Running this function two to three times on the input mesh gives us more than enough resolution to see the deformations that the simulation applies to the volumetric mesh.

As mentioned earlier the output from SOFA was originally just a still image that depicted the deformed actuator after 100 time steps of simulation. Later on we found that SOFA

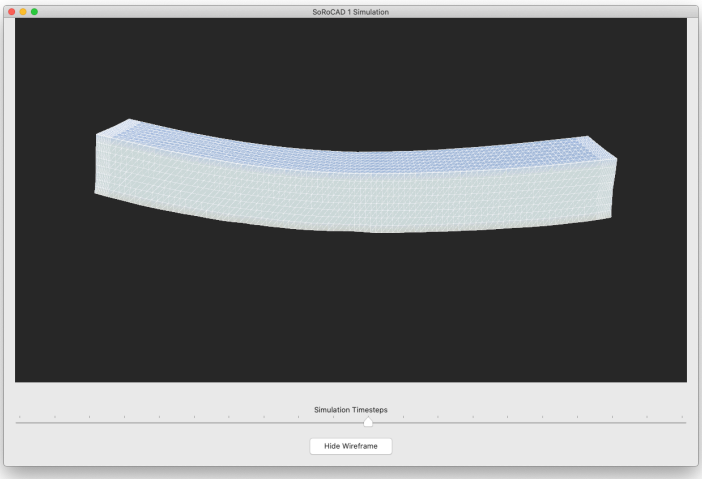

Figure 3.11: SoRoCAD Simulation Window

was capable of exporting the deformed mesh itself and that it could do so at every individual time step. This means that with the current 20 time steps per simulation run, the output consists of 20 different meshes, where each one is given a number and exported to a file. SoRoCAD gives each simulation run an ID and opens the corresponding 20 files to load deformed the meshes back into the tool, where they are displayed with the help a time scale slider, as can be seen in Figure 3.11.

Chapter 4

Evaluation

Similarly to the pre-study we conducted, we wanted to evaluate the redesigned user interface and workflow as well as the finished simulation feature with a qualitative user study. In the pre-study participiants were asked questions about the tool after a short five minutes session where they could familiarize with the tool, but they were never given an explicit task to complete by using it.

In the final study, we decided to give them a series of tasks which are designed to encourage the use of several features of the tool, including the templates and the simulation feature. We wanted to see whether participants can find and understand these features, whether they try to use them and whether they would continue to use them throughout the study.

4.1 Apparatus

Due to the ongoing pandemic, we decided not to conduct the study in person to ensure the participants' safety. For this reason we started looking for remote alternatives, but due to implementation problems as well as a lack of Mac computers among the participants, simply distributing the tool to the participants directly was not an option. We ex-

plored other options and ultimately decided on a remote desktop solution, where participants are given control of the interviewer's computer where they can use the tool. We originally planned on using TeamViewer for this task, however we discovered Zoom can provide this functionality as well and so we ultimately decided to use Zoom for the video call as well as for the remote desktop control.

The whole study was recorded using Zoom as well. All audio was recorded as well as the content of the screen that was provided to the participant. Most studies took around 60-75 minutes with the longest one taking around 90 minutes. Participants were also provided with a questionnaire to collect demographic data. Apart from age and profession, participants were also asked about their experience levels with Soft Robotics, MacOS and CAD tools. If participants had any experience with CAD tools, they were also asked which specific tools they had used in the past.

4.2 Participants

We had 10 paricipants in the study, 6 of which were male and 4 were female, with an average age of 25(SD: 2.21). Participants had a variety of professions, though most of them were students. Only one participant reported middling experience with MacOS, while most of the participants reported either little or very little experience with the operating system. Two participants reported high or very high experience with CAD tools, one participant reported middling experience, but the rest of the participants reported little or very little experience with CAD tools. Every participant across the board reported little or very little experience with Soft Robotics.

Participants also each provided a list of CAD tools that they had used before, if any. Four participants named no tools, while most others named one tool each. The two participants with the highest amount of experience with CAD tools also provided the broadest variety of tools that they had used before. The most commonly named tools were Fusion360, FreeCAD and Blender.

4.3 Study Procedure

Participants were given a questionnaire as well as a consent form to fill out and return before the study. Also before the study they were given a short introduction to Soft Robotics. This introduction included an explanation by the interviewer, who would contrast soft robotics with regular robotics. After the explanation, users were given some application examples, including an explanation of a soft robotic glove as well as a video of a soft robotic gripper. To finish the introduction users were shown a video of a bending actuator, that was designed with the tool and fabricated by the interviewer, to emphasize how an actuator made with the tool might look when fabricated.

They were also given a very short introduction to MacOS. This introduction only consisted of telling participants where they could find an application's toolbar on the top of the screen. To finish off the introduction they were told about the camera controls, which needed explaining because of the remote desktop setup of the study. Usually the camera is mostly controlled by touchpad gestures, but since most users did not have a touchpad and had Windows computers anyway, they had to press the Alt key to move the camera around which was neither intended nor intuitive.

