IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Rolling soft membrane-driven tensegrity robots

Robert L. Baines, Joran W. Booth, and Rebecca Kramer-Bottiglio

***Abstract*— We present a methodology for designing, fabricating, and controlling rolling membrane-driven tensegrity robots. This methodology is enabled by pneumatic membrane actuators and a generalized path planning algorithm for rolling polyhedra. Membrane actuators are planar, assembled in a scalable fashion, and amenable to arbitrary geometries. Their deformation trajectories can be tuned by varying the stacking sequence and orientation of layers of unidirectional lamina placed on their surfaces. We demonstrate the application of the same set of membrane actuators consisting of polygonal faces of Platonic Solids to create polyhedral tensegrity variants. Three specific tensegrities in the forms of cube, dodecahedron, and rhombicuboctahedron are chosen to demonstrate the path planning algorithm, though the algorithm is generalizable to any uniform or non-uniform n-sided polyhedra. The membrane-driven tensegrities are able to roll in unique trajectories and circumvent obstacles contingent on the distribution and types of polygons which constitute their faces.**

I. INTRODUCTION

The word *tensegrity* was coined by Buckminster Fuller, as a blend word to describe a type of structure composed of rigid parts under compression and compliant elements in tension [1]. The tensegrity paradigm is widespread, arising in biological structures, interpretations in art and architecture, as well as mechanisms with high strength-to-weight ratios [1]-[4]. More recently, tensegrities have gained traction in robotics as a means to navigate in unstructured or dynamically changing environments, places where traditional rigid robots often under-perform because external disturbances may damage or entirely destroy them. Tensegrities distribute external forces or disturbances internally through their network of tensile and compression elements, mitigating stress concentrations that might otherwise lead to mechanical failure in a traditional robot [5].

To impart motion onto a tensegrity, actuation schemes usually incorporate rigid motors that spool and contract the tensile elements [5]-[9]. Contracting tensile elements shifts the center of mass of a tensegrity, making it statically unstable, and as a consequence, it rolls to an adjacent face. One group inverted convention, making a tensegrity with linear actuators as struts to displace the cables [10]. A marked departure from controlling tensions in cables to destabilize the center of mass used vibration as a means for small tensegrities to move [11].

Considering previous work, it may be advantageous to rely on soft material actuators for tensegrity locomotion, rather than rigid motors. For example, replacing motors with soft actuators could enable new locomotion modes. A tensegrity

Department of Mechanical Engineering & Materials Science, Yale University, 10 Hillhouse Avenue, New Haven, CT 06520, USA. (email: `Rebecca.kramer@yale.edu`

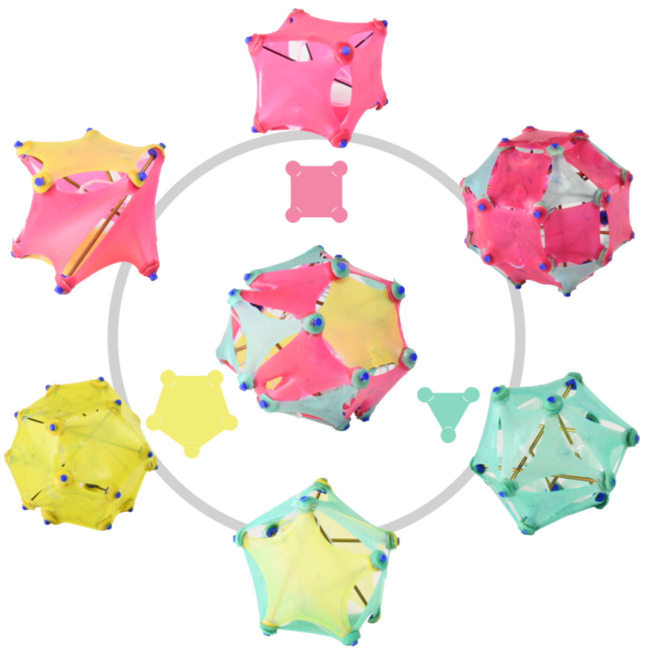

Fig. 1. With square, triangle, and pentagon -shaped membrane actuators as building blocks, we can construct a variety of polyhedra tensegrities. Based on their face arrangement and shapes, tensegrities may be created to more accurately roll on specified paths.

driven by soft contracting McKibben actuators introduced the first platform that rolled without motors [12]. More recently, our group showed a tensegrity driven by contraction of robotic skins consisting of McKibben actuators and soft sensors embedded into a single substrate [13]. In addition, groups have demonstrated tensegrities that actuated using shape memory alloy [14], [15] and shape memory polymer struts with elastomer tensile elements [16]. The aforementioned demonstrations relied on several actuators per face to impart deformation, making control highly complex. In addition, the designs relied on contractile motion between the nodes, overlooking the wealth of possible locomotion strategies that could be achieved by other motion primitives, such as extension or out-of-plane expansion.

This paper's first contribution is a design space for a new class of tensegrity, which we coin the membrane tensegrity. In contrast to a typical tensegrity where every member experiences exclusively axial forces concentrated at nodes, a membrane tensegrity's struts are held in compression by entirely soft membranes that sustain forces in a plane, rather than an axis, and simultaneously serve as actuators. Membrane actuators can impart motion onto tensegrities in new and unexplored ways, including out-of-plane expansion

