Metamaterial Mechanisms

Alexandra Ion, Johannes Frohnhofen, Ludwig Wall, Robert Kovacs, Mirela Alistar, Jack Lindsay, Pedro Lopes, Hsiang-Ting Chen, and Patrick Baudisch
Hasso Plattner Institute, Potsdam, Germany
{firstname.lastname}@hpi.de

ABSTRACT
Recently, researchers started to engineer not only the outer shape of objects, but also their *internal microstructure*. Such objects, typically based on 3D cell grids, are also known as metamaterials. Metamaterials have been used, for example, to create materials with soft and hard regions.

So far, metamaterials were understood as materials—we want to think of them as *machines*. We demonstrate metamaterial objects that perform a mechanical function. Such *metamaterial mechanisms* consist of a single block of material the cells of which play together in a well-defined way in order to achieve macroscopic movement. Our metamaterial door latch, for example, transforms the rotary movement of its handle into a linear motion of the latch. Our metamaterial Jansen walker consists of a single block of cells—that can walk. The key element behind our metamaterial mechanisms is a specialized type of cell, the only ability of which is to shear.

In order to allow users to create metamaterial mechanisms efficiently we implemented a specialized 3D editor. It allows users to place different types of cells, including the shear cell, thereby allowing users to add mechanical functionality to their objects. To help users verify their designs during editing, our editor allows users to apply forces and simulates how the object deforms in response.

Author Keywords
metamaterial; mechanism; fabrication; 3d printing;

ACM Classification Keywords
H.5.m. [Information interfaces and presentation]: Misc.

INTRODUCTION
Researchers in HCI have explored the use of personal fabrication tools, such as 3D printers [33] to help users design the external shape of 3D objects [36]. In order to add functionality to 3D printed objects, researchers integrated electronics [28], even printed optics [37], or loudspeakers [11].

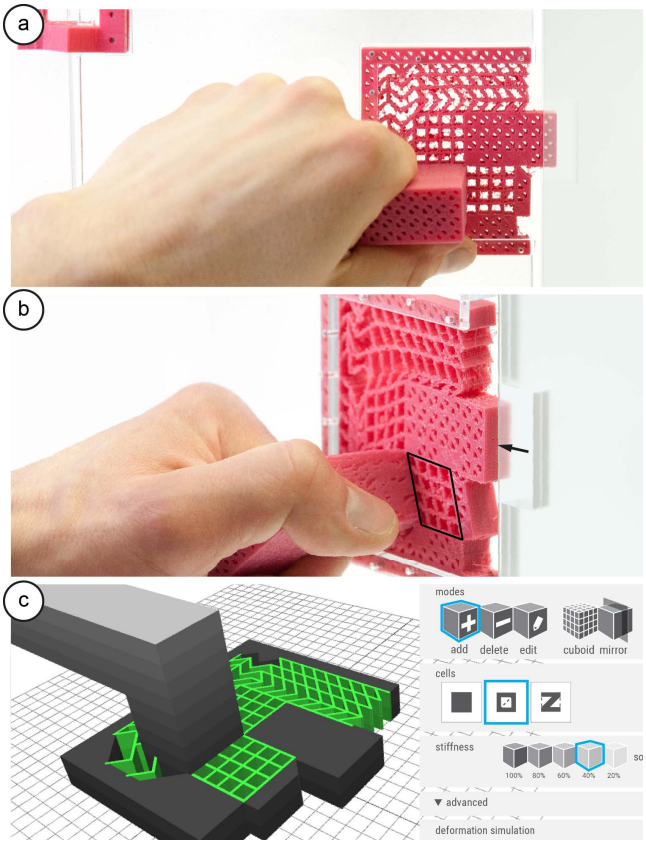

Figure 1: (a) This door latch is implemented as a metamaterial mechanism; it consists of a single block of material based on a regular grid of cells that together implement handle, latch, and springs. (b) Turning the handle causes the central hinge array to deform and to pull the latch inwards, which unlocks the door. (c) We created this mechanism in our custom editor. Here, we placed two hinge arrays that mechanically couple the handle to the latch, and cells that couple to the doorframe.

Researchers also started to design the inside of 3D objects by changing the structure of the 3D printed object itself. Initial projects optimized only a single parameter, such as the object's strength-to-weight ratio [14] or the position of the object's center of mass [26].

Recently, researchers started to push internal structures even further and created objects that consist internally of large numbers of 3D cells organized on a regular grid [30]. Since these objects allow each cell to be designed differently, the resulting objects literally offer thousands of degrees of freedom. These types of structures have also been referred to as *metamaterials*. Metamaterials are artificial

Authors' copy.
Published in Proceedings of *UIST '16*, October 16 - 19, 2016, Tokyo, Japan.

structures with mechanical properties that are defined by their usually repetitive cell patterns, rather than the material they are made of [25].

Based on this concept, researchers have created objects with unusual behaviors, such as metamaterials that collapse abruptly when compressed [22], that shrink in two dimensions upon one-dimensional compression [6], or objects that mix layers of soft and hard cells in order to emulate different materials [3].

So far, metamaterials have been understood as materials. The main contribution of this paper is that we want to think of them as *machines*.

In this paper, we push the concept of metamaterials further by creating objects that allow for *controlled directional movement*. This allows users to create objects that perform mechanical functions. Our objects thereby implement devices that transform input forces and movement into a desired set of output forces and movement—also known as *mechanisms.*

METAMATERIAL MECHANISMS

Figure 1a shows an example of a metamaterial mechanism: a door latch mechanism. Its interior is a regular grid of 3D cells; however, the cells are of different types. Figure 1b shows how applying a force causes the cells to deform in a controlled way, thereby performing the intended mechanical function. In the example, rotating the door handle causes cells inside of the object to deform, ultimately pulling the latch towards the left and thereby unlocking the door.

While most of the object consists of *rigid* cells (cells that are reinforced with a diagonal), the object also contains several rectangular regions of cells that lack such a diagonal reinforcement. These are the key to creating mechanisms, as they are able to deform in a very specific way: when subjected to an external force, these cells shear and thereby apply a force to their neighboring cells. We will discuss throughout the remainder of this paper how this basic principle allows creating mechanisms.

Metamaterial mechanisms are simple. While the traditional door latch mechanism shown in Figure 2 consists of several parts, including an axle, bearings, springs, etc., the metamaterial door latch in Figure 1 consists of a single block of material, as it is groups of cells inside the object that perform the mechanical function.

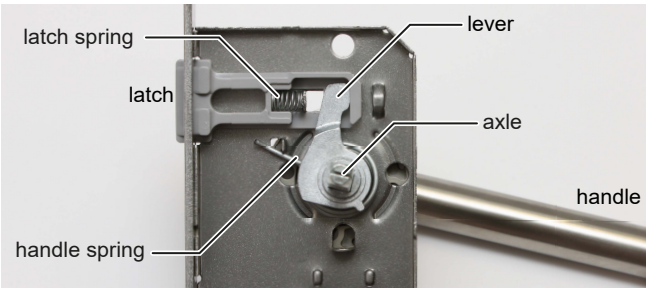

Figure 2: This traditional door latch design consists of many parts, thus requires assembly.