Following the introduction and before participants saw or interacted with the tool, participants were asked about what they would want from a tool that would support them in designing a soft robotic actuator. Participants were told not to focus on implementation details or feasibility but to describe anything they would wish for.

Now participants were presented with the first of four tasks. The first task asked the participants to use the tool to design an actuator with desired measurements and a desired behaviour and to validate that the actuator actually exhibits that behaviour. The first actuator participants were asked to design had to be 10cm long, 2cm wide and 2cm high. When inflated the actuator should simply enlarge in all directions. After this they were asked to design

a second actuator with the same measurements but a different behaviour. When inflated this actuator should bend upward. After each completed subtask, participants were provided with a new instance of the tool, resetting anything they had done with the tool. This task was designed in such a way that the use of templates would trivialize the design process. The enlargement template results in the first actuator, while using the bending template results in the second. Successfully designing and validating the behaviour of each of these two actuators concluded the first task of the study.

The second task that participants were presented with, was similar to the first. Participants were again asked to design an actuator and to validate its behaviour. In this task though, the measurements were irrelevant and only the behaviour of the actuator was of interest. The behaviour of the actuator was split into two halfs. The right half of the actuator was to bend upwards, while the left half of the actuator was to not bend at all. The left half could inflate or do nothing, as long as it doesn't bend. This task was designed in a way that made the use of templates still valuable but the task required more than simply applying the template from the participant. Using the first or second template could give them a starting point, as the resulting actuator would fulfill one of the two requirements and participants would only need to adjust the other half of the actuator. Again, successfully designing and validating the behaviour of the actuator concluded the second task.

The third task was comprised of two questions that would test their comprehension of soft robotic design parameters and how they affect the behaviour of an actuator. This task was done without the use of the tool, as we only wanted to see whether they successfully got an understanding of design parameters like the wall thicknesses by using the tool in the first two tasks. The first question was how they would go about designing an actuator with a specific desired behaviour. The left half of the actuator was to bend downwards and the right half of the actuator was to bend upwards, effectively creating an S-shape. This is similar to the second actuator of the first task, but participants need to understand how to create a bending motion that goes

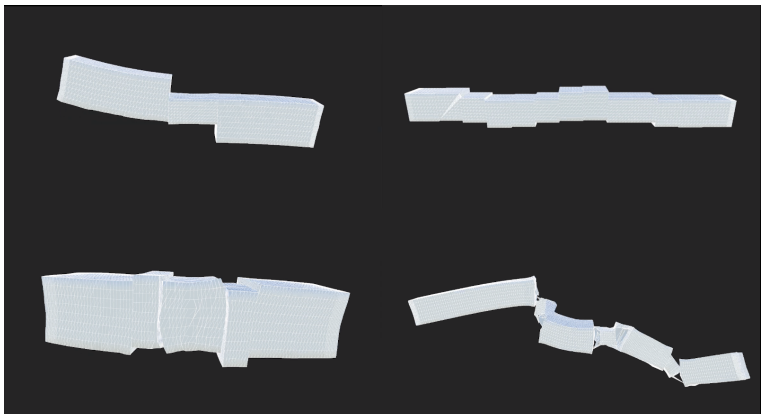

Figure 4.1: Simulations of different actuators that participants produced in the 15 minute creative session

in a different direction than the bending motion provided by the template. The second questions was similar to the first, but with a different desired behaviour. The left half of the actuator should enlarge in all directions, while the right half should bend upwards. Both of these questions were intended to check the participants understanding of how different wall thicknesses influence the behaviour of an actuator. Correctly answering both of these questions would conclude the third task.

The fourth task was designed to let participants explore the tool and to get creative. Participants were given 15 minutes to play around the tool and were given free reign, although they were encouraged to work on an actuator that they find exciting or interesting. This task was designed to let participants test the limits of the system by freely working with the tool. This was also where we wanted to observe whether participants took the knowledge they had gained in the previous task to apply it to their actuator. Figure 4.1 shows a collection of different actuators that were produced in this part of the study by participants.

The final part of the study involved a short interview of the participant. There were some questions that were asked of every participants but the interviewer also asked about any individual issues that were noticed by the interviewer in the specific study. Participants were asked about whether

they had any issues when working with the tool, or anything that did not seem intuitive to them. Based on their answers the interviewer would then investigate further on topics. One question that every participant was asked was about how they managed to gather the necessary knowledge about design parameters like wall thicknesses to design the actuators needed for the different tasks. After the short interview session, the study was finished.

4.4 Results

The resulting video recordings were transcribed and the transcriptions were then coded to easier evaluate them. During the course of the study, several interesting results were found that span multiple categories. Like we mentioned earlier, we looked at the different ways that participants gathered the necessary knowledge about design parameters to solve the tasks. This topic comprises the first category of findings. We also had some findings about several different UI elements or workflow steps that were designed to help the user generate that knowledge, like the help window, the template selection or the GIFs that accompanied the design process. Each of these parts of the UI has its own category of findings as well. We also looked at the viewport interactions that we implemented and evaluated whether they were helpful to users. The final category comprises general user interaction issues, like problems with the camera, the sliders or missing features that participants expected to be there.

