Toward Shape Optimization of Soft Robots

Thomas Morzadec, Damien Marchal, Christian Duriez

Toward Shape Optimization of Soft Robots

Thomas Morzadec[1], Damien Marchal[2], Christian Duriez[1]

***Abstract*— In this paper we present our work on shape optimization for soft robotics where the shape is optimized for a given soft robot usage. To obtain a parametric optimization with a reduced number of parameters, we rely on an approach where the designer progressively refines the parameter space and the fitness function until a satisfactory design is obtained. In our approach, we automatically generate FEM simulations of the soft robot and its environment to evaluate a fitness function while checking the consistency of the solution. Finally, we have coupled our framework to an evolutionary optimization algorithm, and demonstrated its use for optimizing the design of a deformable leg of a locomotive robot.**

I. Introduction

Designing a soft robot is a challenging task. Compared to rigid robotics a first difficulty is that the kinematics of soft robots is not a fixed part of the design process. A second difficulty with soft robotics is that softness can be achieved with various soft materials like silicone [1], [2] micro-structured materials [3] or specifically designed geometrical arrangement of rigid parts as with tensegrity structures [4] allowing a wide range of variations in design. Combined with the recent developments of $3D$ printing, new possibilities of manufacturing materials appear allowing the fabrication of deformable objects exhibiting complex deformation kinematics by controlling the in-filling [5], [6].

In addition to the material itself, actuation systems are very diverse with approaches including cables [7], pneumatics [8], [9], shape memory alloys [10] or chemical reaction [11]. Any subtle change in the geometry, the choice of material as well as the actuators' or sensors' position may impact the deformation behavior and thus the overall robot's kinematics. In addition, as soft robotics is a recent field of research where researchers are still actively exploring robot designs and their usages but where there is a lack of established know-how to design soft robots. This is why there is dire needs for assistance tools to help designers.

Some tools, like SOFA [12] and its SoftRobots plugins allows modelling, simulation and control of soft robots [13]. The framework contains FEM based simulation of deformable material, deformable inverse kinematics, contact modelling and model order reduction. It has been used to model and control a large variety of soft robots. This framework allows the designers to simulate the physical behavior of the robots before their actual fabrication. But it is far from straightforward for a designer to understand how to adapt the shape of the robot to satisfy some targeted capabilities resulting in a lot of trial-and-error approach testing alternative design.

1 INRIA, 2 CNRS

In the present paper we introduce our work on shape optimization toolkit in SOFA and the results of our experiments done by numerical simulation as well as their comparison to a real physical test bench.

II. Related work

From a mathematical point of view, shape optimization consists in minimizing a function ϕ defined on the space of subsets of a bounded domain of $\mathbb{R}^2$ or $\mathbb{R}^3$ (i.e. the *shapes*) with values in $\mathbb{R}$ (i.e. the *fitness* of the shape). Shape optimization is an active field of research with multiple applications: mechanical design [14], shape modelling, 3D printing [6] or robotics [15], [16]. With the development of 3D printing, shape optimization has been applied to the creation of in-filling structures [5] or surface pattern [6] exhibiting custom mechanical behavior. In the work [17] optimization techniques have been used to design complete mechanisms that can be built. In [18], the authors are providing co-design tools to help designing cable-driven kinematic chains and trees for animatronics.

Methods based on partial differential equations, like level set methods (see for example [19]), are commonly used to move a shape in an externally generated velocity field. These methods can perform shape optimization if the velocity field is obtained, for example, from the *shape deritative* of the fitness function (see [20]). But sometimes, there is no access to derivatives and optimization is based on the evaluation of the fitness function ϕ itself. Other optimization methods, like evolutionary strategies, can be used in such a case. The calculation of derivatives is replaced by the evaluation of not only one but a population of a few shapes [21].