While in previous work metamaterials were typically generated by scripts [22, 25], creating mechanisms requires a dedicated design/engineering process that we argue is best performed by means of an interactive editor. Figure 1c shows a preview of the custom 3D editor we created specifically to allow users to create and modify metamaterial mechanisms. It contains a range of functions that help users assemble specialized cells into basic mechanisms and to assemble such basic mechanisms into more complex mechanisms and simple machines. Using this editor, we have created a series of demo objects including the door latch from Figure 1, a Jansen walker, or a functional pair of pliers. Our examples were printed on an *Ultimaker 2* 3D printer using *NinjaFlex* filament (in pink). The basic mechanisms (hinge, four-bar) were laser cut from 3 mm rubber foam (in black).

CONTRIBUTION, BENEFITS, AND LIMITATIONS

Our main contribution is the concept of metamaterial mechanisms. Our main software contribution is a specialized editor that allows users to create them.

We extend the research field of metamaterials by contributing a general-purpose approach to creating mechanisms. Mechanisms are a new genre of metamaterial structures that is of higher complexity and that exploits more degrees of freedom than previous work in this field, and that allow metamaterials to tackle problems they have traditionally not been able to address.

Compared to traditional multi-part mechanisms, metamaterial mechanisms offer several benefits. (1) The resulting devices consist of a single part. They can thus be created using particularly simple fabrication processes, such as single-material 3D printers (e.g., FDM printers). (2) As they consist of a single piece, they require no assembly. (3) Since the movement is performed by deformation there is virtually no friction, no need for lubrication, and thus for maintenance [9].

However, the resulting designs are also subject to limitations. Adding more cells increases the stiffness, and as a result, metamaterial mechanisms are not suitable for mechanisms that are to be operated with very small forces. Furthermore, our approach is unable to produce continuous rotation. Objects such as the Jansen walker, for example, thus require a separate axle. Also, cell designs are limited by the quality of the 3D printer. In particular, shear cells work best if their internal hinges are thin, which requires high-resolution 3D printers. Finally, while our editor vastly simplifies the creation of metamaterial mechanisms, any type of mechanical engineering requires experience—and metamaterial mechanisms are no exception here.

RELATED WORK

Our work builds on previous work in interactive personal fabrication, in particular on techniques that modify the internal structure of 3D printed objects, multi-material printing, and mechanical metamaterials.

Personal fabrication and user interaction

Personal fabrication devices such as 3D printers and laser cutters allow users to fabricate personalized physical objects. Besides fabricating common decorative objects from rigid plastic, researchers showed how to fabricate a variety of objects, such as printing soft teddy bears [10], optical sensors [37], or loudspeakers [11].

While users traditionally use CAD software to model objects that are to be fabricated, researchers started to explore tools that help users design objects by sketching [16, 27] directly on the machine [7, 21], or by using physical objects [8, 36].

Researchers in HCI and computer graphics explored how to define the *behavior* of printed objects. In particular they changed the rigid inside structure of objects in order to optimize the object's strength-to-weight ratio [14], to balance the object (e.g. "make it stand" [26]) or to allow it to spin [2]. Savage et al. help adding electronic sensing capabilities to printed objects [28] by creating tubes and holes inside an object [29].

One contribution of our work is to allow users to interactively define the mechanical behavior of 3D printed objects by arranging many cells. Since this requires users to interact with discrete elements, we provide a voxel-like editor that allows users to edit metamaterials efficiently.

Compliant mechanisms and flexures

Our work builds on deformable struts that transfer forces. Similar mechanical structures have been examined in the context of compliant mechanisms, i.e., monolithic structures that transfer motion, force and energy without traditional hinges, but using flexure hinges (thin regions that are bendable and allow for rotation) [9].

Multi-material printing

The work presented in this paper explores how to create objects the behavior of which varies across the object's geometry. A more traditional solution to this question is the use of multi-material 3D printers (e.g. Objet Connex), which allow printing different regions of an object from different materials in order to achieve different mechanical properties, such as elasticity. Recently, researchers developed a printer that prints hydraulics [15]. Vidimče et al. proposed a programmable rendering pipeline which integrates the printing material into shaders [34]. Researchers also optimized material distribution, they varied it over an object's volume to match pre-defined target poses [32], or mathematically modeled the material distribution [12, 13] based on, e.g., MRI data for medical purposes [5]. Such materials are also referred to as functionally graded materials.

However multi-material printers are more complex, the resulting objects are harder to recycle, and limited to the omnidirectional mechanical behavior of traditional materials. This led researchers into exploring how to obtain mechanical behavior by arranging a single material.

Mechanical metamaterials

Metamaterials are artificial structures, usually repetitive patterns. Their unusual properties originate from their geometry, rather than the material they are made of [25].

Researchers in the fields of mechanical engineering and material science discovered designs for cell structures to create material properties that can only be achieved using metamaterials. For example, when conventional materials are stretched, they compress in the orthogonal direction. However, researchers designed cell structures that let the material stretch in both directions (so-called *auxetic* behavior [6, 18, 31]). Metamaterial structures were also designed to create materials that "pull" in the direction of compression rather than resisting it (negative stiffness materials [22]) as usual materials would. Researchers created materials that damp (large energy absorption capabilities [20]), and materials that behave like "liquid solids", i.e., they are hard to compress but easy to deform (pentamode metamaterials [17]) Such metamaterial structures were typically designed by a mathematical formulation of the desired behavior or empirical experimentation.

The evolution of metamaterials has been further driven by recent advances in high-resolution 3D printing. An example of this is a printed material by Bickel et al. with structured pores that lowers the material's resistance to uniform compression, i.e., its overall stiffness [3]. Chu et al. went further by introducing a set of unit cells and an optimization method that creates unit cells to match a specified stiffness distribution [4]. Recently, Schumacher et al. and Panetta et al. focused on elastic properties which they computationally varied across an object [24, 30].

Metamaterial editing

One of the contributions of our work is our interactive metamaterial editor. So far, metamaterials have mostly been created by scripting them (e.g. in matlab [22, 25]) or the structures were generated after specifying forces or displacements on the object's boundary [30]. However, generating the microstructure based on forces that are applied on the outside of an object only allows for interpolating elasticity parameters, i.e., creating gradients between soft and rigid cells. Similarly, there are other editors that optimize the stiffness distribution of an homogenous internal structure [1, 19, 23] based on force vectors.

Integrating mechanisms into metamaterials is more complex than varying the local compliance of 3D printed objects. Creating metamaterial mechanisms involves directional compliance of cells as well as specific spatial arrangements of such blocks to enable the functionality of the machine. We provide a 3D editor that allows users to create and simulate cell structures interactively.

MEMBERS & HINGES BASED ON CELLS

The mechanism behind the door latch in Figure 1 ultimately consists of rigid regions (such as the latch itself), which we will refer to as *members* and compliant regions that implement *hinges* (such as the region right of the handle). Members and hinges both consist of cells on an evenly

spaced grid. This is the general schema behind metamaterial mechanisms.

In this section, we take a closer look at these two basic elements and illustrate how to use them to build mechanisms.

Members from cells

Members consist of rigid cells. A rigid unit cell is reinforced with a diagonal, as shown in Figure 3. The diagonal reinforces the cells against shear forces and thus makes them rigid. Adding two diagonals is possible, but not necessary, hence we use only one diagonal for our member structures in order to save material.

Figure 3: (a) The basic grid structure allows reinforcing rigid member cells (b-c) with one diagonal or (d) two diagonals.