The first and most interesting category of findings was all about knowledge gathering. We had asked participants about how they manage to figure out what the different parameters do and found that there were a variety of knowledge sources that they reported. Four out of ten participants reported that they had gained knowledge from using the simulation feature. ”I had it completely backwards. I realized that because I used the simulation and was able to see it”(P2). This kind of experimental behaviour, where participants took a theory from somewhere else and used the simulation to confirm or disprove their theory, was ob-

served in other participants as well. "The right side is bending in exactly the wrong direction, that is interesting. That's the exact opposite of what I would have expected"(P6). Participants were typically observed using the simulation in such a way, when they had started with incorrect assumptions about design parameters. Participants who had extracted a right assumption from other sources, typically used the simulation as part of the task and didn't take much more knowledge from it other than the validation of their assumptions.

Other participants answered that they had gained knowledge from using the templates. Two out of ten participants reported such a finding. "If I look at this, then I see, okay, if there is more empty space up here and less empty space down here, then it presses upwards."(P3). This kind of knowledge gained by observation was probably aided by the structure of the study. The first task asked users to first create an enlarging actuator, followed by a bending one. Since most participants had used templates for both of these tasks, some participants checked for the differences in parameters between the two actuators that were generated by the template, in order to better understand which parameters caused the shift in behaviour. "Primarily, because I used the templates at the beginning, and I saw that the wall thicknesses for one of the actuators were lower than for the other actuator. If you see that once, it becomes pretty clear that that influences something"(P2). This avenue of knowledge gathering only worked for people who had actually used the templates, which was not everyone, and it also did not work for everyone who had used the templates. There were participants who had used templates to complete task 1 and still struggled with task 2 where templates were no longer as helpful as they were in task 1.

Other participants took their knowledge about design parameters from sources completely outside of the tool. Two participants reported that they had formed their first theories through experience with topics that had comparable aspects to soft robotics or simply through deduction from information that they received earlier. "Since I got an explanation of the basic functioning, so we have Air Channels and we pump air into them. You can figure out how it

would behave with thinner walls but the same pressure. So it was mostly deduction."(P1). The other participant took his first theory from his chemistry classes, that he once took in school. "We had Bi-metals back in chemistry class and they behave somewhat similar. You have a metal that bends easily on one side and a metal that doesn't bend easily on the other. If you heat them up, they expand and it bends in a direction. [...] That's why it seemed a little bit familiar to me."(P3)

Another UI element that was intended to convey knowledge to the participants was the help window. Seven out of ten participants read the tips that were displayed in the help window, while the other three did not read them. Those three participants usually clicked the help window away almost immediately. Of the seven that read the tips, not all of them read every tip. Some of them only read the first tip or the first two tips. Interestingly not every tip reader was able to utilize the concepts that the tips were supposed to teach them. Some struggled for a while in tasks 1 or 2, even though they took a long time, carefully reading every tip. Interestingly those participants made remarks that suggest that they had completely ignored the tips or didn't read them at all, even though they clearly read them at the beginning. "Yeah, [I only did that], because you get the explanation from this tutorial, which I completely ignored."

The last helpful UI element was the GIFs that played whenever participants changed a wall thickness value. For five out of ten participants situations were observed where the GIFs were considered to be helpful. However this was mostly in situations where the participant was already on the right track and the GIF only reinforced them in their theory. "Ah, I just saw that when I clicked on this, it already showed me how the actuator can potentially behave, in the bottom right. That was already bending the way that I envisioned it."(P5). One participant seemed to think that the GIF was completely determined by the template that they had selected earlier. Two participants did not realize that the GIF had changed throughout their session, which also lead them to be unsure about its purpose. "[...]And it's also not a different GIF based on the slider that I use, so I'm

not sure what that's supposed to do, except that I'm scaling something, which might lead to different behaviour?"(P2).

Interestingly, all of the ten participants answered the comprehension questions in task 3 correctly. So through a mix of all of these different UI elements and workflow steps, all participants found some way to gather the necessary expertise about design parameters. Additionally almost all of the participants used the expertise they gained in the first two tasks, to design their actuators in the final task.

When it comes to viewport interactions, five out of ten participants found that they could click on one of the buttons that enabled a viewport interaction. But no participant used all three of the available interactions. There were even some participants that only used a part of a single interaction as an unintended feature. "I'll have to take a look. Can I display the axes with this button up here? Yes perfect!"(P5). The scaling interaction that this participant clicked on, displays handles along each axis through the object, that can be pulled on in order to scale the object. In this case they were only looking for a way to visualize which axis was where in the viewport. They used the axis handle for this purpose and never discovered that they could pull on them to scale the object. There were also participants who found the viewport interactions, tried them out for a little bit and then turned back to other interaction modes, like the text fields.