In robotics, evolutionary approaches aims at the emergence of complex mechanical behavior in embodied behavioral agents. The approach is named *Embodied Evolution* [22]. In particular, the performance of the mechanical design of the "embodied agent" (the robot) in interaction with its environment is optimized. It is also referred as *Morphological Evolution*. This approach has been applied to soft robotics in [23], where the shape and the distribution of artificial muscle are optimized to make locomotion emerge. A population of individuals is set and evolves, according to their evaluation thanks to a voxel based mechanical simulation. The shape is encoded with Compositional Pattern Producing Networks (CPPN) and the evaluation is made with the software Voxelyze [24]. In [16], the same authors added interaction between the robot and its environment. But the evaluation of shapes or mechanical objects can also be based on a physical test bench without the need of a model [25]. It has the main advantage to avoid the pitfalls

of the simulations. Since the fabrication of large population of candidate robots is still a time consuming process, authors proposed to automate the fabrication and testing as in [26].

These methods seek the optimization algorithm or the evolutionary process to provide new design ideas. The search space is quite big (in particular when evaluation is made by simulation) and the designer will not impose precise constraints on it. The situation could be the opposite: the designer could have the global idea of the soft robot to be designed. In such a case, the optimization is still useful to adapt the body's shape of the robot to specific constraints or to optimize some behaviors.

[ToEdit: The contribution of this work is to provide one solution] for this situation with an approach in which the designer is able to define the space of shapes and explore it with a reduce number of parameters.

III. Shape optimization

The starting point of our approach is that, given a soft robot design within its operational environment we want to explore alternative designs as well as optimizing some of its properties. We use a black-box optimization technique that does not require the objective function to be continuous or differentiable. In practice, the optimization relies on the evaluation of this fitness function. This evaluation is done in several steps: the shape is generated out of a set of parameters; then we automatically create a mesh, launch a simulation into SOFA and finally make an evaluation based on the simulated physics.

A. Shape Modelling

To model the shapes we use a procedural modelling based on implicit functions similar to ones we can find in [27]. This approach allows to easily generate families of complex shapes from a reduced set of parameters. Reader unfamiliar with modelling with function fields may consult [28]. It contains an overview of the possible shapes defined as field functions, the combining operators to implement *Constructive Solid Geometry* operations and how to implement geometric deformations and transformations. In addition to field functions, it is useful to incorporate shapes defined by other approaches as voxel grids or parametric surfaces & curves. This is possible thank to the computation of an approximation:

- from a vectorized representation (parametric curve or surface, triangle mesh) ones can get a $2D$ and $3D$ grid representation through rasterization.
- from a $2D$ or $3D$ grid one can compute a distance grid with the fast marching algorithm.
- from a $2D$ or $3D$ distance grid as input, an interpolating scheme can be used to get a field function.

To evaluate a shape we use the Finite Element Method to simulate the mechanical behavior of the robots. The FEM implemented in SOFA expects tetrahedral meshes. We use the CGAL library to convert the implicit function describing the shape to tetrahedral mesh, **[ToEdit:** pointing out the sharp edges and corners to CGAL, to be sure that there are preserved].

Once this is done a SOFA scene is made which also contains the modelling of the other soft robot's components like the actuators, the sensors as well as its environment.

B. Optimization

The choice of the fitness function is crucial. Describing the main objective will probably be not too difficult for the designer (for example: optimize energy transmission, optimize stiffness or compliance, reduce weight). But it may appear that optimizing a unique objective leads to undesirable side behaviors or to numerical scenes which are more and more eccentric and less and less reliable. Consequently, we need to define some constraints that penalize these undesirables. The penalization of constraints will be added to the fitness function. The optimization of a fitness function with penalties (**[ToEdit:** that are added up to the fitness function]) is not a multi-level optimization. Multi-level optimization needs specific algorithms. Optimizing a fitness function with penalties works in practice, at least in our example, because we don't try to minimize penalties.

Remark: The reliability of a simulation does not only depend on the shape, but also on some numerical parameters, as the size of the mesh, or the time step value. These parameters cannot be automatically adapted to every case without a significant cost on the performance, as it would require several evaluations of the same individual. These parameters are then fixed. We also have an option to test two or three different mesh resolutions for the same case, to obtain an idea of the influence of these parameters. Moreover, some heuristics can detect when a simulated individual is obviously not reliable (inversion of elements, buckling, undesirable self-collision..). In such cases, we can again use a penalization.