While we will focus our discussion on only two types of cells, namely stiff member cells and shearable hinge cells, our research can be combined with any other system of metamaterials already done as long as they are on a cube grid. Just to pick one example, we can make cells soft or hard [4] by weakening their beams so as to allow the cells to compress more easily, or reinforcing their beams to make the cells stiffer (Figure 4).

Figure 4: Member cells of increasing stiffness.

Hinges

We can create a (naïve) hinge by connecting two cells diagonally as shown in Figure 5a. Here the material connecting the two cell blocks forms a thin flexible hinge made from the same material as the two rigid members it connects. As common for this type of contraption, we will refer to it as a *living hinge*.

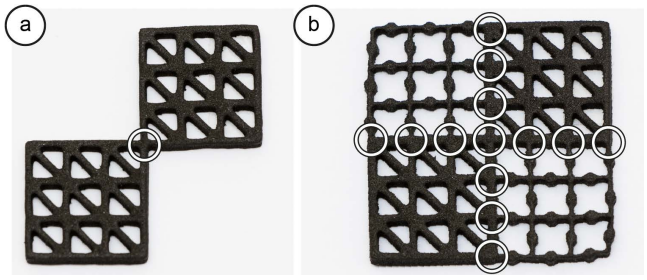

Figure 5: (a) A naïve living hinge, (b) reinforced with two arrays of hinges. To showcase the deformation, we laser cut these structures from rubber foam.

Next we add reinforcement to the hinge. Metamaterial cells can, at least in theory, be arbitrarily small—the hinge connecting two cells can thus arbitrarily thin, making mechanisms based on such naïve living hinges fragile, i.e., pulling them apart with very little force will rip them off.

We address the issue by extending the living hinge into an array of living hinges, as illustrated by Figure 5b. We call them *hinge arrays*. Figure 6 shows that the array flexes in concert with the main living hinge. The hinge array on the inside of the rotation complies by compressing in a shearing motion; the hinge array on the outside complies by stretching in a shearing motion. All hinges together perform a rotation around the original living hinge.

Figure 6: The reinforced hinge in action. Note how the hinge array deforms as one while the rigid members remain undeformed. (Cells with gray tinted backgrounds are anchored to the ground, as indicated by the lock symbol. We are pushing in the direction of the arrow using the blue rod).

However, when this reinforced hinge is stretched as shown in Figure 7a, its hinge arrays evade the tension by shearing. As a result, all tension rests on the central living hinge, causing it to break. As shown in Figure 7b, we address this by adding a border of rigid cells to the hinge arrays. These borders force all living hinges to follow the motion of the central living hinge, thereby support the main hinge by assuming part of the load, which distributes the tension across all the hinges.

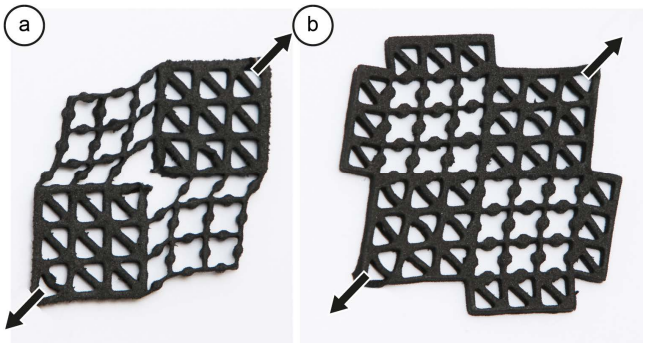

Figure 7: (a) When this hinge array is stretched as shown, the central hinge breaks. (b) We address this by adding borders of rigid cells; they help distribute the tension more evenly.

Tensile strength increases linearly with the diagonal of the hinge array, because tension applies to the shortest path of hinges to tear in order to separate the two members. The tensile strength of the hinge array in Figure 7, for example, is 7× larger than of a single hinge.

Figure 8 shows the resulting stress distribution after reinforcement (simulated using Autodesk Fusion 360; light colors indicate regions of high mechanical stress).

Hinge arrays also allow tuning the bending stiffness of a hinge, e.g., to make sure the door latch in Figure 1 offers appropriate resistance when pushed down, as well as sufficient force to push the handle back up. Bending stiffness increases proportional to the surface of the hinge array, i.e.,

(width+1) × (height+1), as all individual hinges need to be bent. The hinge array in Figure 7, for example, uses 2 hinge arrays of (3+1) × (3+1) = 32 hinges, thus it takes 32 times more force to bend it than the single hinge in the center.

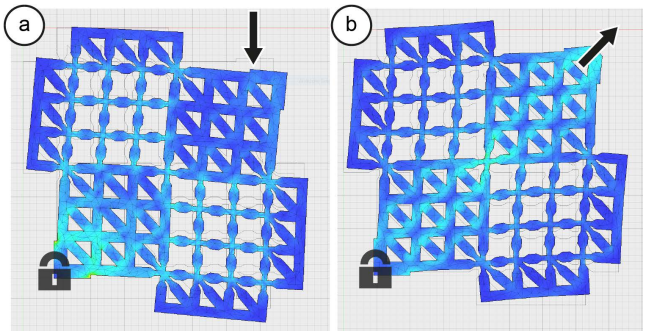

Figure 8: Resulting stress distribution on (a) rotation and (b) tension after reinforcement.

Four-bars implement parallel motion

In Figure 9, we use the concept of rigid members connected by hinge arrays to construct a shearing frame of four links, also known as *four-bars*.

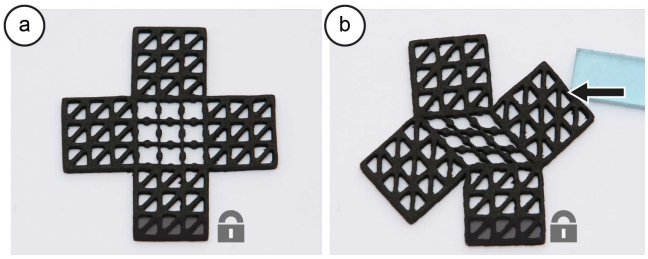

Figure 9: A metamaterial *four-bar*.

Four-bars consist of four members attached to the *opposite* sides of a hinge array (unlike the reinforced hinge, which consists of two members attached to *adjacent* sides of a hinge array). It then contains the two opposing members to parallel motion. Note how the rigid member that is pushed to the left in Figure 9b rotates, while the top member moves in parallel with the rigid area underneath it.

In Figure 10 we exploit this property of the four-bar in order to create a functional pair of pliers. The hinge array in the center acts as a four-bar that connects the right handle with the left bracket *and* the left handle with the right bracket, yet allows the two sides to move with respect to each other—somewhat similar to how the axle in a traditional pair of scissors allows the two handle and blade elements to move with respect to each other.

Cascades of hinge arrays form a scissors mechanism

We chain multiple four-bars to create a scissors mechanism, which is a linkage that arranges links in a criss-cross pattern and is used. Figure 11 shows how we implement a pantograph by chaining multiple metamaterial four-bars. The pantograph holds two pencils. While one pencil is moved by the user to draw, the second pencil moves along and replicates the user's drawing.