There also were more general interaction problems. This category is for any issue or missing feature that came up, that didn't fit into the other categories, so the content of this category is quite broad. One issue that came up a lot was the lack of physical units on many sliders and text fields. Four out of ten participants reported something like this. Usually this was followed by assumed units that the participants thought would fit. This issue mostly came up, when selecting a template. After a template button is clicked, the tool asks the user for dimensions. These are given in cm, even though the UI does not specifically says so. "So, the question is, what unit are those? I think it's probably centimeters but it could also be millimeters."(P1). Another part of the UI where this came up was the simulation, where

users could step through the different time steps of the simulation with a slider. ”Are these steps in seconds?”(P10), ”This just says ’Simulation Timesteps’. What are those exactly? It would be good if that was displayed, because right now it doesn’t mean anything to me.”(P2)

Another big issue was a quality-of-life feature that participants expected to see. Five out of ten participants voiced the wish to select and manipulate multiple blocks at the same time. ”That’s my big take-away. I want to be able to edit multiple blocks at the same time.”(P1). This concern was usually voiced whenever participants knew how they wanted the actuator to look, but they had to manually change 8-10 blocks to achieve it. This is usually where they looked for a faster solution, that was not there. Another related feature that was requested by participants, was a copy-and-paste function. This is basically the same idea just implemented in a different way. Instead of trying to select and manipulate multiple blocks at the same time and giving them the same parameters, participants tried to select one block, that they had already finished and tried to copy over its parameters to other blocks.

Due to a bug in template selection, that was fixed early on, some participants entered task 1 with a generated actuator that had the wrong dimensions. This is where another interesting finding was discovered. Participants realized that they had to change the size of their actuator and they were looking for a global size parameter. In the top right of the screen there were some labels that showed the current size of the actuator in width, height and length. Two participants tried to click these labels to see if they could simply re-enter their desired dimensions there, like they had done earlier in the template selection.

An interesting UI-specific finding was that most participants did not use the sliders, but instead used either viewport interaction or, mostly, the text fields next to the sliders. When asked about why they preferred the use of the text fields over the sliders, some participants answered, that they did not like how imprecise the sliders were. ”I think any sliders that don’t snap to values are pretty unintuitive.”(P2). The same participant enjoyed the use of

the viewport controls for scaling, which is interesting because the viewport controls only change the wall thicknesses in increments of 0.1, which is exactly the ”snapping”, they describe in the quote. Similarly participants voiced concerns that the value of the slider was too specific and that there were too many floating point digits. Even though the sliders try to snap to values in increments of 0.1 as well, users can set values between those snap points and putting the slider into the right position for it to snap can require some precision. One participant reported that they did not find the arrangement of sliders very intuitive. They were ordered by axes, so ”top” and ”bottom” were the first two sliders followed by ”front” and ”back”. The participant said that they expected the ”top” slider to be the first one and the ”bottom” slider to be last, to match the arrangement of actuator walls in the viewport. This was an interesting finding, as multiple participants experienced misclicks, where they wanted to adjust the bottom wall thickness, but accidentally clicked on the ”back” slider.

The camera was another point where many participants voiced concerns. Since the 3D space was completely empty apart from the actuator, participants had some trouble trying to orient themselves properly. This happened especially when they tried to select a block and accidentally rotated the camera. ”It’s a little hard to guess which side is up, down, right or left, especially when you’ve rotated the camera”(P3). One participant accidentally rotated the camera early on and looked at the actuator upside down. This was only noticed when they tried to change the top wall thickness and the lower wall in the viewport changed its size instead. Instead of doing the arduous task of realigning the camera, they instead kept it that way for the remaining task because top and bottom was essentially arbitrary anyway. Three participants were looking for some way to see the different axes, to more easily see how the actuator was currently oriented, ”If the axes were displayed, then one could identify the different directions that way.”(P4) and one participant said that he had expected an element in the viewport that helps with orienting yourself. ”[...]usually, in CAD tools, you have some kind of gizmo in a corner that turns with the camera.”(P2). Other participants wished for a camera reset button to get back to the default orienta-

tion. ”Can you reset the perspective? Because that would be very helpful to go back.”(P7).

Similar to this, two participants had issues identifying the different walls by their given names. Especially when it came to the front and back walls, which are perpendicular to the Z-Axis. The names ”front” and ”back” to some were very similar to the ”left” and ”right” walls perpendicular to the X-Axis and the naming seemed arbitrary. ”I thought ’Front’ was this side right here. *points to the left wall*”(P3). To remedy issues like this, two participants voiced the wish for a displayed grid in the viewport. ”An orientation in general would be good, with axes or a coordinate system, so you can better orientate yourself.”(P10).

Another thing we observed was that some participants focused on the air channels instead of the wall thicknesses as their parameters of choice. There were participants, especially among those that did not use the templates right away, that started out working on the air channels exclusively. This was usually before they even realized that there were multiple tabs. Another group of participants focused on air channels in the exploration of task 4. This group had used the templates for the earlier tasks, which immediately puts the user into the ”Silicone Exterior” tab, skipping the air channel design, so in their exploration of the tool they found the first tab and started experimenting with it and trying out how the air channel design influences behaviour. Usually this resulted in very complex air channel designs with wildly rotated segments that were often disconnected from one another. This is also the point where some participants started thinking about the air supply and how the air is distributed inside of the actuator, although some participants even considered this in the earlier tasks as well.

