



Morphing-Enabled Path Planning for Flying Tensegrity Robots as a Semidefinite Program

Sergei Savin * and Alexandre Klimchik

Robotics and Computer Vision Institute, Innopolis University, Innopolis, Russia

The development of deformable drones is of high importance but presents significant challenges. Such drones can be based on tensegrity structures, which leaves open the questions of configuration-space path planning for such robots. In this paper we propose a method that takes advantage of a simplified encoding of the drone's shape, allowing to turn the path planning into a sequence of semidefinite programs. The mapping from the simplified description and the actual tensegrity configuration is done *via* a data-driven method, using a pre-computed dataset of statically stable configurations and their outer Löwner-John ellipsoids, as well as eigendecompositions of the ellipsoid matrices. Together it allows rapid containment check, whose computational cost depends linearly on the number of dataset entries. Thus, the proposed method offloads computationally-intensive parts to the offline dataset generation procedure, speeding up the algorithm execution.

OPEN ACCESS

Edited by:

Adnan Kefal,
Sabanci University, Turkey

Reviewed by:

Mark Post,
University of York, United Kingdom
Changsheng Dai,
University of Toronto, Canada

***Correspondence:**

Sergei Savin
s.savin@innopolis.ru

Specialty section:

This article was submitted to
Soft Robotics,
a section of the journal
Frontiers in Robotics and AI

Received: 10 November 2021

Accepted: 21 January 2022

Published: 14 March 2022

Citation:

Savin S and Klimchik A (2022) Morphing-Enabled Path Planning for Flying Tensegrity Robots as a Semidefinite Program. *Front. Robot. AI* 9:812849.
doi: 10.3389/frobt.2022.812849

1 INTRODUCTION

The last 2 decades have seen steady progress in the theory of tensegrity structures and their applications in Robotics. From tensegrity planetary landers [see Sabelhaus et al. (2015a); Vespiagnani et al. (2018)] to tensegrity spines for quadruped robots [Mirletz et al. (2014); Sabelhaus et al. (2015b); Zappetti et al. (2020)], the properties of these structures have provided new ways to design mobile objects performing difficult tasks. These properties include low weight, resistance to damage from collisions, the ability to control the stiffness of the structures, and the ability to control the deformation of the robot [see Liu et al. (2022) for a review of other properties of the tensegrity structures]. The latter properties motivate the interest in tensegrity as a structural element of soft robots [Lee et al. (2020); Rieffel and Mouret (2018); Kimber et al. (2019)].

One of the possible areas of application for tensegrity structures is deformable drones. Such robots can be used to traverse the cluttered environments, temporarily adjusting their shape to fit through narrow spaces, as well as take advantage of their higher maneuverability [Deng et al. (2020)]. Collision resilience should allow such drones to navigate partially uncertain environments, where exact positions and shapes of the obstacles are not always available. However, in order to effectively use such drones, two connected classes of problems applied to deformable tensegrity robots need to be solved: path planning and control. This paper is focused on the problems of the first class.

Same as in other areas of under-actuated robotics, path planning for deformable tensegrity structures requires finding a feasible path in the state space of the robot (assuming the environment is static, and the robot dynamics can be described by autonomous ODEs). And similar as in other high-dimensional systems, such as walking robots, a direct search for such paths is difficult; this makes

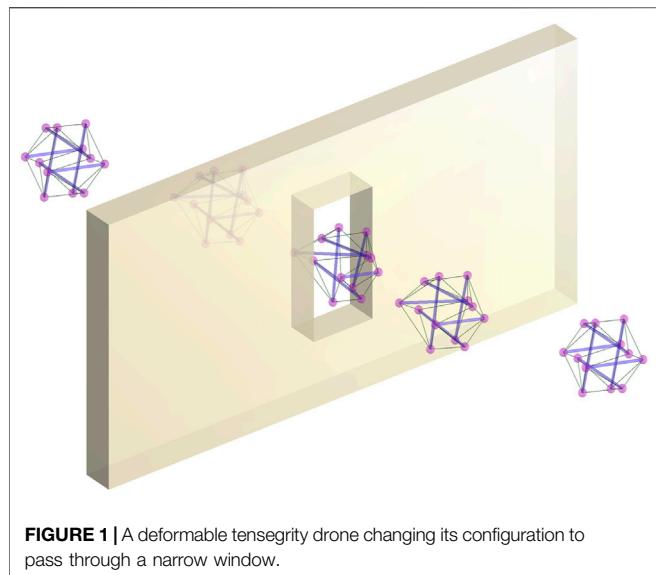


FIGURE 1 | A deformable tensegrity drone changing its configuration to pass through a narrow window.