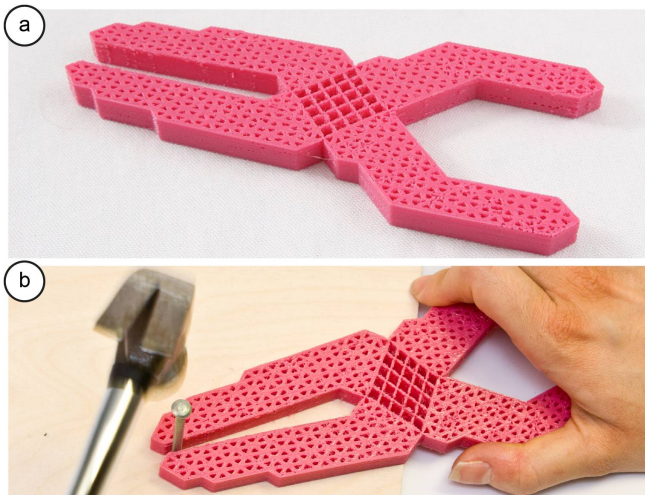

Figure 10: (a) This pair of pliers is based on a single metamaterial four-bar. (b) When the user applies a force to the handles, the hinge array in the center transmits this force to the brackets, and the pliers close.

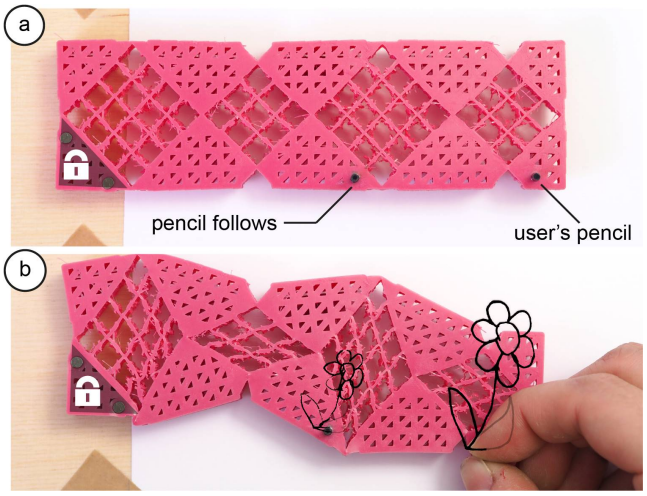

Figure 11: The user draws a flower using (a) our metamaterial pantograph, which (b) replicates the flower as the user is drawing.

UNDER THE HOOD: THE SHEAR CELL

So far, we only introduced the rigid cell. However, without explicitly mentioning it, the previous section introduced a second type of cell: a cell designed to shear. Unlike the rigid cell, this *shear cell* is designed to deform when a force is applied, more specifically to *shear* (Figure 12).

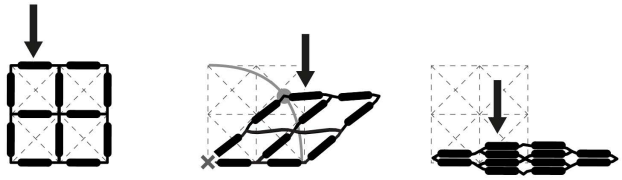

Figure 12: When a shear cell in this 2 × 2 block is subject to compression forces, it complies by shearing on a circular trajectory until its members are packed tightly.

As illustrated by Figure 12, shear cells compress when a force is applied. The cell itself consists of miniature members and miniature living hinges—similar to the macro-

scopic structures we build from them. When a force is applied, the hinges start to bend, while the members largely maintain their shape (increasing the thickness of a beam results in a cubed increase in stiffness). The shear cell resists the external force based on the springiness of its hinges, until eventually the members touch and the resulting tightly packed block of cells is not compressible anymore apart from the compressibility of the material itself.

Since shear cells are based on an unit cell of our grid, they are always oriented along the cardinal directions. To allow handling forces and shearing along additional orientations, we introduce rotated hinges. As illustrated by Figure 13, we obtain a rotated design by combining groups of 2 × 2 cells, each of which contains (only) a diagonal.

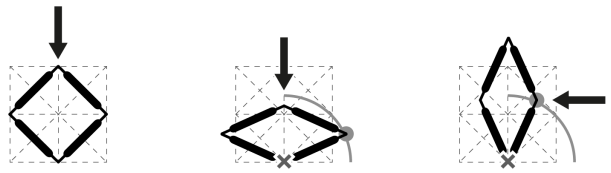

Figure 13: The rotated shear cell.

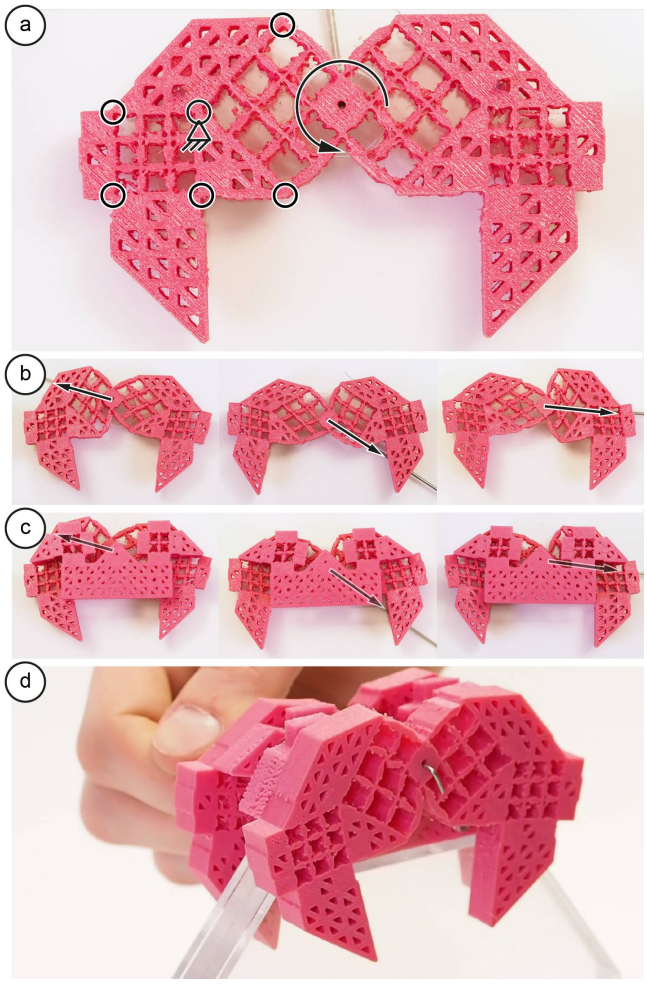

Figure 14: (a) Each leg of the Jansen walker has 6 hinges. (b) As the center is actuated by a crank, the legs deform in a walking motion. (c) We constrain two hinges to rotate, but not translate, using a layer of hinges. (d) The complete walker.

Combining regular and rotated hinges allows us to engineer additional mechanisms, such as the leg pair of a Jansen walker mechanism that is shown in Figure 14. Each leg consists of 6 hinges that make up the organic walking motion when the center is actuated on a circular path using a crank. One hinge per leg, however, needs to be constrained so as to only rotate, not translate. We implement this by adding a layer of two metamaterial hinges for the two legs that are connected to stay at constant distance, but decoupled to move independently from each other.

Finally, we have created a third type of shear cell by leaving out one of the Cartesian members. It is shown in Figure 15 and we refer to it as *z-cell*.