4.5 Discussion

The main finding of the study is that all of the participants were able to gain some expertise in designing soft robots with the help of the tool. That mainly means that they learned how the different wall thicknesses effect the bend-

ing behaviour of a soft robotic actuator. However it is not easy to determine whether that was only because of the design and the workflow of the SoRoCAD tool, but maybe also because of the structure of the study. All of the helpful features that we implemented in the tool appeared to help at least in some degree, although some were clearly more helpful than others. Interestingly all of those features seemed to help participants in different ways and some participants relied heavily on the help from one feature while completely ignoring another feature.

We think the most helpful feature was probably the simulation. Everybody used the simulation and there were no large issues with the use of the feature, apart from specific details like the missing units on the time step slider for example. It appeared that the simulation was mainly used not in an exploratory way but more in an experimental way. Instead of creating an actuator and completely relying on the simulation to show them the behaviour of the actuator, participants usually formed a theory on how it would behave at design-time and used the simulation to confirm or refute their theory. Often times in these situations the simulation was more helpful in refuting an idea than in reinforcing one. Participants were never explicitly instructed to run a simulation, but simply to validate the movement of their actuator. Every participant found the simulation feature to fulfil that task quickly as soon as they found the second tab, so it seems that the feature is easy to discover as well.

The templates seemed helpful to most participants as well, however they definitely had more discoverability issues. There were participants that completely disregarded the templates for the entirety of the study, while others had misunderstandings about their function. One user had stated that they thought the templates were not buttons, but just a kind of preview, a list of things that they could build with the SoRoCAD tool. This is probably indicative of an issue with the buttons and their presentation, that doesn't make them look "clickable" enough. Another participant had envisioned the first window of the tool to function as a sort of toolbar, from where they could drag and drop the templates into the 3D viewport. After click-

ing on the template and being presented with the popup window where users can input their desired dimensions, they thought that they might have set individual blocks to follow that template and those dimensions that they had entered into the tool. This shows that there were not only discoverability issues but also some issues in understand what the feature even does. This could be made clearer to the user, by something as simple as a label above the buttons, or clearer language on the buttons. Overall the feature was helpful because it provided participants with concrete examples of how an actuator might look, what it might do and how those two things are linked. However this helpfulness is only achieved when people use the feature, which is not a given, and it is most helpful if they use it multiple times with different templates so they can see the difference between actuators and how those difference effect behaviour.

The help window with the three tips seemed helpful as well, although again, that helpfulness did not extend to every participant. The tips serves as a front loaded tutorial that just gives users information in form of text and an animated GIF. This is pretty crude in terms of UI design but it seemed to be somewhat effective in conveying information to users. However it was still evident that some users read the tips but don't retain the information very well. This might be precisely because it is front loaded. Users see the tips before they even had a chance to see the tool or familiarise themselves with the parameters that they can manipulate. It might be a good idea to give more targeted tips later in the workflow instead and to display them in a less disruptive way. The tips could even be integrated into different UI elements similar to how the GIFs were implemented. One could display more granular tips whenever the user interacts with a slider for example. One participant had the idea to make the tips dependent on the selected template, to give deeper insight to the user about their specific use-case and that might be another interesting avenue to explore when improving the User Experience.

The GIFs seemed to be helpful in very limited cases, though that might be very dependent on the current implementation. Similar to how the simulation was mostly useful for

refuting wrong assumptions, the GIFs appeared to mostly affirm to participants that they were on the right track. This might be a useful effect, even though the GIFs never directly taught a participant anything like the other features did. One participant had the idea to put an image of the used slider into the GIF to ensure that the connection is easier to make. As stated in an earlier chapter, the ideal solution would be a live simulation instead of premade GIFs, but in the absence of such a solution it might be possible to make the GIFs themselves more specific or to make it clearer what they represent. The current implementation checks each half of the actuator and displays the corresponding half of a GIF, decided by the behaviours of the majority of the blocks. This makes it somewhat accurate, but it also lead to situations where participants had to change many blocks to see any change in the GIF, if they even saw any change at all.

The workflow itself appeared to work fine, although there were some cases where participants tried to influence wall thicknesses in the mold tab. This might be caused by the sliders in the mold tab that change how thick the walls of the mold are, which is purely a fabrication parameter for 3D printing. It might be a good idea to rename that slider or to remove the functionality entirely.

Viewport interaction was sometimes used, but not to the extent that we anticipated, although most people that used some viewport interaction seemed to prefer it to sliders or text fields, especially when it came to manipulating many blocks in a row in the same manner. This issue with discoverability probably came to be due to misleading icons and buttons that had an unfortunate color scheme, that lead users to believe that the button was not clickable. Obviously those are areas where things can be changed, but it might even be interesting to have some viewport interactions enabled by default and allowing users to opt out of them, instead of opting in.