simplified methods, based on conservative assumptions a favorable alternative. One of the most basic simplifications is the assumption that the trajectory of the robot is *quasi-static*, meaning that each configuration the robot assumes while traversing the path is statically stable (or there exists a control law that can make it become so). For a tensegrity structure, this means that each configuration it assumes should correspond to a local minimum of the potential energy of the structure. This allows us to transpose the path planning problem into a problem of finding a sequence of configurations that would path through the environment.

In this paper, we propose a method for finding sequences of configurations of deformable tensegrity drones as a sequence of deformations of the original shape, with the space of deformations being equivalent to the space of positive-definite matrices. This limits the possible shapes the structure can assume, but at the same time, it allows to cast the problem as a semi-definite program (SDP), taking advantage of the widely used solvers with well-understood properties. Moreover, the constraints in the proposed method have the form of linear-matrix inequalities (LMI), which is a familiar constraint type in a number of control-related applications. **Figure 1** illustrates the work of the algorithm.

2 STATE OF THE ART

Motion planning for deformable objects is a wide field of study, with notable sub-fields. For example, motion planning for rod-like structures (generalized under the category of deformable linear objects) has been well studied [see Moll and Kavraki (2006); Roussel et al. (2014); Shah and Shah (2016)]; there, such structures are usually manipulated by external forces. Similarly, methods for manipulation-based motion planning for objects of other shapes have been proposed Anshelevich et al. (2000). Here we concentrate on the case when the robot

uses internal resources for deformation. This requires a way to predict the form of the structure, given the distribution of forces and torques acting on it.

As was discussed previously, limiting ourselves to the quasi-static case (when the speeds and accelerations of the motion are such that the inertial forces do not play a significant role in the robot's dynamics) allows us to cast the problem of finding the shape of the structure assumes under the given distribution of forces as a problem of finding static equilibrium. For fixed rest lengths of the elastic elements that form tensegrity structure, the static equilibrium problem can be solved by minimizing potential energy of the structure, or equivalently, by solving a feasibility problem, given force equilibrium equations and force limits arising from the physical properties of the elements of which the structure consists: the cables can only experience tensile forces, while the struts are designed to experience compressive forces. Both result in nonlinear optimization problems. Fixing positions of the nodes, the feasibility problem can be solved as a single convex program, using force density variables [see Xu et al. (2018); Khafizov and Savin (2021)]. A more general form of this problem, known as form-finding, is discussed in a number of publications, including Tibert and Pellegrino (2003); Masic et al. (2005); Zhang et al. (2006).

Alternatively, data-driven and learning-based methods can be used for the same end. It was shown that a neural network can be used to predict the statically-stable configuration of the tensegrity structure, given the rest lengths of its elastic elements [see Zalyaev et al. (2020)]. Similarly, evolutionary methods for the design of statically stable structures have been proposed Paul et al. (2005). Note that the latter is close to topological tensegrity design methods based on force density variables, as presented in Xu et al. (2018); Kohestani (2020), as they focus on the design of the structures, rather than finding the static equilibrium of the existing ones. We note that algorithms mentioned in this paragraph are either limited to the specific types of tensegrity robots, or are computationally intensive, requiring the solution of optimization problems, or running neural networks.

One of the many challenges associated with motion planning for deformable robots is finding paths that require feasible deformations Mahoney et al. (2010). Simplified models of the deformable objects are an essential tool in motion planning for such robots Mahoney et al. (2010). In Gayle et al. (2005), deformable robot was modelled as an elastic system. One of the methods proposed in Bayazit et al. (2002) uses bounding box deformation to correct the shape of the robot. Here we propose to encode deformations with as positive-definite matrices, thus limiting possible deformations, but making the space of all considered deformations convex.

Taking advantage of the simplified encoding described above we propose a data-driven ellipsoid mapping, which allows us to rapidly search for tensegrity configurations compatible with the requested deformation, lying inside the requested ellipsoids. This is done by pre-computing covering Löwner-John ellipsoids and the use of eigendecomposition of these ellipsoids, which allows checking ellipsoid containment based on algebraic operations.