To understand the choice of the optimization algorithm, let's focus now on the properties of the problem. We want to minimize a fitness function that can be *multimodal*, i.e. with many local optimum. It could be ill-conditionned, in particular if parameters are numerous and if penalty constraints are used. Its evaluation can be sometimes very sensitive to the parameters. Finally, the problem is not *separable*, i.e. the n-dimension function cannot be optimized in a sequence of n independent 1-D optimization processes.

For all these reasons, it appears that the evolutionary strategy called CMA-ES (covariance matrix adaptation evolution strategy) is particularly suited for this kind of complex optimization problem.

IV. Evaluation on the design of a soft leg

The legs of the robot are soft legs made of silicone, animated by servo motors. When running, the motor is cycling which results in compressing-releasing the deformable part of the leg. This release of energy is a very nice property because it helps the leg to support the weight of the robot body when the leg is in extension. It allows to have a more constant torque on the motors during the gait. In practice, we

want to design the legs so that they match the capacities of the motors while optimizing the energy accumulation-release property.

Consequently, we have applied our design optimization method to find the shape of the deformable part of the leg. In the following we present the different steps of the method applied on this example. Then, we set up a physical test bench to validate the numerical results.

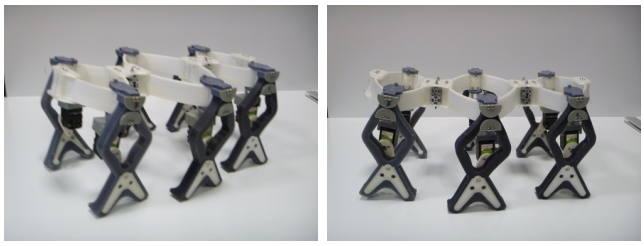

Fig. 1. The Sofia soft robot

A. Leg's procedural modelling

We modelled the Sofia's leg as the extrusion of a $2D$ profile. The extrusion thickness is fixed. This $2D$ profile is the difference of two $2D$ shapes, each defined by splines (one for the external profile of the leg, the other for the inside border of the hole). **[ToEdit:** Each spline is pixelated and then we get the discretized distance function from it thanks to a fast marching algorithm. We thus get two $(2D)$ relative distance functions d_1, d_2. The $2D$-difference is got by the the fuction $f = \max(d_1, -d_2)$. Finally, the extrusion of thickness e is obtained by the function $\max(|z - \frac{e}{2}| - \frac{e}{2}, f)$.**]**

In our case, the motor rotates in both orientations. Consequently, we impose axial symmetry between left and right side (see Figure 2). The splines' control points are procedurally generated allowing to control how many control points to use. **[ToEdit:** In practice, the top and the bottom part of the external spline are frozen, so three pairs of control points are fixed (two for the top, one for the bottom). We tried two configurations: 3 free pairs for the hole and 1 for the external spline (tot. 4), or $5 + 4$. The pairs of control points are uniformly distributed between two extremities. These extremities are frozen for the external spline but free for the hole**]**.

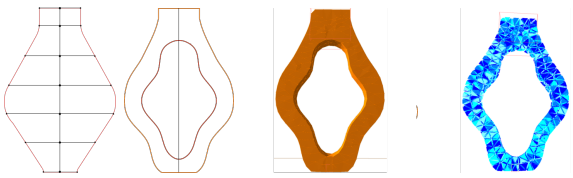

Fig. 2. From left to right. The 2D parametrization. The 2D difference of the two 2D shapes. The resulting extruded shape. The resulting meshed version of the leg.