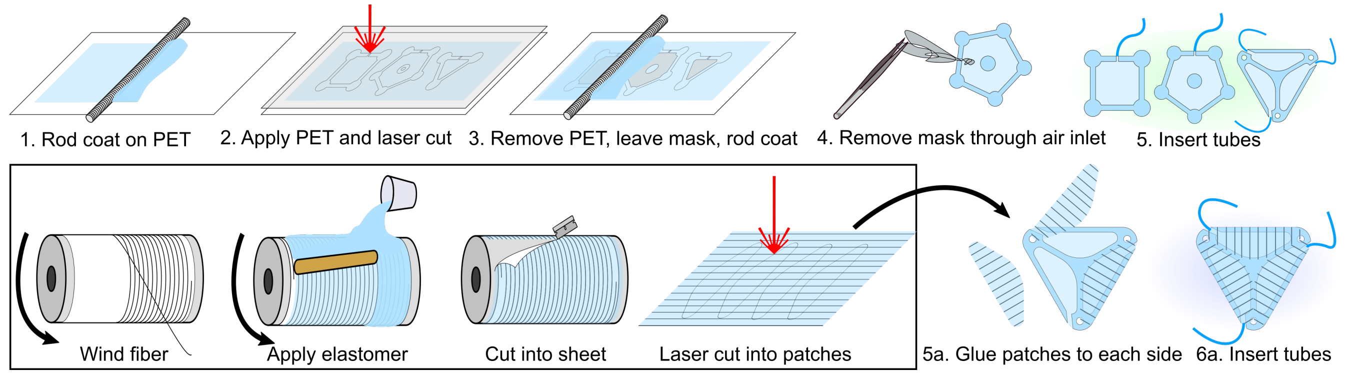
1. Rod coat on PET
2. Apply PET and laser cut
3. Remove PET, leave mask, rod coat
4. Remove mask through air inlet
5. Insert tubes
Wind fiber
Apply elastomer
Cut into sheet
Laser cut into patches
5a. Glue patches to each side
6a. Insert tubes

Fig. 2. Fabrication of membrane actuators is simple and scalable. Membrane actuators operate via volumetric expansion, which can be either constrained or unconstrained. After steps 1-4, step 5 shows unconstrained membrane actuators that expand out-of-plane. With the addition of unidirectional laminae patches (step 5a), the actuator expansion can be constrained and directed into areal extension or contraction of the membrane. In step 6a, the depicted triangular membrane is divided into three cavities, each with its own inlet. All three tubes are connected to the same air supply; the cavities inflate together.

to change center of mass without displacing the nodes, and areal extension and contraction that change center of mass by displacing the nodes. The same set of membranes and struts can be used to create myriad tensegrities in the space of uniform and non-uniform polyhedra (Figure 1). An outcome of using membrane actuators is that the arrangement of polygons which constitute a tensegrity's faces can be modulated to tailor performance in a given set of path planning problems. Incidentally, the problem of path planning for tensegrities is a relatively unexplored field. Thus, the second contribution of this paper is a generalized path planning algorithm for rolling polyhedra that goes hand-in-hand with the modular nature of membrane actuators.

II. Membrane Actuators

A. Fabrication of Membrane Actuators

Membrane actuators form the basis of the tensegrities constructed herein. Membrane actuators are simple to fabricate, easily formed to unique geometries, lightweight, and, due to their thinness, can be compacted into a small volume. Figure 2 illustrates the fabrication process for membrane actuators. Membrane actuators operate via volumetric expansion, which can be either constrained or unconstrained. Unconstrained membranes result in out-of-plane expansion (top), while constrained membranes (via the addition of unidirectional laminae) result in motions such as in-plane areal extension and contraction (top plus bottom). Both fabrication procedures follow the same first four steps.

First, a 2 mm layer of elastomer resin (Dragon Skin 10 Fast, SmoothOn Inc.) was rod-coated onto polyethelene terephthalate (PET) film with a precision threaded rod. Then, we flipped the rod-coated sheet over, PET-side up, and laser-cut away the PET to selectively mask the area of a desired internal bladder, or multiple segmented bladders (VSL 2.30, Universal Laser Systems). Next, another 2 mm thick layer of elastomer was applied via rod coating. To make out-of-plane expanding membranes, we laser cut out the desired shape contour, removed the internal PET sheet, and inserted silicone tubing to enable inflation. For directionally constrained membranes (Figure 2, bottom), we additionally cut out elastomer laminae with embedded unidirectional fibers [17], and adhered those to the both sides of the membrane to govern its inflation trajectory.

B. Motion Primitives by Varying Fiber Orientation

We are able to elicit a number of motion primitives from membrane actuators by adhering directionally-constraining laminae onto membrane surfaces (Figure 3). In addition to expansion, extension, contraction, twist, and bending motion primitives, it is possible to elicit highly complex deformations by superposition and localization of lamina stacking sequences.

We created oblong rectangular membrane actuator specimens (125 mm x 25 mm) to better visualize the motion primitives. The far left of Figure 3 presents a side view of a deflated membrane actuator for reference. To the right are side views of actuators inflated with 150 mL of air, numbered as specimens 1–5. Fiber orientation of the laminae influenced axial strain ϵ_a, transverse strain (*i.e.* bulge) ϵ_t, and if applicable, twisting γ_r and bending γ_b. Specimen 1, with no laminae, exhibits $\epsilon_t/\epsilon_a = 69$. For the extension actuator, specimen 2, $\epsilon_t/\epsilon_a = 11.32$. The ratio is small relative to that of specimen 1 and testifies to the fact that much of the deformation has been forced in the desired direction. Essentially a 2D McKibben muscle, specimen 3 maintains a relatively small $\epsilon_t/\epsilon_a = -22$ as well, yet undergoes contraction comparable to traditional McKibben muscles [18]. Of course, as with traditional McKibben muscles, the exact strains achievable are a function of the fiber braid angle. Owing to the simplicity of the fabrication process, the braid angle can easily be tuned. Moving on to the next motion primitives, specimen 4 twists one complete revolution while maintaining a smaller $\epsilon_a = -0.11$. However, the lack of axial deformation manifested itself transversely; ϵ_t was comparable to that of specimen 1, making $\epsilon_t/\epsilon_a = 102$. Lastly, specimen 5 bends at $\gamma_b = 350°$, nearly completing a full curl, with $\epsilon_t/\epsilon_a = -25$.

The exact strain-to-pressure or -volume relationship for a membrane actuator depends on geometry, pre-strains, and forces induced when they are initially placed between tenseg-

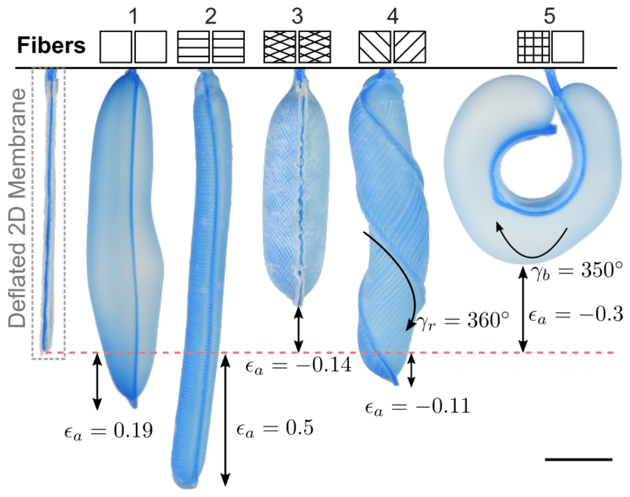

Fig. 3. Oblong 2D rectangular membrane actuators to better visualize motion primitives. The schametic above each specimen shows the angle, relative to vertical, at which fibers are embedded in any attached laminae. Left to right: at rest, inflated without any lamina, with fibers at 90° to force extension, $\pm 30°$ on both sides to create a McKibben-style contraction, 45° mirrored on each side for twist, and $90/0°$ on one side for bending. All were inflated with 150 mL of air. Scale bar: 30 mm.