The rest of the paper is organized as follows. **Section 3** describes static equilibrium conditions for the structure. **Section 4** discusses the use of Löwner-John ellipsoids as bounding volumes, presents optimization problems associated with computing these ellipsoids, and gives forms of self-intersection problem for the structure; all three topics form the basis for dataset generation algorithm and the mapping method, presented in **Section 5**. **Section 6** presents motion planning based on the proposed method. The computational cost associated with the proposed algorithm is given in **Section 8**.

3 STATIC EQUILIBRIUM OF TENSEGRITY STRUCTURES

The main elements making a tensegrity structure are cables and struts, with cables experiencing purely tensile forces, and struts experiencing purely compressive forces. Here we focus specifically on a sub-class of tensegrity structures where no two struts are immediately connected; this is the class of tensegrity structures that have been the focus of the majority of the tensegrity-related research, due to the comparative ease of design of such structures.

The connection points between a strut and cables are called nodes. One of the ways to describe a configuration of a tensegrity structure is to describe the position of its nodes. Here we denote the position of the nodes by vector variables \mathbf{r}_i .

3.1 Static Equilibrium Conditions

In order for the structure to maintain static equilibrium, the sum of forces acting on each node should be zero. Denoting a force acting on the node \mathbf{r}_i from the node \mathbf{r}_j as $\mathbf{f}_{i,j}$, this condition can be written as:

$$\sum_{j=1}^n \mathbf{f}_{i,j}(\mathbf{r}_i, \mathbf{r}_j) = 0, \forall i \quad (1)$$

where n is the total number of nodes in tensegrity structure. Assuming that both the struts and the cables can be modelled as linear springs, the forces $\mathbf{f}_{i,j}$ acting between two nodes can be modelled as follows:

$$\mathbf{f}_{i,j} = C_{i,j} \mu_{i,j} (\|\mathbf{r}_i - \mathbf{r}_j\| - \rho_{i,j}) \frac{\mathbf{r}_i - \mathbf{r}_j}{\|\mathbf{r}_i - \mathbf{r}_j\|}. \quad (2)$$

where $\mu_{i,j}$ is the stiffness coefficient, $\rho_{i,j}$ is the rest length of the given elastic element (distance between the nodes when the force $\mathbf{f}_{i,j}$ equals zero) and $C_{i,j} \in \{0, 1\}$ is the connectivity identifier and is equal to zero if no connection exists between the two elements.

3.2 Stable Configuration Problem

Assuming that stiffness coefficients $\mu_{i,j}$ are given and constant, but node positions \mathbf{r}_i are functions of the rest lengths $\rho_{i,j}$, we can formulate the direct stable configuration problem: given $\rho_{i,j}$, find \mathbf{r}_i such that the system is in static equilibrium, and the inverse stable configuration problem: given \mathbf{r}_i , find such $\rho_{i,j}$ that the system is in static equilibrium. Denoting vector of all parameters \mathbf{r}_i as \mathbf{r} and vector of all parameters $\rho_{i,j}$ as ρ , we can define functions

$\text{DFF}(\cdot)$, $\text{IFF}(\cdot)$ solving direct and inverse stable configuration problems:

$$\mathbf{r} = \text{DFF}(\rho), \quad \rho = \text{IFF}(\mathbf{r}). \quad (3)$$

Multiple algorithms implementing functions (3) can be found in the literature. However, all implementations share a number of properties.

Firstly, we note that for tensegrity structures not in contact with the environment, simultaneous translations of all nodes do not influence the static equilibrium conditions. Thus, there are infinitely many solutions to direct stable configuration problem. Moreover, since a stable equilibrium configuration with a non-degenerate stiffness matrix corresponds to a local minimum of the potential energy of the structure, a small change in $\rho_{i,j}$ corresponds to a small shift in that local minimum, implying the existence of the solution to the perturbed problem.

Secondly, we note that for practical tensegrity structures [including tensegrity prism and 6-bar tensegrity structure, see Vespignani et al. (2018)] variables \mathbf{r} span a higher dimensional space than ρ ; this implies that there are configurations that cannot be stabilized, i.e., the inverse problem does not always have a solution.

4 SIMPLIFIED DESCRIPTIONS OF THE ROBOT AND THE ENVIRONMENT

In order to make the motion planning problem tractable, both the robot and the environment need to be described in a form that facilitates efficient computations. To this end we describe the environment as a collection of convex polytopic obstacle-free spaces \mathcal{S}_p , given as H-polytopes: $\mathcal{S}_p = \{\mathbf{r}: \mathbf{S}_p \mathbf{r} \leq \mathbf{h}_p\}$. In the following subsections, we demonstrate how a deformable drone shape can be represented, and how this representation interacts with the obstacle-free spaces.