B. Optimization with design constraints

The optimization is conducted on FEM simulations. We suppose that the shape is attached at its top part. A virtual motor wheel is attached to the bottom of the leg. Note that to perform equivalent tests in simulation and with the experimental set up (see section IV-C), we imposed that the motor has a slow circular movement. In practice, this leads to underestimation of the damping. At each time step, the relative power and torque provided by the virtual motor are evaluated. **[ToEdit:** The simulation provides the forces applied by the virtual wheel to the leg and the velocities of the attachment points: the torque is the sum of the vectorial products between the forces and the vectors between the attachment points and the center of the wheel, and the power is the sum of the scalar products between the forces and the velocities of the attachment points**]**. We record the maximal torque, and we integrate the power, separating the positive from the negative part. The "positive" part is the energy provided by the motor to store the potential energy of deformation inside the leg. The "negative" part is the energy released by the leg.

The objective is: given a torque threshold T_{thr}, we want to get a shape such that the maximal torque provided by the motor does not go beyond the torque threshold and secondly that maximizes the amount of released energy.

1) Fitness function: Let E denotes the amount of released energy and E_0 be its order of magnitude for a reference shape (we chose $E_0 = 0.10J$ because $E = 0.40J$ for the initial shape). Let T_{thr} be the torque that should not be overtaken, and let T_{max} be the maximal torque provided by the virtual motor.

We have a first fitness function $\phi_1 = -\frac{E}{E_0}$ and a second expression with the constraint $\phi_2 = \max(\frac{T_{max} - T_{thr}}{T_{thr}}, 0)$. Minimizing ϕ_2 and ϕ_1 conducts respectively to $T_{max} \leq T_{thr}$ and E maximized. In this formula, $\frac{E}{E_0}$ is the term to optimize, and the second term is a penalty: $\max(\frac{T_{max} - T_{thr}}{T_{thr}}, 0) = 0$ means $T_{max} \leq T_{thr}$. We multiply this term by k, so that the penalty is greater than 1 if $T_{max} > (1.0 + 1.0/k)T_{thr}$. Since vanishing the penalty is much more important than maximizing E/E_0, in practice we use the fitness function:

$$\phi = \phi_1 + (k\phi_2)^2. \tag{1}$$

[ToEdit: Remark: ϕ_1 and ϕ_2 are built to be dimension free and the values of ϕ_1 are expected to be a few unities at convergence, and the values of ϕ_2 to be zero. In practice, we used $k = 10$ so that the penalty function becomes significant when the maximal torque exceed the threshold more than 10 percent. That is why it is not ensured that the result will strictly respect the threshold. To ensure it, or reduce the margin, the penalty value should be increased but it may prevent the convergence of the optimization**]**.

2) Other penalties: Undesirable side effects may appear while maximizing E. Therefore, we had to refine the fitness function to penalize them. First, to be realistic, we need to take into account the self-collisions. However, they are

undesirable since they could cause erosion to the real leg. We thus add a penalty term in the fitness function penalizing self-collision. **[ToEdit:** At each time step, the normal self-collision forces are added up together, and the maximum of these sums (for every time steps) is recorded. The self-collisions are thus penalized similarly to the maximal torque, but the threshold is zero. The corresponding penalty function is then added to the function ϕ.**]** Secondly, it may happen that, when compressed, the leg buckles and leaves the xy-plane. It is penalized too, **[ToEdit:** comparing the maximal distance of the points in the mesh to the xy-plane, to a threshold, around 20 percents of the thickness of the leg. Similarly, the corresponding penalty function is added to the function ϕ.**]**

Finally, as mentioned in Section III, we wanted to identify a "reliability criteria". In practice, we used the total integral of the relative power during the whole rotation. This value should be close to zero as the damping is underestimated (see above) and the integration scheme (semi-implicit Euler) should only lead to small energy loss. But, in practice, if for some reason the scene is not reliable and cannot be correctly simulated, this value is far from zero. In such case, penalization allows to reject these shapes from the optimization.

At that point, we can define the fitness function. Let $\mathbb{F}$ be the set of our admissible shapes. We defined above a fitness function $\phi : \mathbb{F} \to \mathbb{R}$. The space $\mathbb{F}$ is parametrized by the y-coordinates of the pair of control points, and their distance to the y-axis, for the external and internal splines. In order to resize the parameters, we actually impose bounds to the distances, and the y-coordinates are relatively uniformly distributed between two ends. These ends are fixed for the external spline, but free for the internal one. Thereby, we get a function $[0,1]^n \to \mathbb{F}$, where n is related to the number of control points, and consequently a function $\Phi : [0,1]^n \to \mathbb{R}$, that can be minimized by the CMA-ES algorithm. For this, we have to choose CMA-ES parameters like population size and searching radius, that will not be discussed here.