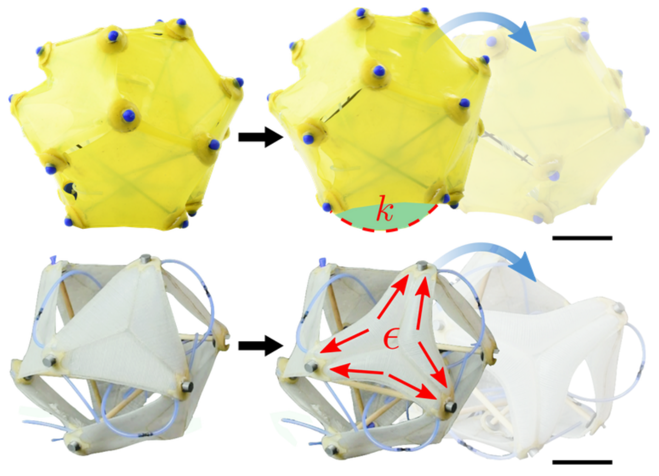

Fig. 4. Rolling mechanisms for tensegrities with out-of-plane expanding membrane actuators, which minimally shift the struts (top dodecahedron), and extension membrane actuators, which significantly shift the struts (bottom icosahedron). Scale bars: 25 mm (top), 40 mm (bottom).

rity nodes. Due to the extent of the membrane actuator design space, data presented in Figure 3 is intended to indicate general relationships between fiber placement and motion primitives, not definitive specifications for implementation. In this paper, we utilize square, pentagon, and triangle out-of-plane expanding membrane actuators for hardware implementation of path planning.

C. Assembly of Membrane-Driven Tensegrity Robots

The mechanism by which a tensegrity outfitted with out-of-plane expanding membrane actuators rolls is depicted in the top of Figure 4. Inflating a downward membrane creates a bubble of curvature k, an unstable equilibrium which, perturbed by inflation of neighboring faces, biases the center of mass to favor rolling to a particular face. Note the bubble is highlighted green. On the other hand, the bottom of Figure 4 shows how a tensegrity with constrained extension membrane actuators rolls. Inflation of a face displaces nodes with a strain ϵ, shifting center of gravity and toppling the robot. Both rolling mechanisms are shown in real-time in Supplementary Video Part 1.

In the present paper, we explore the utility of out-of-plane expanding membrane actuators on a rolling tensegrity because: 1) among the motion primitives in Figure 3, we believe expansion to dislodge tensegrity center of mass is most conducive to making tensegrities roll, 2) the expansion membrane lends itself to a simplistic actuation strategy compared to the constrained actuators, and 3) tensegrities utilizing out-of-plane expansion to move have not yet been reported. Though we proceed with this limited scope, we believe the motion primitives present in Fig. 3, and combinations thereof, should provide engineers an ample tool set to devise novel locomoting membrane tensegrities.

Polyhedra tensegrities were constructed using the same set of membrane actuators (out-of-plane expanding square, pentagon, and triangle with side lengths 50 mm) and 1/16 in diameter carbon fiber composite struts (McMaster Carr) of 125 mm length. In our specific hardware implementation, the cube, dodecahedron, and rhombicuboctahedron tensegrities consisted of four, ten, and twelve struts, respectively. At each end of each strut, 3D-printed nodes provided an interface site for the membranes. To preserve modularity and interchangeability, we did not bond the membranes to the nodes. Rather, compression supplied by the membrane network held the structure intact.

III. Path Planning For Membrane Tensegrities

A sizable body of literature employs machine learning techniques for control of tensegrity robots. These works focus on how to generate actuator commands required to travel the furthest by some heuristic within a given time frame (usually euclidean distance between two points over 60 s) [19]-[21]. Other research developed actuator policies for rolling icosahedron and rhombicuboctahedron tensegrities, but in consideration of the latter shape, did not allow for rolling to and from triangular faces that are a source of great complexity in movement [22].

In contrast to the large amount of literature developing actuator control policies for tensegrities, scant work pertains to path planning in the traditional sense of solving for a route from point A to B in the presence of obstacles. Prior research most related to the tensegrity path planning domain presented algorithms for a specific geometric configuration: the popular six-strut icosahedron tensegrity. One group introduced a steerable locomotion controller for an icosahedron tensegrity which mapped user joystick inputs to a rolling sequence [23]. Another group presented a sampling-based kinodynamic planning approach for an icosahedron tensegrity approximated as a spherical entity [24].

The use of membrane actuators—modular units adaptable to arbitrary shapes and easily assembled on-the-fly to create unique polyhedra tensegrities—demands a path planning