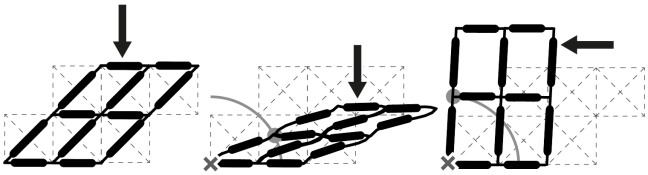

Figure 15: "z-cells" shear on a larger circular path than regular shear cells, which is also rotated by 45°.

Z-cells are very similar to regular shear cells, except that they are "pre-sheared", which results in three differences: (1) they have an asymmetric shearing behavior (range of 135° vs. 45°), (2) they expand as they shear against their pre-sheared direction, and (3) compress in the other direction.

Figure 16 shows a switch based on z-cells. The button stays parallel while going down to hit the contact point reliably. A similar switch implemented from only soft and hard cells requires making very thin beams to allow the switch to deform. However, this does not implement a well-defined directional movement and closing at the contact point is not ensured.

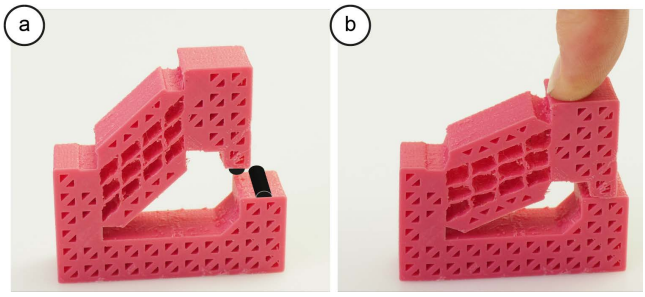

Figure 16: (a) This switch is designed based on z-cells, which allow it to compress in a controlled motion and (b) close reliably at the black contact point.

Padding

Metamaterial mechanisms enable directional movement of regions of material, i.e., they occupy space when they are in their deformed state that was not occupied in their undeformed state. Most of our examples—the pliers, the walker, or the pantograph—are self-contained and can deform independently. However, the door latch is not self-contained but embedded in a rigid door. This requires appropriate clearance for the respective region to move into.

We surround the door latch shown in Figure 17 with what we call *padding* areas. Padding areas consist of cells that are not intended to transmit any forces, but rather to receive the forces exerted by the main mechanism. The padding serves two purposes here: (1) to connect the movable door latch with the rigid door and (2) to increase its stability, i.e., to prevent the mechanism from buckling out of plane and add support against operation in unintended directions. In case the handle is pushed up, the z-cells above the latch prevent the latch from moving outward (to the right).

Figure 17a shows that after the door latch mechanism is deformed we see two types of transformations on the outside edges: (1) a parallel movement that is indicated by the blue line and (2) a rotational movement of the edge indicated by the green line. Additionally, all the edges undergo a vertical and horizontal translation. For all outside edges we need to allow translation and for some we need to additionally allow rotation. Therefore, the latch mechanism in Figure 17b uses two types of padding cells, chosen to allow for the intended movement, yet to suppress other types of movement.

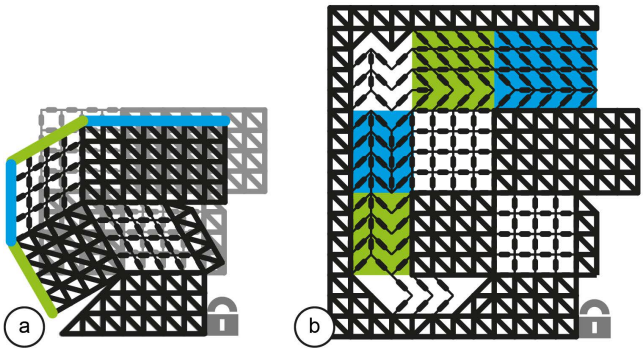

Figure 17: (a) When the door latch deforms, the edges labeled in blue remain parallel, while the green edges rotate. (b) Consequently, we reinforce the green regions using padding cells that allow for translation and rotation deformation, while we reinforce the blue regions to only allow for translation.

Z-cell padding absorbs parallel movement, like the movement of the latch. Z-cell padding allows a member on one side to translate in 2D with respect to a member on the other side and keeps the two members parallel. Since z-cells shear on a circular path that is rotated by 45°, they allow for a vertical extension of $\sqrt{2}-1$ cells and a horizontal displacement of 1 cell. We simply stack enough layers of z-cells to compensate for the downward and inward movement of the latch.

Note that we added one row of z-cells that is flipped. Figure 18a shows that this additional row is necessary since the shear cells underneath the latch and the z-cells above the latch travel on opposing circular trajectories and cannot pass each other. The flipped row of z-cells however moves upwards (Figure 18b) and gives space for the lower row of z-cells to pass and thus the latch to move.

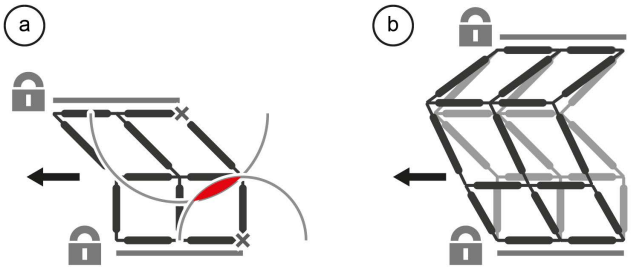

Figure 18: (a) The shear cells cannot pass each other (red area), (b) therefore we add a flipped row of z-cells that compress and give space for the other shearing cells to move.

V-cell padding absorbs rotation. V-cells are similar to the flipped z-cell group, but they additionally allow two opposing members to rotate, as shown in Figure 19. We add this degree of freedom by removing the vertical beam from the z-cells. We use v-cells in combination with z-cells on the rotating edges illustrated by Figure 17b.

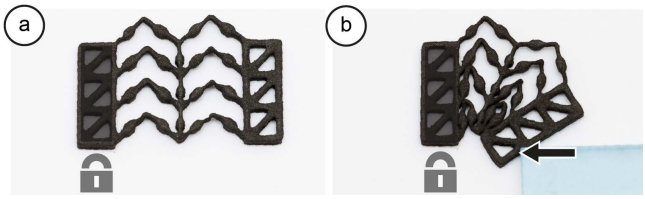

Figure 19: V-cells allow for rotation, as well as for compression and extension.

HIERARCHY OF METAMATERIAL MECHANISMS

In this section, we presented the main elements of metamaterial mechanisms. These elements form the following five-level hierarchy.

1. *Cells* are the lowest-level element of metamaterial mechanisms and perform an elementary mechanical function. We presented shear cells, which implement four-bars on a cell level, as well as z-cells.
2. *Compound cells* still perform an elementary function, yet are implemented using multiple physical cells, such as the rotated shear cell or the v-cell.
3. *Basic mechanisms* are created by repeating cells, in particular the reinforced hinge mechanism and the four-bar mechanism.
4. *Compound mechanisms* consist of multiple basic mechanisms, such as the scissors mechanism implemented in the pantograph.
5. *Machines* perform a mechanical function and consist of one or more basic or compound mechanisms, such as the door latch (Figure 1 and Figure 17), the pliers (Figure 10), the Jansen walker (Figure 14).