C. The physical test bench

In order to validate our numerical results, we set up a test bench. **[ToEdit:** Since the motion in the numerical simulation is very slow, in order to underestimate the damping, we first made a quasi-static experiment, evaluating the torque in many positions, to estimate the maximal torque, and check that it is consistent with the numerical evaluation. Readers should notice that because of this bias, numerical and physical experiments do not have the same behavior in the video.**]** The results are summarized in Table II (in $cN.m$). We used a system of two pulleys with a belt to multiply the maximal torque delivered by the motor. However, it appears that the test bench was not strong enough to actuate the shapes whose maximal torque were supposed to be at least $0.20N.m$. The maximal torque is reached while the compression phase of the shape. **[ToEdit:** Unfortunately, we could not reproduce exactly the numerical experiment, because we did not manage to get a slow constant speed. Consequently, we did not evaluate the energy transfer.**]**

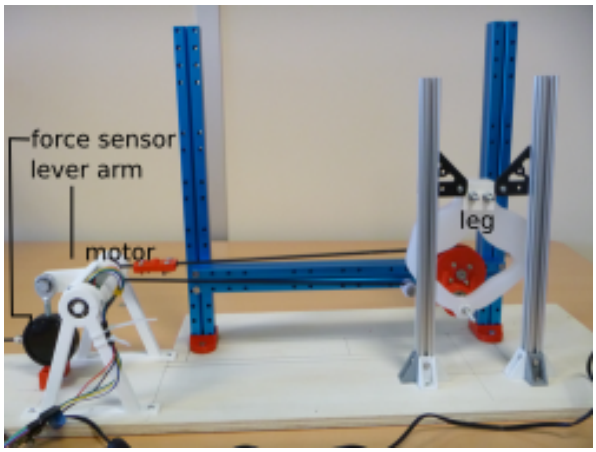

Fig. 3. The physical test bench. On the right we can see the tested leg and on the left the motor and the force sensor.

V. Results

We run several experiments with different maximal torque thresholds. The results we got are gathered in the following tables. **[ToEdit:** For each experiment, we started from the same initial handmade shape. This initial shape is defined by 4 free pairs of control points. However, the number of control points is constant during the experiment. Thus, in case we want to optimize the shape with 9 pairs of control points, the algorithm first adds some pairs, with very little modification of the shape**]**. The two metrics in our experiments are the maximal torque and the amount of energy released (in cJ and $cN.m$).

A. Simulation's results

In Figure 4 we expose the results we got when varying the torque threshold and the number of spline's pairs of control points. Numerical results are presented in Table I.

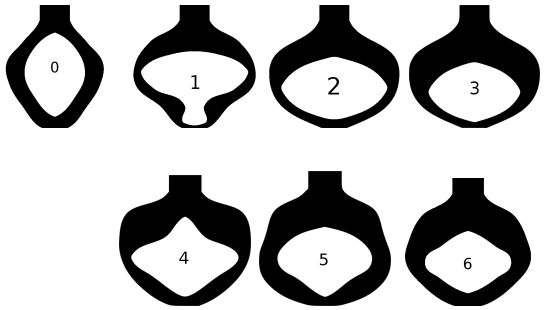

Fig. 4. Initial shape (labelled 0) for the optimization and different shapes (labelled 1 to 6) obtained by numerical experiments (see table I)

From the results, we can see that the torque thresholds are reached. **[ToEdit:** In some (not reported) experiments, it may happen that they are overtaken, typically of 10 percents, because the penalty function is such that it becomes greater

Torque Threshold	Max. Torque	Energy	Nb of Pairs	Shape
initial shape	30	40	4	0
10	9.9	19	9	1
10	10	20	4	2
15	15	32	9	3
15	15	32	4	4
25	25	46	9	5
30	29	53	9	6

TABLE I

THE DIFFERENT TARGETED TORQUE THRESHOLDS AND THE RESULTS OBTAINED FROM THE ALGORITHM AS WELL AS ENERGY RELEASED FOR DIFFERENT NUMBERS OF PAIRS OF CONTROL POINTS (UNIT: $cN.m$, cJ)

than 1.0 only when the threshold is overtaken of 10 percents. Refining the penalty function could correct this bias]. We also noticed that the quality of the result is independent of the complexity of the geometry. Poor geometries, encoded by 4 pairs of free control points reach the same level than those encoded by 9.