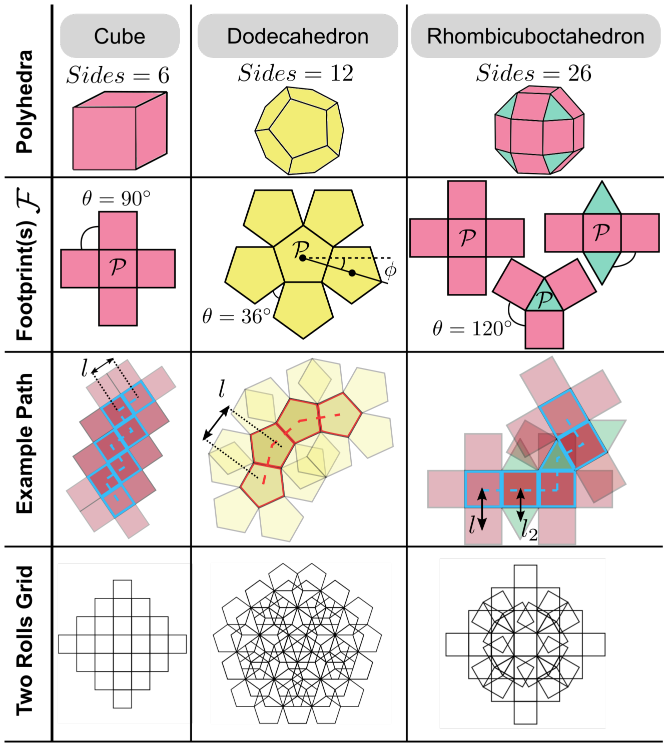

Fig. 5. Each polyhedra has a unique set of footprints that determine the possible set of paths it can achieve via rolling. Imperfectly tessellating polyhedra pose the challenge of a large, complex grid for planning motions.

algorithm agnostic to shape and configuration to truly realize their potential. Pioneering work furnished mathematical context for generalized polyhedra rolling on a plane yet conceded they were not able to test their algorithm on a system, simulated or physical [25]. We build on prior work and create a generalized path planning algorithm for rolling n-sided polyhedra, uniform or non-uniform. The algorithm combines the A* graph search with geometric constraints of a given polyhedra. Instead of relying on hard-coded coordinate frame assignments and a series of affine transformations, variables that change based on shape, our algorithm maintains generality by evoking an optimization routine to keep track of the orientation of a polyhedra as it rolls.

A. *Geometric Reasoning for Planning Algorithm*

As a polyhedra tensegrity rolls, its downward face polygon $\mathcal{P}$ will occupy a unique orientation on the ground. The orientation-invariant possible set of moves from any downward face contacting the ground to its neighboring faces which share an edge can be considered as its geometric footprints $\mathcal{F}$. Figure 5 gives examples of different polyhedra downward faces and their neighbors flattened into footprints. Notice that the cube has one such footprint, and it forms a perfectly tessellating square grid. In contrast, the dodecahedron and rhombicuboctahedron do not have perfectly tessellating grids. Furthermore, $\mathcal{F}$ for the rhombicuboctahedron contains three items.

The subset of $\mathcal{F}$ that is a grid of perfectly tessellating polygons, like the triangle, square and hexagon, is easy to visualize and a simple space in which to plan movement from point A to point B. Consider the cube. Evoking the famous A* algorithm, an evaluation function is used to find the optimal path across the grid in the presence of obstacles:

$$f(n) = g(n) + h(n) \tag{1}$$

Here, $g(n)$ is the cost from the initial node to the present node n, and $h(n)$ estimates the cost of an optimal path from node n to the goal node; a value that is defined by the heuristic (commonly euclidean distance). At each step, the algorithm picks the next n to move to with the lowest f [26].

We cannot constrain our tensegrity design space to uniform grids of perfectly tessellating shapes because membrane actuators can achieve any shape and tensegrities can be composed of multiple differently shaped polygonal membrane actuators. Imperfectly tessellating polyhedra footprints and those which are non-uniform (constituted of disparate polygonal faces) may exhibit non-holonomy. That is, rolling from point A to point B, each step of the tensegrity depends on the previous step(s) taken. The sheer magnitude of configurations of $\mathcal{F}$ on the plane prohibits explicitly developing a grid of possible moves in $\mathcal{F}$ *a priori* for larger search spaces. For visualization, the bottom row of Figure 5 shows the stacked contours $\mathcal{F}$ for two moves. It is clear that planning optimal movements in large grids—where rolls may be on the order of hundreds—becomes computationally expensive. Thus we take an alternative approach. We sacrifice guarantees of admissibility for practicality and fast convergence on a viable path.

Algorithm 1 Rolling n-sided Tensegrity Following A*-Generated Path

```
Input: $A^*, \mathcal{O}, \mathcal{F}$            ▷ $A^*$ Path, Obstacles, Footprints
Output: $storedFaces$                ▷ Vector of Faces Traversed
 1: $cpoints = createCheckPoints(A^*)$
 2: $d = \|[x,y]_{Face},\ [x,y]_{cpoint}\|_{L2}$  ▷ Initialize Distance to
    Checkpoint
 3: for $length(cpoints)$ do
 4:     while $d > tol$ do  ▷ Roll Until Within $tol$ Tolerance
 5:         $[l,\phi] = findCentroids(\mathcal{F},\mathcal{A})$
 6:         $[\xi, faces] = generalOrient([x,y]_{prevFace})$
 7:         $ctroid = [x,y]_{Face} * l * cos(\phi+\xi)$
 8:         for $n \in \mathcal{P}$ do      ▷ $n$-sided Polygon on Ground
 9:             if $ctroid(n) \in \mathcal{O}$ then
10:                 $ctroid(n) = \infty$      ▷ Penalize Collisions
11:         $[[x,y]_{nextFace},\ i] = \min \|ctroid,\ [x,y]_{cpoint}\|_{L2}$
12:         $faceNum_{selected} = faces(i)$
13:         $rollToNextFace([x,y]_{nextFace})$
14:         $[x,y]_{prevFace} = [x,y]_{Face}$
15:         $[x,y]_{Face} = [x,y]_{nextFace}$
16:         $d = \|[x,y]_{Face},\ [x,y]_{cpoint}\|_{L2}$          ▷ Update
17:         $storedFaces.append(faceNum_{selected})$
```

B. *General Path Planning for Polyhedra Tensegrity*

In our path planning pipeline, we first impose a dense square grid on the environment and perform a traditional