METAMATERIAL MECHANISM EDITOR

To allow users to design, fabricate, and test metamaterials containing mechanisms we implemented the specialized editor shown in Figure 20.

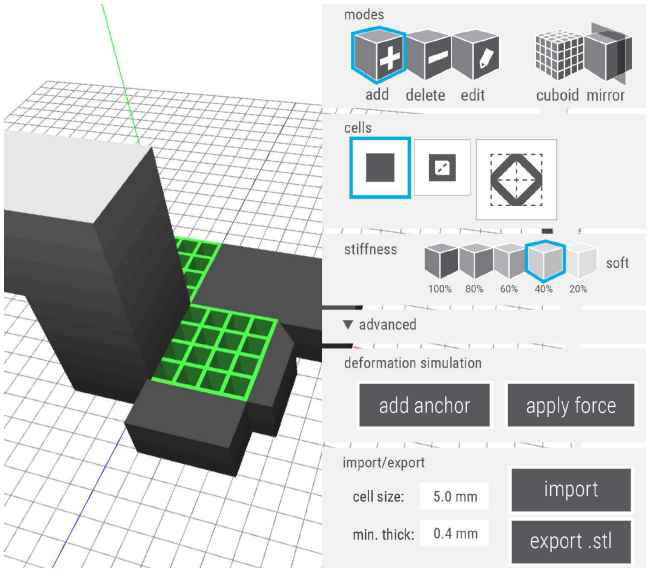

Figure 20: Our editor allows users to edit and simulate the deformation of metamaterial mechanisms interactively.

The main intent behind it is not only to make the editing process more efficient than the more traditional script-based editing, but also to provide users with an overview of their design, encouraging design by trial-and-error.

Our editor is based on interaction techniques known from voxel editors (such as [35]). However, in addition our software also offers specific supports for creating mechanisms, such as tools for drawing hinge arrays, etc. In order to allow users to validate their designs, the editor also allows them to apply forces and see how the object deforms in order to then refine their design directly inside the editor, before exporting to the 3D printer.

Walkthrough

Figure 21 illustrates how we created the door handle.

(a) We start by creating a block of rigid cells using the *add brush* (we can remove cells using the *delete brush*). Here we use the tool in *cuboid mode*, which allows us to draw a filled rectangular region at once by just drawing the diagonal. (b) By adding another two cuboids on top, we create the handle.

(c) We select the *shear brush*. Still in cuboid mode, we paint the central hinge array using a single drag interaction, which causes rigid cells to turn into shear cells. Even though the block of material we painted on is two cells high, the *shear brush* paints cells all the way through—as we can tell from the sidewall now being all green. This is one of the features of this brush: since shear cells backed by rigid cells would still be rigid, thus have no effect, the *shear brush* always cuts shear cells through the entire object.

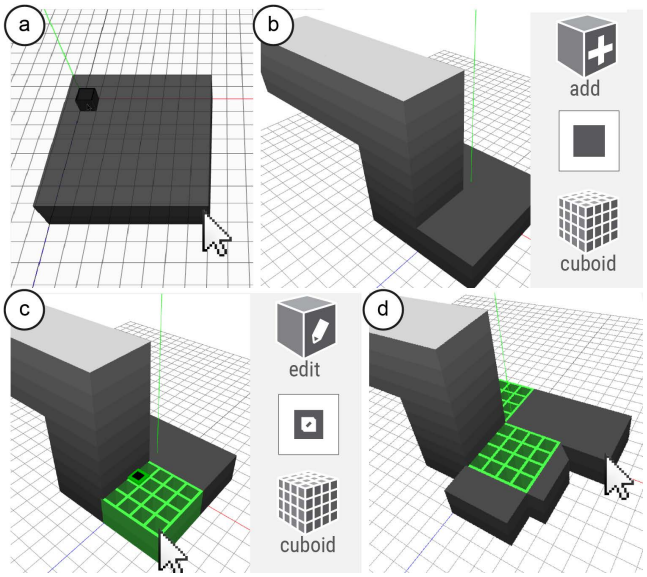

Figure 21: Walkthrough of the creating door latch mechanism. The UI elements on the right show the active tools for the respective interaction steps.

We now verify our design directly from within the editor, as illustrated by Figure 22. (a) We select the *anchor tool* and use it to place a few anchor points at the bottom, indicating that the door latch is here rigidly connected to the doorframe. (b) Now we use the *force tool* to apply a force to the door handle. We attach a force arrow to one of the handle's cell vertices. As we are building up the force by dragging the force tool the system already responds by showing the resulting deformation of our door latch.

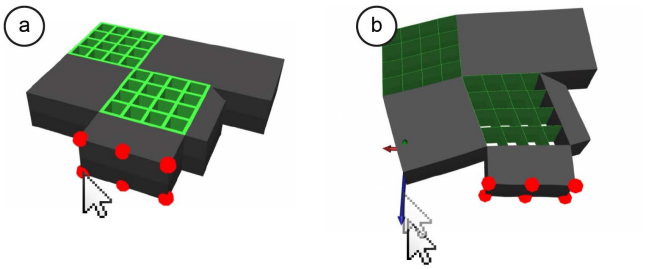

Figure 22: To simulate the deformation in real-time in the editor, (a) users set anchor points and (b) adjust forces using the force tool.

Multiple dimensions

While the door latch mechanism actuates in only two dimensions, our editor also supports placing mechanisms in 3 dimensions. The latch mechanism shown in Figure 23, for example, combines a horizontal hinge array (blue) and a vertical hinge array (green) in order to create a mechanism that users operate by pressing down, sliding over, and releasing.

Our editor color-codes mechanisms automatically according to their orientation in space. This is intended to provide users with a fast overview of the main dimensions of action in their devices and to help recognize hinge arrays from odd viewing angles. In the latch example, green denotes "shearing on the x/z plane" and blue stands for "shearing

on the x/y plane". Analogously, red stands for "shearing on the y/z plane".

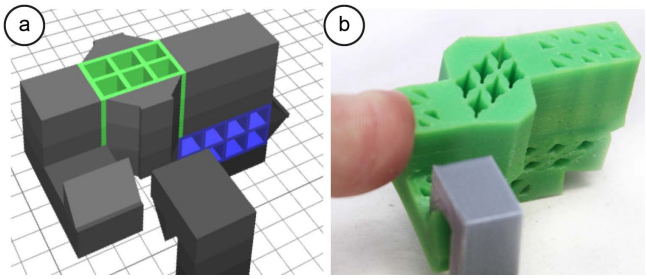

Figure 23: This latch requires the ability to shear on two planes, i.e., on the x/z plane denoted in green, and on the on x/y plane indicated in blue.

Note that hinge arrays can overlap. In this case, cells at the intersection bear the combination of all holes. These cells are rendered as the additive mixture of the involved colors, such as yellow, for cells at the intersection between green and red.

Integration with other metamaterial systems

The shear cell is the main element that enables metamaterial mechanisms. However, to allow for the integration with metamaterials by other researchers, the editor can be extended to allow for other cell types.

In order to allow users to explore their own cell types, we offer the *advanced panel* shown in Figure 24. Users compose cells from individual edges by selecting the respective edges. The editor automatically adds all custom cells used in the current model to the cells panel for quick reuse.