B. Test bench results

To get insight on the approach, we tested the shape produced by our algorithm with a real test bench. **[ToEdit:** As previously mentioned, we did not reproduce the numerical experiment with the test bench, since we did not manage to get a constant slow speed. Instead, we just evaluated the torque in many positions to check the coherency on the measures of maximal torques. Each experiment was repeated a few times, and the extremal measures are reported.]

Exp. (cN.m)	Min Measure (cN.m)	Max Measure (cN.m)
15	12.9	15.9
10	9.4	10.5
5	4.0	5.8

TABLE II

ON THE LEFT ARE THE EXPECTED TORQUES, AND ON THE RIGHT THE EXTREMAL MEASURES.

VI. DISCUSSION

Two points of the method and the results need to be discussed.

- The definition of the fitness function has obviously an important impact on the results. It is important to penalize all the constraints to avoid unexpected behaviors. Yet, these constraints are sometimes not obvious at the beginning of the design process. For instance, in our case, we have run several optimizations before getting the final fitness function, in particular because we have progressively added new constraints.
- One of the good properties of the FEM models is that the increase in mesh resolution converges towards a single solution. However, in the particular case of the leg studied in this article, it happens that the optimal solutions are often around forms, which, when they deform during the imposed motion, are at the limit of the self-collision. Then, a small change in mesh resolution can create a self-collision when there was none before, which has a big impact on the fitness function.

VII. CONCLUSION AND FUTURE WORK

In this paper we present our work on soft robots optimization. To make this possible we presented how we model soft robots' shape using a procedural approach and how it interacts with the simulation framework to evaluate the fitness function of the generated shape.

Several points were raised during the discussion and there are many ways to improve this rather preliminary work. In the future, we plan to integrate tools to facilitate the implementation of formatting functions and to provide a solution to better deal with auto-collision cases. In addition, we would like to test the method for more difficult design tasks and therefore further increase the computational requirements. However, it will also be important to keep in mind that in a practical case, it will be necessary for the optimization to be done in reasonable time to actually be used in a design pipeline.

VIII. ACKNOWLEDGEMENT

The authors wish to thank all the members of the DEFROST team for their support, especially B. Carrez, S. Escaida Navarro, A. Kruszewski, T. Morales Bieze, and M. Thieffry who set up the bench test, and E. Coevoet for IT support. The SOFA's developers for their open-source framework. We also thank N. Bredeche for advising us to use CMA-ES, and C. Toure and N. Hansen who helped us to use it. This work has been financed by the Conseil Regional Haut-de-France and by the European Union through the European Regional Development Fund (ERDF).

REFERENCES

[1] 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. 1, 2014.

[2] R. V. Martinez, J. L. Branch, C. R. Fish, L. Jin, R. F. Shepherd, R. M. D. Nunes, Z. Suo, and G. M. Whitesides, "Robotic tentacles with three-dimensional mobility based on flexible elastomers," *Advanced Materials*, vol. 25, no. 2, 2013.

[3] C. Schumacher, B. Bickel, J. Rys, S. Marschner, C. Daraio, and M. Gross, "Microstructures to control elasticity in 3d printing," *ACM Trans. Graph.*, vol. 34, no. 4, 2015.

[4] K. Caluwaerts, J. Despraz, A. Işçen, A. P. Sabelhaus, J. Bruce, B. Schrauwen, and V. SunSpiral, "Design and control of compliant tensegrity robots through simulation and hardware validation," *Journal of The Royal Society Interface*, vol. 11, no. 98, 2014.