Chapter 5

Summary and future work

5.1 Summary and contributions

In this paper, we presented SoRoCAD, a specialized CAD tool for designing, simulating and fabricating soft robots. We took a look at the the previous state of the tool and conducted a pre-study to gather feedback and design ideas to improve the user interface of the tool. We also implemented a simulation feature, by integrating SOFA into the tool, a general physics simulation framework with a specialized plugin for soft robotics.

Based on the feedback that we gathered from the pre-study, we proposed and fully implemented a redesign of the tool's workflow and user interface. The scope of the tool was reduced and it was geared towards users that have little experience with soft robotics. In order to support this group of users we added several features to assist users in gathering information on how to design soft pneumatic actuators and how the different design parameters affect the behaviour of an actuator.

The first and biggest one of those features was the simulation, allowing users to see the behaviour of their design in

an animated way. Implementing this feature required both a working connection with the SOFA tool as well as a functioning user interface to display the results. We also implemented other features to assist users in their exploratory workflows, such as a help window that displays helpful tips, a template system that gives users a start-off point, as well as little GIFs that display whenever the user changes parameters, and gives them a broad overview of how the parameter changes might affect the outcome. We also decided that any user interaction that the tool offers through sliders should be mirrored by direct interactions through the viewport and while we did not manage to mirror every single slider interaction, we did manage to implement quite a few viewport interaction techniques.

To evaluate our new design and the new simulation features, we decided to conduct another qualitative study. We wanted to answer two questions in particular:

1. Can novice users design functional soft robotic actuators using SoRoCAD?

2. To which capacity are the different parts of the tool able to assist users?

In order to answer these research questions, we gave participants tasks and goals that they had to fulfill with the help of the SoRoCAD tool. Afterwards they were asked questions about their work with the tool and what they thought about it. All participants were able to complete the tasks, despite only being given a very minimal introduction to the domain, and having no prior knowledge of soft robotics. We found that the assistive features described earlier helped participants in different ways, some being more effective in conveying knowledge to participants than others. We also found that the viewport interactions were hard to find and as a consequence rarely used. The simulation feature however was found and used by every participant without instruction by the interviewer.

Overall, the tool seemed to be successful in giving novice users an introduction to the world of soft robotics and giving them an understanding of the design parameters in-

volved. There are still several areas in which the tool can be improved that we will describe in the next chapter, but the results of the study seemed to show an improvement when compared to the results of the pre-study, which tried to evaluate the earlier design of the user interface and the workflow.

5.2 Future work

Despite the good results of the study, we identified several issues with different user interface elements that were either not found by users, or confused users in their current implementation. However one of the biggest problems is a technical problem with the SOFA integration, that causes deployment issues. Currently SOFA and SoRoCAD need to be manually compiled on the user's machine because we were not able to compile SOFA in a way that removed all non-relative file paths within the tool. There are pre-compiled versions of SOFA available and even ones that include the Soft Robotics plugin, however these pre-compiled versions do not include the CGAL plugin that we need for conversion from surface meshes to volumetric meshes. This leads to a situation where we can deploy SoRoCAD to other macOS machines, but the tool crashes as soon as it tries to run a simulation. If this problem was solved then deployment to other macOS machines would be easier.

The next issue is with templates, and while they were used by the majority of participants, there was still some confusion about what they are for some. Some participants completely ignored the templates, while others had misunderstandings about their purpose. However, most of these problems can be solved by altering the language on the buttons or by slightly redesigning the template selection window to make the buttons more obviously clickable.

Another discoverability problem was found with the viewport interactions. Those who found the interactions, often preferred them over using the sliders, but finding them was not as simple as we had hoped. Part of the problem was due to an unfortunate selection of icons, that did not con-

vey the right signal to users. Especially in the second tab, users thought that the button would only display the different axes to them. More meaningful icons would obviously solve this problem, but it might also be interesting to look into enabling the viewport interaction by default. Letting users opt-out of using them, instead of needing to find them first, might help discoverability.

The GIFs were somewhat helpful but only in a very limited way. They helped to reinforce users when they had already made the right decisions. There were a few instances in the study where this sort of situation happened, but most users were either confused by the GIFs or outright ignored them. It might be useful to bind them closer to the actions that caused them to pop up, i.e. putting them closer to the individual slider that the user just used. This could help users understand that the GIFs are there to give them feedback on their actions.

The final area of improvement is the help window that displays useful tips to the user. While most users read the tips, only a fraction of them were actually able to transfer the knowledge conveyed through the tips to their tasks. The window always displays the same tips, independent of what the users selected at the start. It might be interesting to display more context dependent tips. For example, if the user selected the bending template, you can give them more tips on how bending works, how to increase the force of the bend and how to influence the direction in which the actuator bends. Users might be more open to help and more capable of absorbing the information if it helps them at their current tasks.