4.1 Bounding Volume for a Tensegrity Structure

The method proposed in this paper requires the use of bounding volumes. Both bounding boxes and bounding ellipsoids can be used, but we limit our discussion to bounding ellipsoids. We use outer Löwner-John ellipsoid (the smallest volume ellipsoid that contains a given set of points) as a bounding volume containing all nodes of the tensegrity structure; **Figure 2** illustrates the geometry of Löwner-John ellipsoids.

Finding outer Löwner-John ellipsoid containing all nodes \mathbf{r}_i can be cast as a convex optimization [see Boyd et al. (2004)]:

$$\begin{aligned} & \underset{\mathbf{Y}, \mathbf{y}}{\text{minimize}} && \log \det \mathbf{Y}^{-1}, \\ & \text{subject to} && \|\mathbf{Y}\mathbf{r}_i + \mathbf{y}\| \leq 1, \quad \forall i, \end{aligned} \quad (4)$$

where \mathbf{Y} is positive semidefinite ellipsoid deformation matrix and \mathbf{y} is ellipsoid center. Resulting ellipsoid \mathcal{E} in set notation is given as $\mathcal{E} = \{\mathbf{r}: \|\mathbf{Y}\mathbf{r} + \mathbf{y}\| \leq 1\}$. In practice, minimization of \log

$\det \mathbf{Y}^{-1}$ is often replaced by maximization of $\det (\mathbf{Y})^{1/n}$ function.

4.2 Free-Space Containment

We need to be able to check if the ellipsoid representing the outer approximation of the tensegrity structure fits inside a given obstacle-free region \mathcal{S} . To this end, it is more convenient to use a dual ellipsoid representation $\mathcal{E} = \{\mathbf{Xr} + \mathbf{x}: \|\mathbf{r}\| \leq 1\}$. The connection with the primal representation is established by the following relations: $\mathbf{X} = \mathbf{Y}^{-1}$, $\mathbf{x} = -\mathbf{Y}^{-1}\mathbf{y}$. We do not attempt to find the dual representation of the outer Löwner-John ellipsoid as a single optimization problem; instead, we find the primal representation of the ellipsoid, and then convert it when needed.

Given an obstacle-free region $\mathcal{S}_p = \{\mathbf{r}: \mathbf{S}_p \mathbf{r} \leq \mathbf{h}_p\}$, where $\mathbf{S}_p = [\mathbf{s}_{1,p} \dots \mathbf{s}_{m,p}]^\top$, with $\mathbf{s}_{j,p}$ forming rows of the matrix \mathbf{S}_p in the H-representation of the obstacle-free region and $\mathbf{h}_p = [h_{1,p} \dots h_{m,p}]^\top$, finding inner Löwner-John ellipsoid in dual representation contained in \mathcal{S}_p can be written as the following optimization problem [see Boyd et al. (2004) for the derivation]:

$$\begin{aligned} & \underset{\mathbf{X}, \mathbf{x}}{\text{minimize}} && \log \det \mathbf{X}^{-1}, \\ & \text{subject to} && \|\mathbf{Xs}_{j,p}\| + \mathbf{s}_{j,p}^\top \mathbf{x} \leq h_{j,p}, \quad \forall j \end{aligned} \quad (5)$$

4.3 Self-Intersection

One of the typical problems for tensegrity structures is self-collisions and self-intersections. While self-collisions do not pose practical problems during the motion of the robot, they need to be taken into account when a configuration-space trajectory for such a robot is planned.

Assume a pair of nodes \mathbf{r}_1 and \mathbf{r}_2 is connected via a cable or a rod, and so is a pair of nodes \mathbf{r}_3 and \mathbf{r}_4 . In order to check if these two structural elements (rods or cables) of the structure intersect in a given configuration, we can solve the following linear problem:

$$\begin{aligned} & \text{find} && \mathbf{z}, \\ & \text{subject to} && \begin{cases} [(\mathbf{r}_1 - \mathbf{r}_2) \ (\mathbf{r}_3 - \mathbf{r}_4)]\mathbf{z} = \mathbf{r}_4 - \mathbf{r}_2 \\ \mathbf{z} \geq 0 \\ \|\mathbf{z}\|_\infty \leq 1, \end{cases} \end{aligned} \quad (6)$$