[5] W. Chen, X. Zhang, S. Xin, Y. Xia, S. Lefebvre, and W. Wang, "Synthesis of filigrees for digital fabrication," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, 2016.

[6] J. Wu, N. Aage, R. Westermann, and O. Sigmund, "Infill optimization for additive manufacturing - approaching bone-like porous structures," *CoRR*, vol. abs/1608.04366, 2016. arXiv: 1608.04366. [Online]. Available: http://arxiv.org/abs/1608.04366.

[7] T. Hassan, M. Manti, G. Passetti, N. d'Elia, M. Cianchetti, and C. Laschi, "Design and development of a bio-inspired, under-actuated soft gripper," in *Proceedings of the Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015.

[8] 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, 2011.

[9] R. V. Martinez, C. R. Fish, X. Chen, and G. M. Whitesides, "Elastomeric origami: Programmable paper-elastomer composites as pneumatic actuators," *Advanced Functional Materials*, vol. 22, no. 7, 2012.

[10] A. Villanueva, C. Smith, and S. Priya, "A biomimetic robotic jellyfish (robojelly) actuated by shape memory alloy composite actuators," *Bioinspiration & biomimetics*, vol. 6, no. 3, 2011.

[11] 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, 2014.

[12] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, and S. Cotin, "Sofa: A multi-model framework for interactive physical simulation," in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*. 2012.

[13] E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt, and C. Duriez, "Software toolkit for modeling, simulation and control of soft robots," *Advanced Robotics*, 2017.

[14] G. Allaire, *Conception optimale de structures*. Springer Berlin Heidelberg, 2006.

[15] H. Zhang, M. Y. Wang, F. Chen, Y. Wang, A. S. Kumar, and J. Y. H. Fuh, "Design and development of a soft gripper with topology optimization," *IROS*, 2017. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8206527.

[16] N. Cheney, J. Bongard, and H. Lipson, "Evolving soft robots in tight spaces," in *Proceedings of the Conference on Genetic and Evolutionary Computation*, 2015.

[17] R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung, "Make It Stand: Balancing shapes for 3D fabrication," *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, vol. 32, no. 4, 2013.

[18] V. Megaro, E. Knoop, A. Spielberg, D. I. W. Levin, W. Matusik, M. Gross, B. Thomaszewski, and M. Bächer, "Designing cable-driven actuation networks for kinematic chains and trees," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2017.

[19] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.

[20] A. Henrot and M. Pierre, *Shape Variation and Optimization: A Geometrical Analysis*. European Mathematical Society, Mar. 2018, ISBN: 3037191783.

[21] P. Bernardoni, P. Bidaud, C. Bidard, and F. Gosselin, "A new compliant mechanism design methodology based on flexible building blocks," in *Smart Structures and Materials 2004: Modeling, Signal Processing, and Control*, International Society for Optics and Photonics, vol. 5383, 2004.

[22] S. G. Ficici, R. A. Watson, and J. B. Pollack, "Embodied evolution: A response to challenges in evolutionary robotics," in *Proceedings of the eighth European workshop on learning robots*, 1999.

[23] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, "Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding," *SIGEVOlution*, vol. 7, no. 1, 2014.

[24] J. Hiller and H. Lipson, "Dynamic Simulation of Soft Multimaterial 3D-Printed Objects," *Soft Robotics*, vol. 1, no. 1, 2014.

[25] L. Brodbeck, S. Hauser, and F. Iida, "Morphological evolution of physical robots through model-free phenotype development," *PloS one*, vol. 10, no. 6, 2015.

[26] J. Rieffel and D. Sayles, "Evofab: A fully embodied evolutionary fabricator," in *Evolvable Systems: From Biology to Hardware*, 2010.

[27] S. Lefebvre, "Icesl : A gpu accelerated modeler and slicer," in *18th European Forum on Additive Manufacturing*, 2013.

[28] M. Keeter, "Hierarchical volumetric object representations for digital fabrication workflows," Massashusetts Institute of Technology, Jun. 2013.