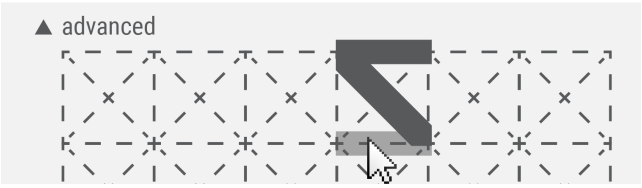

Figure 24: Users compose custom cells by adding individual edges in the "advanced" panel.

Furthermore, users can also create and store groups of cells, for example to create auxetic materials [6, 18], as shown in Figure 25. Since metamaterial mechanisms adhere to the standard structure of 3D cell grids that is common for metamaterials, they integrate with earlier research [24, 30].

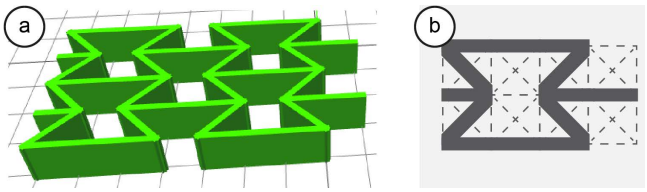

Figure 25: Users add groups of cells, here they create an auxetic material [18] **from a 4 × 2 group of cells.**

SYSTEM IMPLEMENTATION

In the following, we provide details on the internal processes implemented in our metamaterial mechanisms editor.

Import

Users can import mesh geometries directly into our editor. We voxelize the meshes using *binvox*[1] according to the cell size that the user defined.

Editor

Our 3D editor is based on WebGL and uses three.js. Internally, the editor creates a dictionary of cells that can be accessed using their position on the grid. Each cell is defined by the 8 vertices making up its bounding box by and the edges that define how the vertices are connected. Note that not all 8 vertices need to be connected by edges.

All vertices lie on our uniform 3-dimensional grid. To generate the 3D cells' structures, i.e., to generate 3D beams from 1D edges, we apply an offset to the vertices' positions on the GPU. Since WebGL does not offer geometry shaders, we use a vertex shader and pass the offset direction and the cell's position with each vertex. The 8 vertices that form a beam are offset uniformly from the two edge vertices on the GPU. To pass additional information about the color and thickness of beams to the shader, we generate a texture where each pixel holds these data for one cell. The color maps directly and the thickness is encoded in the alpha component. In the shader, every vertex looks up its thickness in the texture and calculates the offsets for the new vertices that render a beam from an edge. This enables us to emulate a geometry shader in WebGL and perform all geometry processing on the GPU, which keeps the user experience of our editor smooth.

Simulation

For simulating the deformation of the user's cell structure, we use the finite elements solver *karamba*[2], which is a plugin-for Grasshopper/Rhinoceros. We implemented a custom C#-Grasshopper-component that receives the mesh data (vertices and edges) and the data for the simulation (anchored vertices, force and vertex where the force applies) via a web socket connection. When the simulation is complete, a second custom component receives the transformed mesh vertices and sends them back to the editor. The vertices are kept in the same order within the array as they were received from the editor. We run the simulation on a separate machine to keep the editor running smoothly.

Maintaining the order of the vertices is important to enable geometry processing on the GPU. In the editor, we generate another texture and store the transformed vertices, where XYZ is mapped to RGB. The shader knows the vertex' undeformed position on the grid and looks up the deformed position in the texture.

Depending on the size of the object that is simulated, solving for the deformation can lead to perceivable delays. To

[1] http://www.cs.princeton.edu/~min/binvox/
[2] http://www.karamba3d.com/

compensate for this, our editor interpolates the deformation while the response from the simulation is pending. To do so, we pass the last force where we received the transformation from the server, and the current force that was submitted to the simulation and interpolate the vertex transformation linearly.

Export

We generate an .stl file for the user that is ready to be 3D printed. Our export is based on OpenJSCAD. In this step, we refine the cell structure from the simplified editor view to our beams with stiff members and thin living hinges. For every edge that belongs to a shear cell, we create a beam with a thick part in the middle. Edges that are part of rigid cells are generated as simple straight beams. Finally, we use OpenJSCAD's built-in render engine, which we invoke directly from our 3D editor to perform the union operations and generate the .stl file.

DISCUSSION ON MATERIAL & DURABILITY

The ideal material for metamaterial mechanisms is (1) very elastic, i.e., goes back to its original shape without deforming permanently and (2) can withstand high forces without breaking. Spring steel fulfills these criteria well, yet is not possible to fabricate using today's 3D printers. Among commonly available 3D printing materials, flexible materials (such as the TPU *NinjaFlex* and *SemiFlex*) fulfill these criteria well, more so than PLA and certain types of photocuring resins (such as the *spot-e* resin).

When we created 2D metamaterial mechanisms using our laser cutter, we obtained best results with rubber foam, more so than with ABS sheets and acrylic, which were not compliant enough. Our metamaterial mechanisms worked across fabrication techniques, since the functionality is determined by the cell structure.

In our experience, material fatigue is not a problem, because we operate our metamaterial mechanisms, i.e., bend their living hinges, only to the point where they still fully return to their original shape (the material stays "in the elastic regime"). As an example, operating the *NinjaFlex*-printed Jansen walker shown in Figure 26 about ~5000 times (at ~120 rpm) using a 300 W power drill, did not lead to any visible fatigue or damage in the metamaterial structure.

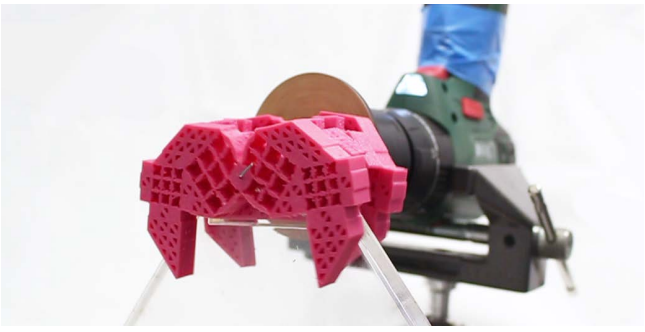

Figure 26: We verified the durability of the Jansen walker by operating it ~5000 times using a power drill.

CONCLUSIONS & FUTURE WORK

In this paper, we introduced metamaterial mechanisms. While metamaterials so far had been understood as materials, the main contribution of this paper is that we think of them as *machines*.

On the most basic level, it was the shear cell that allowed us to implement this new perspective on metamaterials. The shear cell allowed us to redirect forces and thus to create basic mechanisms, compound mechanisms, and ultimately simple machines.

While our approach offers tangible benefits for users (e.g., it solves mechanical problems in a single part, thereby eliminates the need for assembly), we see the main promise of this work in that it allows us to achieve a deeper integration between the structural and the mechanical functions of materials.

For future work, we plan to continue on this path by investigating how to integrate logical functions into material.

Acknowledgements

We thank David Lindlbauer for his insights and for printing many of our prototypes. We also thank Louis Kirsch, Moritz Hilscher, David Stangl, Arthur Silber, Friedrich Horschig and Noel Danz for their contribution to earlier versions of this work.