In the future the SoRoCAD tool can be expanded in a number of different ways. For one, the complexity of producable actuators can be increased, by offering more air channel shapes, new exterior shapes or by allowing users to stack segments in multiple dimensions. Another possible expansion would be allowing multiple actuators in a single scene. This is something that would allow users to design and simulate a complete gripper system for example and it also creates the opportunity for more complex soft robots without making individual actuators more complex.

Appendix A

Study Materials

Informed Consent Form

Evaluating a tool for designing and simulating Soft Pneumatic Actuators

PRINCIPAL INVESTIGATOR Jakob Strüver
Master Student
RWTH Aachen University
Email: jakob.struever@rwth-aachen.de

Purpose of the study: The goal of this study is to evaluate the current state of a tool for designing and simulating Soft Pneumatic Actuators. Therefore, participants will be asked to use the tool and answer interview questions about their experience while doing so.

Procedure: Before the study, the participants are asked to fill out a questionnaire with some information about themselves. During the study, participants will be given multiple tasks that they will have to solve using the SoRoCAD tool. During the study, audio as well as the screen will be recorded. This study will take about 45 minutes to complete.

Risks/Discomfort: You may become fatigued during the course of your participation in the study. You will be given several opportunities to rest, and additional breaks are also possible. There are no other risks associated with participation in the study. Should completion of either the task or the questionnaire become distressing to you, it will be terminated immediately.

Benefits: The results of this study will help to identify issues with the current design of the tool. It will also gather information about which features are expected by potential users.

Alternatives to Participation: Participation in this study is voluntary. You are free to withdraw or discontinue the participation.

Cost and Compensation: Participation in this study will involve no cost to you.

Confidentiality: All information collected during the study period will be kept strictly confidential. You will be identified through identification numbers. No publications or reports from this project will include identifying information on any participant. If you agree to join this study, please sign your name below.

_____ I have read and understood the information on this form.
_____ I have had the information on this form explained to me.

Participant's Name	Participant's Signature	Date
	Principal Investigator	Date

If you have any questions regarding this study, please contact Jakob Strüver at email: jakob.struever@rwth-aachen.de

Figure A.1: Consent Form for the final study

Questionnaire

How old are you? _____

What is your profession? _____

How much experience do you have with the macOS operating system?

Very little A lot

☐ ☐ ☐ ☐ ☐

How much experience do you have with CAD (Computer-Aided-Design)?

Very little A lot

☐ ☐ ☐ ☐ ☐

If you have experience with CAD-tools, which tools have you used before?

_____

How much experience do you have with Soft Robotics?

Very little A lot

☐ ☐ ☐ ☐ ☐

Figure A.2: Questionnaire for the final study

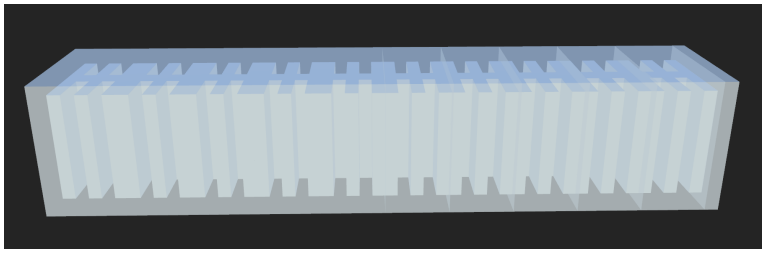

Figure A.3: First image shown during the first build task of the study

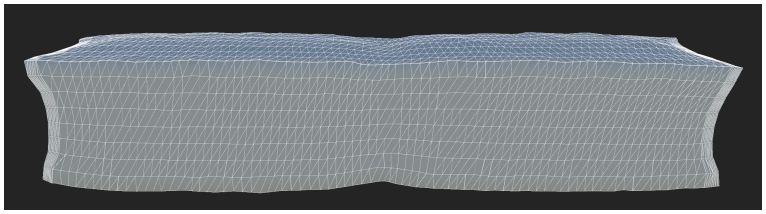

Figure A.4: Second image shown during the first build task of the study

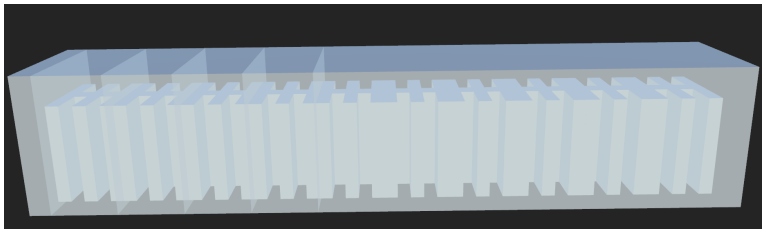

Figure A.5: First image shown during the second build task of the study

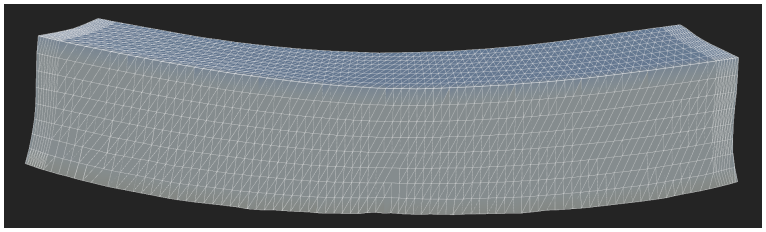

Figure A.6: Second image shown during the second build task of the study

- Did you encounter any challenges when working with the tool? If so, what kind of challenges?