where \mathbf{z} determines the position of the intersection point with respect to the nodes \mathbf{r}_1 , \mathbf{r}_2 , \mathbf{r}_3 and \mathbf{r}_4 . If the problem (6) has a solution, there is an intersection between the two structural elements. In order to explicitly control the precision of the algorithm, we can add numeric tolerance threshold δ_{in} :

$$\begin{aligned} & \text{find} && \mathbf{z}, \\ & \text{subject to} && \begin{cases} \|[(\mathbf{r}_1 - \mathbf{r}_2) \ (\mathbf{r}_3 - \mathbf{r}_4)]\mathbf{z} - \mathbf{r}_4 + \mathbf{r}_2\|_2 \leq \delta_{in} \\ \mathbf{z} \geq 0 \\ \|\mathbf{z}\|_\infty \leq 1, \end{cases} \end{aligned} \quad (7)$$

The functions (6) and (7) can be useful in numeric dataset generation procedures, as will be shown in the next section.

4.4 Cable Slack

Another problem of tensegrity structures is cable slack. While it does not pose immediate problems by itself (as long as the stiffness matrix of the structure does not become degenerate), it may still lead to a more complex behaviour of the system. With that in mind we would like to avoid slackened cables.

One of the simplest ways to encode the taut cables constraint is to require that the current length of cables exceeds their rest length by a given margin l_m :

$$C_{i,j}(\|\mathbf{r}_i - \mathbf{r}_j\| - \rho_{i,j} - l_m) \geq 0, \quad \forall i, j. \quad (8)$$

We denote this constraint as $\text{slack}(\mathbf{r})$ further in the text.

5 MAPPING ELLIPSOIDS TO CONFIGURATIONS

5.1 Configuration Dataset

In order to map an ellipsoid to an actual robot configuration, we use a configuration dataset. Each configuration in the dataset is given by two sets of parameters: \mathbf{r} and ρ . There are two principal ways of obtaining the dataset: 1) collecting the data on statically stable configurations of a given tensegrity structure (*via* a motion capture system, for example), while measuring or estimating rest lengths of the elastic elements ρ , or 2) generating a number of sets of rest lengths ρ and solving the direct stable configuration problem. Here we use the second approach. Its main advantage is that it allows us to easily change the dataset, as the generation process involves solving a pair of small-size quadratic programs; its downside is the necessity to detect and discard infeasible configurations. The following subsections will elaborate on this last point.

Our process of generating the dataset involves the following steps. On the first step, a base rest length vector ρ_b is proposed. We will consider the statically stable configuration $\mathbf{r}_b = \text{DFF}(\rho_b)$, associated with ρ_b , as undeformed. On the second step, an array of perturbation directions $\rho_{d,1}, \dots, \rho_{d,N}$ is generated, where N is the number of directions in the dataset; we sampled $\rho_{d,i}$ it from a uniform distribution. For each direction $\rho_{d,i}$ we solve the direct form finding problem $\mathbf{r} = \text{DFF}(\rho_b + \alpha\rho_{d,i})$, moving the value of alpha from 0 to 1. For each configuration thus obtained we check self-intersection of the structural elements (rods and cables), using expression (7); if intersection did occur, the configuration is not stored, and no further configurations associated with this direction are computed. If no configuration occurred, the configuration \mathbf{r} and rest length vector $\rho = \rho_b + \alpha\rho_{d,i}$ are both stored in the dataset.

We augment the dataset with two additional fields that are used in the following sections: perturbation norm and eigendecomposition of the outer Löwner-John ellipsoid. We define the perturbation norm ν as the 2-norm of the perturbation $\Delta\rho$: $\nu = \|\Delta\rho\|$. It will later be used as a deformation measure.

In order to find the outer Löwner-John ellipsoid for the given configurations \mathbf{r} , we solve problem (4), obtaining corresponding matrix \mathbf{Y} . Then we find the following eigendecomposition:

$$\mathbf{Y}^{-1}\mathbf{V} = \mathbf{VD} \quad (9)$$

Thus found values of $\Delta\rho$, \mathbf{r} , ν , \mathbf{V} and \mathbf{D} form the dataset. The process of compiling the dataset is outlined as Algorithm 1, where Intersection (\cdot) checks if (7) is feasible, Löwner-John (\cdot) solves (4), and eigendecomposition (\cdot) returns an eigendecomposition of a matrix. Note that the perturbation norm ν is augmented with a regularization term $\xi \det(\mathbf{Y}^{-1})$, which aims at punishing configurations that result in low volume ellipsoids; ξ is a scalar regularization coefficient.