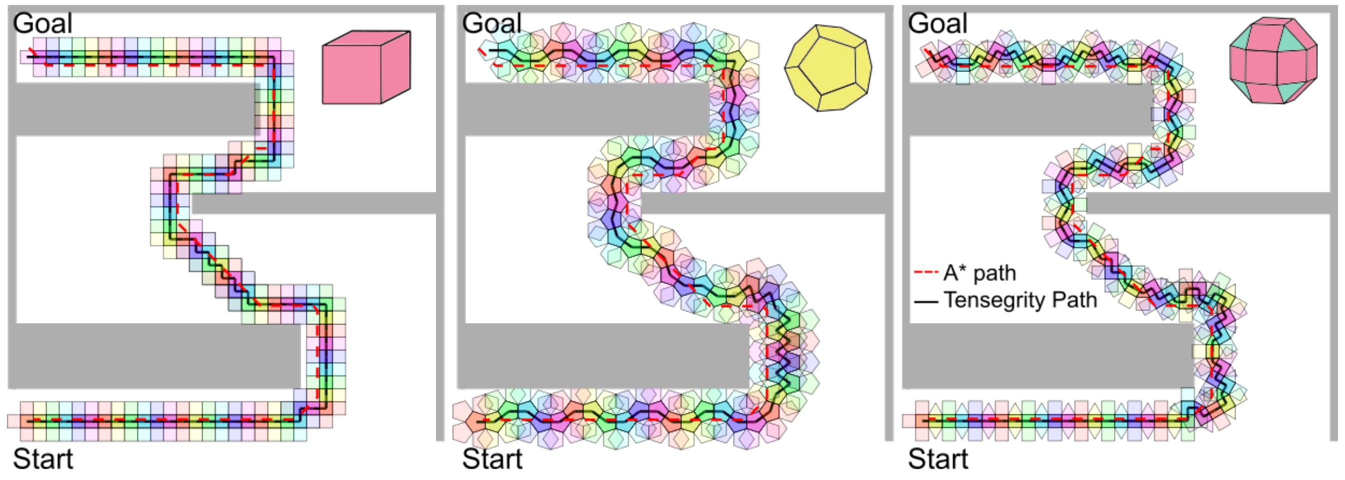

Fig. 6. Solving a simulated maze: rolling polyhedra can traverse a generated A* path broken into checkpoints, irrespective of shape. The geometry of the shape determines the ability to move in straight lines, take sharp or smooth corners, and also influences the number of rolls needed to reach the goal.

A* search defined by Equation 1 to compute a path oblivious to the geometric rolling constraints of the tensegrity. Next, Algorithm 1 begins. The path generated by A* is further discretized into a series of checkpoints via a $createCheckPoints$ method. The number of checkpoints is arbitrarily set to lie between the A* path length divided by the shortest edge of the polyhedra and the A* path length divided by the longest edge. This technique ensures that there is a sufficient number of checkpoints to prevent straying from the A* path or, conversely, excessive meandering. Next, for a given polyhedra comes a set of adjacency relationships $\mathcal{A}$. We can use $\mathcal{A}$ to find centroid-to-centroid distances l_j for each of the footprints, where $j = 1, ..., k$ and k is the number of unique centroid-to-centroid distances for a polyhedra. We can also use $\mathcal{A}$ to find the angles ϕ of neighboring centroids relative to $\mathcal{P}$. Both tasks are accomplished via the $findCentroids$ method. Note that $\mathcal{A}$ depends on a user-defined numbering scheme for faces. For example, on the cube, we could have labeled each of the sides 1–6. If $\mathcal{P} = 1$ then $\mathcal{A}$ contains 2, 3, 4, and 5—faces sharing an edge with 1. l is a centroid-to-centroid distance between neighboring faces and is constant for uniform polyhedra like the cube but can change for non-uniform polyhedra like the rhombicuboctahedron.

After delineating neighboring faces' centroid coordinates relative to $\mathcal{P}$, the $generalOrient$ method is invoked. We keep track of the orientation of the polyhedra as well as user-numbered faces across which a tensegrity rolls while retaining generality for any n-sided polyhedra, uniform or non-uniform. To do so, in $generalOrient$ we employ a 1D optimization that finds the angle ξ which minimizes the distance between $[x, y]_{prevFace}$ and $[x, y]_{Face}$:

$$\min_{\xi} \|[x,y]_{prevFace} - ([x,y]_{Face} + [lcos(\xi+\phi), lsin(\xi+\phi)])\| \tag{2}$$

ξ is propagated throughout the rolling sequence as an additive term to compensate for generic changes in orientation to $\mathcal{F}$. The $generalOrient$ method also outputs the face numbers $faces$ within the footprint that can be used later to convert a computed path to a sequence of actuator commands on the physical system. After the optimization routine, the footprint of possible single moves is searched for the face which lies closest to the current checkpoint in line 11. If there are no obstacles $\mathcal{O}$ contained within the same face as a potential roll, the tensegrity rolls across the corresponding edge. The chosen face number is stored in $storedFaces$. Rolling reveals a new footprint of potential rolls. The process of expanding the footprint and evaluating for the closest face that does not contain obstacles loops until the tensegrity rolls within a threshold tol of the checkpoint. When the tensegrity is within tol, the algorithm increments to the next checkpoint on the A* path. The geometric constraints imposed during rolling are that the tensegrity can only roll over edges, not corners, and it must chose a next move bounded by $\mathcal{F}$.

Using this planning pipeline, we exploit a fast initial search for an optimal path on a grid that is perfectly tessellating, followed by subsequent ordered searches constrained to the number of elements in $\mathcal{F}$ that strive to best follow the optimal path. Convergence on a viable path can occur faster than it does compared to performing an A* with polyhedra geometric constraints defined at each step, especially for polyhedra with $\mathcal{F}$ which do not tessellate the plane.

C. Implementing Membrane Tensegrity Path Planning in a Simulated Maze

To validate the generalized algorithm, we used it in path planning of simulated cube, dodecahedron, and rhombicuboctahedron -shaped membrane tensegrities. In MATLAB, we simulated the same maze for each type of membrane tensegrity which had equivalent edge lengths, matching those of actual hardware used later (50 mm). A* was broken into 70 checkpoints based on the side length of the tensegrity and the size of the grid. Figure 6 presents simulation results. Gray sections denote walls and the A* generated path is the dashed red line. Polyhedra footprints are shown by alternating colors and the black line connects downward