- For each challenge or problem encountered:
  - Which one was the worst?
  - Why did these problems occur (from your perspective)?
  - What can help you for each of the problems (if you have an idea at all)?
  - Name example to solve your problems.

- You can wish for everything: Please describe your ideal environment to fabricate a soft robot

Figure A.7: Follow-up questions that were asked after the study

Bibliography

[1] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen, and G. M. Whitesides, "Soft robotics for chemists," *Angewandte Chemie*, vol. 123, no. 8, pp. 1930–1935, 2011.

[2] P. Polygerinos, Z. Wang, K. C. Galloway, R. J. Wood, and C. J. Walsh, "Soft robotic glove for combined assistance and at-home rehabilitation," *Robotics and Autonomous Systems*, vol. 73, pp. 135–143, 2015.

[3] S. Karmakar and A. Sarkar, "Design and implementation of bio-inspired soft robotic grippers," in *Proceedings of the Advances in Robotics 2019*, pp. 1–6, 2019.

[4] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Applied bionics and biomechanics*, vol. 5, no. 3, pp. 99–117, 2008.

[5] S. Kim, C. Laschi, and B. Trimmer, "Soft robotics: a bioinspired evolution in robotics," *Trends in biotechnology*, vol. 31, no. 5, pp. 287–294, 2013.

[6] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides, "Multigait soft robot," *Proceedings of the national academy of sciences*, vol. 108, no. 51, pp. 20400–20403, 2011.

[7] M. T. Tolley, R. F. Shepherd, B. Mosadegh, K. C. Galloway, M. Wehner, M. Karpelson, R. J. Wood, and G. M. Whitesides, "A resilient, untethered soft robot," *Soft robotics*, vol. 1, no. 3, pp. 213–223, 2014.

[8] A. D. Marchese, C. D. Onal, and D. Rus, "Autonomous soft robotic fish capable of escape maneuvers using

fluidic elastomer actuators," *Soft Robotics*, vol. 1, no. 1, pp. 75–87, 2014.

[9] B. Zhang, Y. Fan, P. Yang, T. Cao, and H. Liao, "Worm-like soft robot for complicated tubular environments," *Soft robotics*, vol. 6, no. 3, pp. 399–413, 2019.

[10] M. Cianchetti, M. Calisti, L. Margheri, M. Kuba, and C. Laschi, "Bioinspired locomotion and grasping in water: the soft eight-arm octopus robot," *Bioinspiration & biomimetics*, vol. 10, no. 3, p. 035003, 2015.

[11] H.-T. Lin, G. G. Leisk, and B. Trimmer, "Goqbot: a caterpillar-inspired soft-bodied rolling robot," *Bioinspiration & biomimetics*, vol. 6, no. 2, p. 026007, 2011.

[12] C. D. Onal and D. Rus, "Autonomous undulatory serpentine locomotion utilizing body dynamics of a fluidic soft robot," *Bioinspiration & biomimetics*, vol. 8, no. 2, p. 026003, 2013.

[13] B. Mosadegh, P. Polygerinos, C. Keplinger, S. Wennstedt, R. F. Shepherd, U. Gupta, J. Shim, K. Bertoldi, C. J. Walsh, and G. M. Whitesides, "Pneumatic networks for soft robotics that actuate rapidly," *Advanced functional materials*, vol. 24, no. 15, pp. 2163–2170, 2014.

[14] J. Zhang, A. Jackson, N. Mentzer, and R. Kramer, "A modular, reconfigurable mold for a soft robotic gripper design activity," *Frontiers in Robotics and AI*, vol. 4, p. 46, 2017.

[15] Y. Sun, Y. S. Song, and J. Paik, "Characterization of silicone rubber based soft pneumatic actuators," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4446–4453, Ieee, 2013.

[16] B. Finio, R. Shepherd, and H. Lipson, "Air-powered soft robots for k-12 classrooms," in *2013 IEEE Integrated STEM Education Conference (ISEC)*, pp. 1–6, IEEE, 2013.

[17] H. Moradi and C. Torres, "Siloseam: A morphogenetic workflow for the design and fabrication of inflatable silicone bladders," in *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, pp. 1995–2006, 2020.

[18] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[19] J. Hiller and H. Lipson, "Dynamic simulation of soft multimaterial 3d-printed objects," *Soft robotics*, vol. 1, no. 1, pp. 88–101, 2014.

[20] S. Consortium, "Sofa - simulation open framework architecture." `https://www.sofa-framework.org`.

[21] I. L. D. team, "Soft robotics plugin for sofa." `https://project.inria.fr/softrobot/`.

Index

Typeset October 17, 2020