REFERENCES

1. Autodesk Within: *http://www.autodesk.com/products/within/overview*. Accessed: 2015-09-20.
2. Bächer, M., Whiting, E., Bickel, B. and Sorkine-Hornung, O. 2014. Spin-It: Optimizing Moment of Inertia for Spinnable Objects. *ACM Transactions on Graphics*. 33, 4 (2014).
3. Bickel, B., Bächer, M., Otaduy, M., Lee, H.R., Pfister, H., Gross, M. and Matusik, W. 2010. Design and fabrication of materials with desired deformation behavior. *ACM Transactions on Graphics*. 29, 4 (2010).
4. Chu, C., Graf, G. and Rosen, D.W. 2008. Design for additive manufacturing of cellular structures. *Computer-Aided Design*. 5, 5 (2008), 686–696.
5. Doubrovski, E.L., Tsai, E.Y., Dikovsky, D., Geraedts, J.M.P., Herr, H. and Oxman, N. 2014. Voxel-based fabrication through material property mapping: A design method for bitmap printing. *Computer-Aided Design*. 60, (2014), 3–13.
6. Elipe, J.C.Á. and Lantada, A.D. 2012. Comparative study of auxetic geometries by means of computer-aided design and engineering. *Smart Materials and Structures*. 21, 10 (2012), 105004.
7. Follmer, S. and Ishii, H. 2012. KidCAD: digitally remixing toys through tangible tools. In *Proceedings of CHI'12* (2012), 2401–2410.
8. Gannon, M., Grossman, T. and Fitzmaurice, G. 2015. Tactum : A Skin-Centric Approach to Digital Design

and Fabrication. In *Proceedings of CHI'15* (2015), 1779–1788.

9. Howell, L.L., Magleby, S.P. and Olsen, B.M. 2013. *Handbook of Compliant Mechanisms*. John Wiley and Sons.
10. Hudson, S.E. 2014. Printing teddy bears: a technique for 3D printing of soft interactive objects. In *Proceedings of CHI'14* (2014), 459–468.
11. Ishiguro, Y. and Poupyrev, I. 2014. 3D printed interactive speakers. In *Proceedings of CHI'14* (2014), 1733–1742.
12. Kou, X.Y. and Tan, S.T. 2007. Heterogeneous object modeling: A review. *Computer-Aided Design*. 39, 4 (2007), 284–301.
13. Liu, H., Maekawa, T., Patrikalakis, N.M., Sachs, E.M. and Cho, W. 2004. Methods for feature-based design of heterogeneous solids. *Computer-Aided Design*. 36, 12 (2004), 1141–1159.
14. Lu, L., Sharf, A., Zhao, H., Wei, Y., Fan, Q., Chen, X., Savoye, Y., Tu, C., Cohen-Or, D. and Chen, B. 2014. Build-to-Last : Strength to Weight 3D Printed Objects. *ACM Transactions on Graphics*. 33, 4 (2014).
15. MacCurdy, R., Katzschmann, R., Kim, Y. and Rus, D. 2016. Printable hydraulics: a method for fabricating robots by 3D co-printing solids and liquids. In *Proceedings of ICRA'16* (2016).
16. McCrae, J., Umetani, N. and Singh, K. 2014. FlatFitFab : Interactive Modeling with Planar Sections. In *Proceedings of UIST'14* (2014), 13–22.
17. Milton, G.W. and Cherkaev, A. V 1995. Which Elasticity Tensors are Realizable? *Journal of engineering materials and technology*. 117, 4 (1995), 483–493.
18. Mir, M., Ali, M.N., Sami, J. and Ansari, U. 2014. Review of Mechanics and Applications of Auxetic Structures. *Advances in Materials Science and Engineering*. (2014), 1–17.
19. Monolith: *http://www.monolith.zone/#introduction*. Accessed: 2015-09-20.
20. Montemayor, L.C., Meza, L.R. and Greer, J.R. 2014. Design and fabrication of hollow rigid nanolattices via two-photon lithography. *Advanced Engineering Materials*. 16, 2 (2014), 184–189.
21. Mueller, S., Lopes, P. and Baudisch, P. 2012. Interactive construction. In *Proceedings of UIST'12* (2012), 599–606.
22. Mullin, T., Deschanel, S., Bertoldi, K. and Boyce, M. 2007. Pattern transformation triggered by deformation. *Physical Review Letters*. 99, 8 (2007), 1–4.
23. netfabb: Selective Space Structures: *http://www.netfabb.com/structure.php*. Accessed: 2015-09-20.
24. Panetta, J., Zhou, Q., Malomo, L., Pietroni, N., Cignoni, P. and Zorin, D. 2015. Elastic Textures for Additive Fabrication. *ACM Transactions on Graphics*. 34, 4 (2015).
25. Paulose, J., Meeussen, A.S. and Vitelli, V. 2015. Selective buckling via states of self-stress in topological metamaterials. *arXiv preprint*. (2015), 12.
26. Prévost, R., Whiting, E., Lefebvre, S. and Sorkine-Hornung, O. 2013. Make It Stand: Balancing Shapes for 3D Fabrication. *ACM Transactions on Graphics*. 32, 4 (2013).
27. Saul, G., Lau, M., Mitani, J. and Igarashi, T. 2011. SketchChair: An All-in-one Chair Design System for End Users. In *Proceedings of TEI'11* (2011), 73–80.
28. Savage, V., Chang, C. and Hartmann, B. 2013. Sauron: embedded single-camera sensing of printed physical user interfaces. In *Proceedings of UIST'13* (2013), 447–456.
29. Savage, V., Schmidt, R., Grossman, T., Fitzmaurice, G. and Hartmann, B. 2014. A Series of Tubes : Adding Interactivity to 3D Prints Using Internal Pipes. In *Proceedings of UIST'14* (2014), 3–12.
30. Schumacher, C., Bickel, B., Rys, J., Marschner, S., Daraio, C. and Gross, M. 2015. Microstructures to control elasticity in 3D printing. *ACM Transactions on Graphics*. 34, 4 (2015).
31. Shim, J., Perdigou, C., Chen, E.R., Bertoldi, K. and Reis, P.M. 2012. Buckling-induced encapsulation of structured elastic shells under pressure. *Proceedings of the National Academy of Sciences*. 109, 16 (2012), 5978–5983.
32. Skouras, M., Thomaszewski, B., Coros, S., Bickel, B. and Gross, M. 2013. Computational design of actuated deformable characters. *ACM Transactions on Graphics*. 32, 4 (2013).
33. Tanenbaum, J.G., Williams, A.M., Desjardins, A. and Tanenbaum, K. 2013. Democratizing technology: pleasure, utility and expressiveness in DIY and maker practice. In *Proceedings of CHI'13* (2013), 2603–2612.
34. Vidimče, K., Wang, S.-P., Ragan-Kelley, J. and Matusik, W. 2013. OpenFab: A Programmable Pipeline for Multi-Material Fabrication. *ACM Transactions on Graphics*. 32, 4 (2013).
35. VoxCAD: *http://www.voxcad.com/*. Accessed: 2015-09-20.
36. Weichel, C., Lau, M., Kim, D., Villar, N. and Gellersen, H.W. 2014. MixFab: A Mixed-reality Environment for Personal Fabrication. In *Proceedings of CHI'14* (2014), 3855–3864.
37. Willis, K., Brockmeyer, E., Hudson, S. and Poupyrev, I. 2012. Printed optics: 3D printing of embedded optical elements for interactive devices. In *Proceedings of UIST'12* (2012), 589–598.