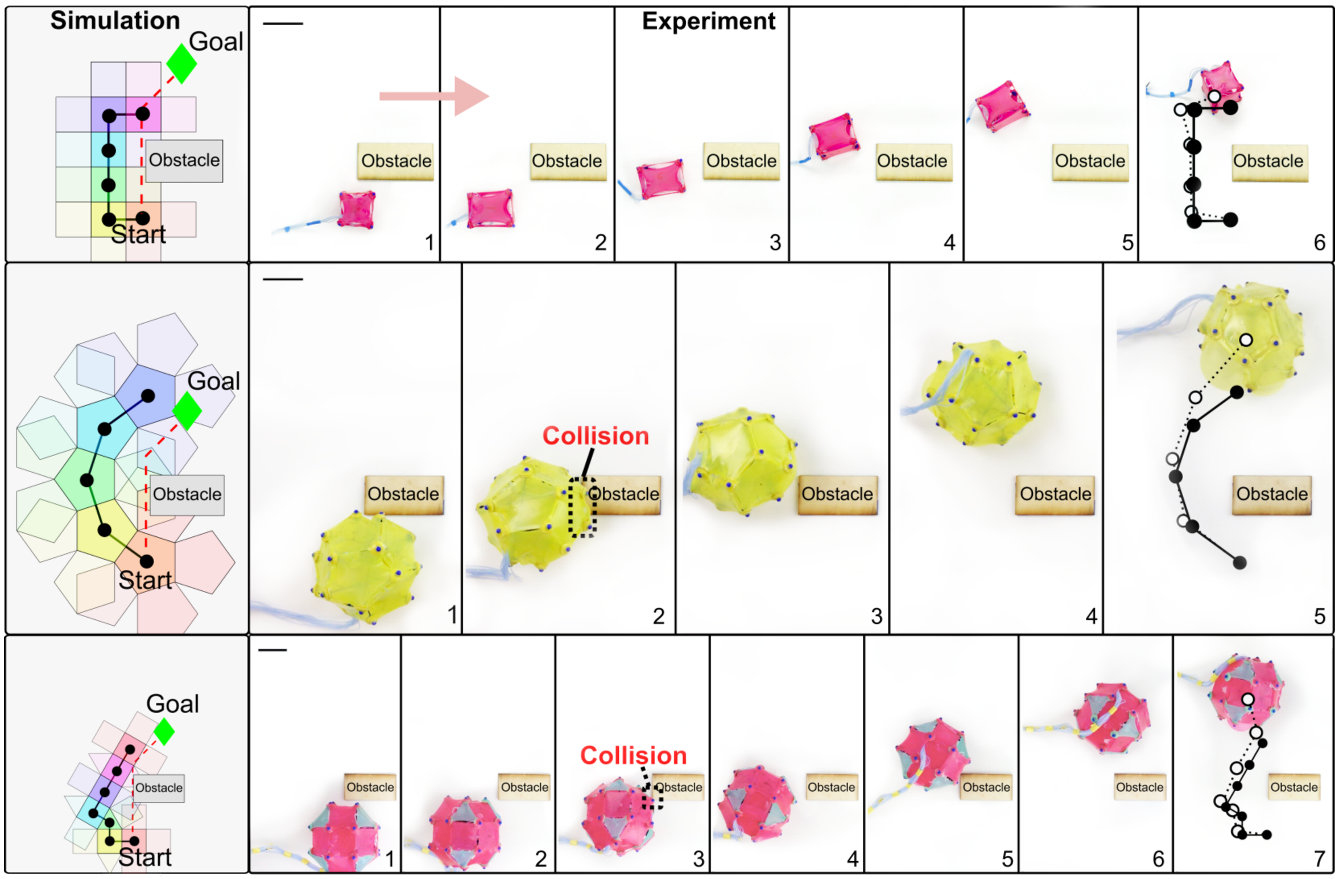

Fig. 7. Path planning algorithm implemented in simulation and translated to actual tensegrities. The same simple problem is shown for each polyhedral variant to home in on idiosyncrasies of operation. The dashed line with white dots at the end of the frames indicates the experimental trajectory of the centroids while the solid line with black dots, the theoretical trajectory. Scale bar in top left of each experiment sequence: 50 mm.

faces step-wise over time. Starting at the bottom left, each case was able to reach the goal at the top left.

The percentage error between the distance of the A*-generated path on the imposed fine square grid and the distance traveled by our simulated tensegrity indicates extent of deviation from an optimal path. For the cube, dodecahedron, and rhombacuboctehedron, the percentage error between the optimal path and the actual route taken via Algorithm 1 was 11.5%, 21.6%, and 28.9%, respectively. The increasing error with complexity of shape is evidenced visually by the straighter routes the cube takes relative to the meandering routes of the other shapes.

Another finding from Figure 6 is that the centroid-to-centroid distance l between each downward face in a footprint governs the number of total rolls required to reach the goal. For instance, the dodecahedron only took 74 rolls to reach the goal compared to the cube's 91 rolls. Interestingly, the rhombicuboctahedron took 116 rolls to reach the goal. Consisting of both triangles and squares, it has greater mobility than the cube. While greater mobility is advantageous for more closely following checkpoints, it can encourage meandering in between those checkpoints, causing polyhedra to take indirect paths toward the goal.

In a similar vein, we note that as a polyhedra tensegrity decreases in θ it approximates a sphere more, and is able to make both sharper turns and smoother diagonal movements. This fact is underscored when comparing the dodecahedron and cube's diagonal movement portions in the maze. The cube makes two movements to approximate a diagonal, owing to the $\theta = 90°$ edge spacing of its footprint. In contrast, the dodecahedron has much more movement flexibility due to its smaller $\theta = 36°$ (Figure 5), though it cannot maintain a purely straight path like the cube. In practical applications, there exists a balancing act between size and shape of a membrane tensegrity to minimize energy of traversal across a path. Larger or fewer faces allow a tensegrity to reach a point with fewer rolls, and therefore fewer actuation cycles. Yet on physical systems larger size implies greater mass; and fewer faces means more shift must occur to destabilize center of mass. On the other hand, smaller or more faces may require more rolls but have less mass to move and may more accurately approximate a path. Here the advantage of membrane actuators comes to the fore: ease adapting them to custom geometries and sizes should help the engineer faced with the unique challenges of tensegrity navigation.

D. *Translating Simulated Path Planning to Reality*

To further evaluate the generalized algorithm pipeline, we simulated a simple path planning problem on the three

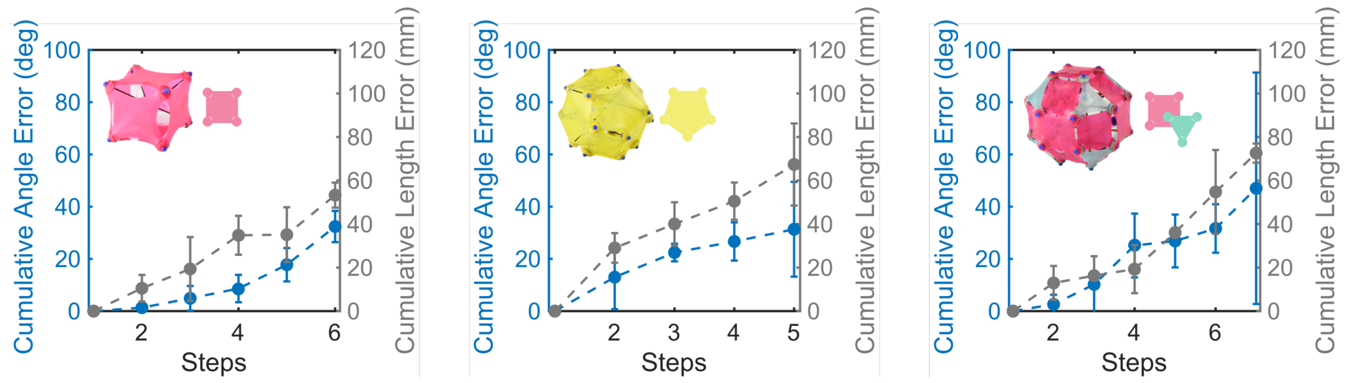

Fig. 8. Cumulative absolute error between simulated and experimental values of the three tensegrities as they roll. Plotted points are a mean of three experimental trials. Confidence bars indicate one standard deviation above and below the mean.

tensegrity variants presented in Figure 5 and translated the solution to hardware (Supplementary Video Part 2). We conducted three trials for each tensegrity to understand the variance associated with implementing a fixed generated path on actual hardware. To track differences between simulation and hardware, we recorded the tensegrity executing the path with an HD camera facing downward. The obstacle was chosen to be a flat wooden block that would indicate whether a collision occurred if it were occluded from above.

Having stored the faces that the tensegrity traversed as the output of our path planning algorithm, we translated the calculated path to a series of actuator commands. Of course, the exact nature of the path-to-actuator command mapping is dependent on the type of actuator. For out-of-plane expanding membrane actuators, we simply inflated the membrane on the downward face of the tensegrity, leveraging the rolling mechanism depicted in the top of Figure 4, one similar to the rolling mechanism proposed for other membrane-driven robot systems [27]. The system becomes unstable and the direction it will roll is unpredictable. However, we can direct the roll by also inflating all the adjacent faces except the one in the direction of the desired roll. We utilized an Arduino Uno in communication with miniature pressure regulators [28] to command rolling sequences.

Figure 7 juxtaposes experimental and simulated path planning results for one of the three conducted trials. Each tensegrity experienced propagation of error in rolling, evidenced by the difference between the simulated trajectory (black line with solid dots pinpointing centroid location) and experimental trajectory (dotted line with white dots).

Figure 8 plots the cumulative absolute angle error as well as the cumulative distance error step-wise throughout the experiment for each tensegrity shape. Plotted points represent an average of the three trials. Confidence bars give one standard deviation from the mean. Step-wise error accrual is due to a number of factors. First, the pneumatic tether and wires emanating from a tensegrity bias direction. To achieve longer and unimpeded rolling sequences, tetherless operation is necessary. Second, our algorithm assumes perfect rolling. As can be seen from Supplementary Video Part 2, sliding and bounce stray the tensegrities from their theoretical perfect rolling trajectories. By switching the membrane type to extension or contraction, it is likely we could reduce bouncing and sliding because in those types of actuation, the tensegrity "relaxes" to its next face rather than "falling" as with inflation. Additionally, the 3D-printed nodes on the tensegrities were made from thermoplastic. Choosing a material for the nodes with a higher friction coefficient could alleviate sliding issues. Third, face edge lengths of the tensegrities were not entirely uniform and therefore did not match those of the perfect polyhedra in simulation. Disparate tensions caused by the arrangement of internal struts skewed edge lengths. The results of Figures 7 and 8 and observation of variance in hardware reinforces that a noise-free estimate of the path-length heuristic cannot be guaranteed admissible.

While conducting the experiment, we found that slow, quasi-static inflation of the actuators lead to more stable and repeatable rolls. Fast inflation imparted significant angular momentum to the tensegrities and caused them to roll multiple times, deviating from steps in the generated path. Although multiple rolls for a single inflation may not be desirable when carefully following a generated path, the behavior could be leveraged in the interest of dynamic or unusual modes of locomotion.

As a last note on practical implementation, though it exhibited the least average angle and length error step-to-step on its trajectory due to its sharp contours, (cumulative angle and length errors of 32.4° and 53.3 mm, respectively) tipping the cube took significantly more inflation than the other polyhedra (which approximate spheres to a greater degree). The initial high-pressure hurdle to tip sometimes prevented the downward membrane's inflation from tipping the cube over the appropriate edge. The opposite problem occurred with the rhombicuboctahedron: it rolled too easily and freely. In step 7 of the trial visualized in Figure 7, the rhombicuboctahedron deviated from the final step of its path substantially, even though the correct actuators inflated. Consequently, the angle error for the rhombicuboctahedron in step 7 of Figure 8 has a large standard deviation. Since a rhombicuboctahedron is more spherical than a cube, rolling to a certain point requires less pressure but is associated with higher variability. Especially with neighboring actuators inflated, the rhombicuboctahedron assumes a very rounded surface that tips with little bias.

Due to the confounds mentioned above, particularly nonequivalent edge lengths induced by disparate tensions, collisions occurred. In the trial pictured in Figure 7, the dodecahedron collided with the right portion of the obstacle in step 2 of its trajectory. The rhombicuboctahedron also collided with the obstacle in step 3. In spite of the collisions and deviations from simulation, the results of Figure 7 serve as a testament to the compatibility of membrane actuators and the generalized polyhedra path planning algorithm. Knowledge of how membrane tensegrities replicate paths in real life can help inform the design of robots for specific path planning problems that require, say, the location certainty of the cube, the maneuverability of the dodecahedron, or the spontaneity of the rhombicuboctahedron.

IV. CONCLUSIONS

We presented a new class of tensegrity robot composed of membrane actuators. Membrane-driven tensegrity robots are easy to fabricate, able to be tailored to arbitrary geometries, and deform in numerous ways. The membrane actuators work in tandem with a generalized path planning algorithm, providing a tool set for rapidly designing and implementing tensegrities for rolling. In future work, we intend to utilize an IMU to account for the disparity between theoretical and actual orientation of a downward face to correct the path planner in real-time. We will also explore path planning with different membrane actuator types introduced herein, such as extension, contraction, and combinations of these types on the same system. Lastly, we should remark that the A* to Algorithm 1 pipeline is by no stretch optimal. The objective of the pipeline was to circumvent the scaling problem associated with large search spaces of potential moves for non-tessellating polyhedra footprints. For smaller-scale problems, we found that A* with step-wise geometric constraints can be used to generate guaranteed admissible paths. Since the intention of the present work was to introduce the concept of complementary hardware and generalized path planner for customized polyhedra membrane-driven tensegrities, we leave heuristic tuning to future work.

ACKNOWLEDGMENT

We thank A. Garcia and S. Sinmaz for making actuators and D. Shah for compelling math discussions. R.L.B. was supported by the National Science Foundation Graduate Research Fellowship (DGE-1333468). J.W.B. was supported by NASA through the Early Career Faculty Program (80NSSC17K0553).

REFERENCES

[1] R. E. Skelton and M. C. De Oliveira, *Tensegrity Systems*, 2009 ed. Springer, Jun. 2009.

[2] S. Levin, The tensegrity-truss as a model for spine mechanics: Biotensegrity. *J Mech Med Biol*, vol. 2, pp. 375-388. 2002.

[3] T. D'Estree Sterk. Shape control in responsive architectural structures - current reasons and challenges. 4th World Conference on Structural Control and Monitoring, 2006.

[4] K. Miura and Y. Miyazak. Concept of the tesnsion truss antenna. *AIAA Journal*, vol. 28, no. 6, pp. 1098–1104, 1990.

[5] C. Paul et al., Design and control of tensegrity robots for locomotion, *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 944-957, 2006.

[6] K. Caluwaerts et al., Design and control of compliant tensegrity robots through simulation and hardware validation, *Journal of The Royal Society Interface*, vol. 11, no. 98, 2014

[7] V. SunSpiral et al., Tensegrity based probes for planetary exploration: Entry, descent and landing and surface mobility analysis. 10th International Planetary Probe Workshop, 2013.

[8] A. Agogino et al., Super Ball Bot - structures for planetary landing and exploration. NASA Innovative Advanced Concepts (NIAC) Program, Final Report. 2013.

[9] M. Vespignani et al., Design of SUPERball V2, a Compliant Tensegrity Robot for Absorbing Large Impacts, IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 2865-2871, 2018.

[10] L.-H. Chen et al., Soft Spherical Tensegrity Robot Design Using Rod-Centered Actuation and Control, *Journal of Mechanisms and Robotics*, vol. 9, no. 2, pp. 025001, 2017.

[11] J. Rieffel and J.-B. Mouret, Adaptive and Resilient Soft Tensegrity Robots, *Soft Robotics*, vol. 5, no. 3, pp. 318-329, 2018.

[12] Y. Koizumi et al., Rolling tensegrity driven by pneumatic soft actuators, in IEEE International Conference on Robotics and Automation, pp. 1988-1993, 2012.

[13] J. W. Booth et al., OmniSkins: Robotic skins that turn inanimate objects into multifunctional robots, *Sci. Robot.*, vol. 3, no. 22, 2018.

[14] F. A. dos Santos et al., Design and experimental testing of an adaptive shape-morphing tensegrity structure, with frequency self-tuning capabilities, using shape-memory alloys, *Smart Mater. Struct.*, vol. 24, no. 10, p. 105008, 2015.

[15] M. Shibata et al., Crawling by body deformation of tensegrity structure robots, IEEE International Conference on Robotics and Automation, pp. 4375-4380, 2009.

[16] K. Liu et al., Programmable Deployment of Tensegrity Structures by Stimulus-Responsive Polymers, *Sci. Rep.*, vol. 7, no. 1, pp. 3511, 2017.

[17] S. Y. Kim and R. Baines et al. Reconfigureable soft body trajectories using unidirectionally stretchable laminae *Nat Comm.* vol. 10, pp. 3464, 2019.

[18] C. S. Kothera et al., Experimental Characterization and Static Modeling of McKibben Actuators, *Journal of Mechanical Design*, vol. 131, no. 9, pp. 091010, 2009.

[19] A. Iscen et al. Controlling tensegrity robots through evolution, in Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference - GECCO 13, 2013, p. 1293.

[20] M. Zhang et al., Deep reinforcement learning for tensegrity robot locomotion, IEEE International Conference on Robotics and Automation, 2017, pp. 634-641.

[21] J. Luo et al. Tensegrity Robot Locomotion Under Limited Sensory Inputs via Deep Reinforcement Learning, IEEE International Conference on Robotics and Automation, 2018, pp. 6260-6267.

[22] K. Kim et al., Rolling Locomotion of Cable-Driven Soft Spherical Tensegrity Robots, *Soft Robotics*, 2020.

[23] M. Vespignani et al., Steerable Locomotion Controller for Six-strut Icosahedral Tensegrity Robots, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2886-2892, 2018.

[24] Z. Littlefield et al., From QuasiStatic To Kinodynamic Planning For Spherical Tensegrity Locomotion, International Symposium on Robotics Research (ISRR), pp. 16, 2017.

[25] S. Piccinocchi et al., Planning Motions of Polyhedral Parts by Rolling, *Algorithmica*, vol. 26, no. 34, pp. 560-576, 2000.

[26] P. Hart et al., A Formal Basis for the Heuristic Determination of Minimum Cost Paths, IEEE Trans. Syst. Sci. Cyber., vol. 4, no. 2, pp. 100107, 1968.

[27] E. Steltz et al., JSEL: Jamming Skin Enabled Locomotion, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 5672-5677.

[28] J. W. Booth et al., An addressable pneumatic regulator for distributed control of soft robots, IEEE International Conference on Soft Robotics pp. 25-30, 2018.