Algorithm 1. Dataset generation algorithm.

Algorithm 1 Dataset generation algorithm

Input: $\rho_b, \Delta\alpha$
Output: Dataset

```

1:  $\rho_{d,i} \sim \mathcal{N}(\cdot)$ ,  $1 \leq i \leq N$ 
2: while  $i \leq N$  do
3:   while  $j \leq 1/\Delta\alpha$  do
4:      $\rho = \rho_b + \Delta\alpha j \rho_{d,i}$ 
5:      $\mathbf{r} = \text{DFF}(\rho)$ 
6:     if Intersection( $\mathbf{r}$ ) or Slack( $\mathbf{r}$ ) then
7:       break
8:     else
9:        $(\mathbf{Y}, \mathbf{y}) = \text{Löwner-John}(\mathbf{r})$ 
10:       $(\mathbf{V}, \mathbf{D}) = \text{eigendecomposition}(\mathbf{Y}^{-1})$ 
11:       $\nu = \Delta\alpha j \|\rho_{d,i}\| + \xi \det(\mathbf{Y}^{-1})$ 
12:      Dataset.append( $\mathbf{r}, \rho, \mathbf{V}, \mathbf{D}, \nu$ )
13:    end if
14:     $j \leftarrow j + 1$ 
15:  end while
16:   $i \leftarrow i + 1$ 
17: end while
```

5.2 Dataset-Based Configuration Mapping

Given an ellipsoid in dual representation $\mathcal{E} = \{\mathbf{Xr} + \mathbf{x}: \|\mathbf{r}\| \leq 1\}$, we can rapidly check which of the configurations from the dataset can be contained in the ellipsoid \mathcal{E} . To do it we define containment matrix \mathbf{W} :

$$\mathbf{W} = \mathbf{X}^{-1}\mathbf{V}\mathbf{D} \quad (10)$$

With that we can assign a measure to each configuration, using either a step function or a log barrier. The expression for the latter is given below:

$$\nu = \nu + \phi(1 - \mathbf{e}_1^T \mathbf{W}^T \mathbf{W} \mathbf{e}_1) + \phi(1 - \mathbf{e}_2^T \mathbf{W}^T \mathbf{W} \mathbf{e}_2) + \phi(1 - \mathbf{e}_3^T \mathbf{W}^T \mathbf{W} \mathbf{e}_3), \quad (11)$$

$$\begin{cases} \phi(x) = -\delta \log(x) & \text{if } x > 0, \\ \nu = \infty & \text{otherwise,} \end{cases} \quad (12)$$

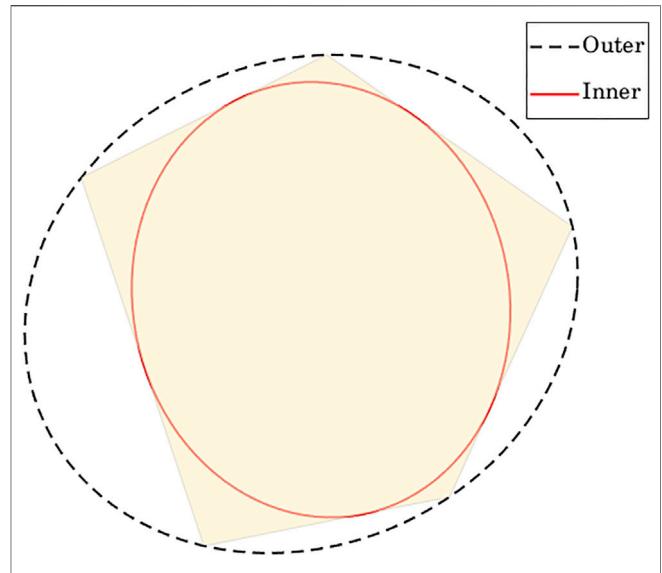


FIGURE 2 | A polytope with inner and outer Löwner-John ellipsoids.

where $\mathbf{e}_1 = [1 \ 0 \ 0]^T$, $\mathbf{e}_2 = [0 \ 1 \ 0]^T$, and $\mathbf{e}_3 = [0 \ 0 \ 1]^T$ are unit vectors forming axes of the world coordinate frame.

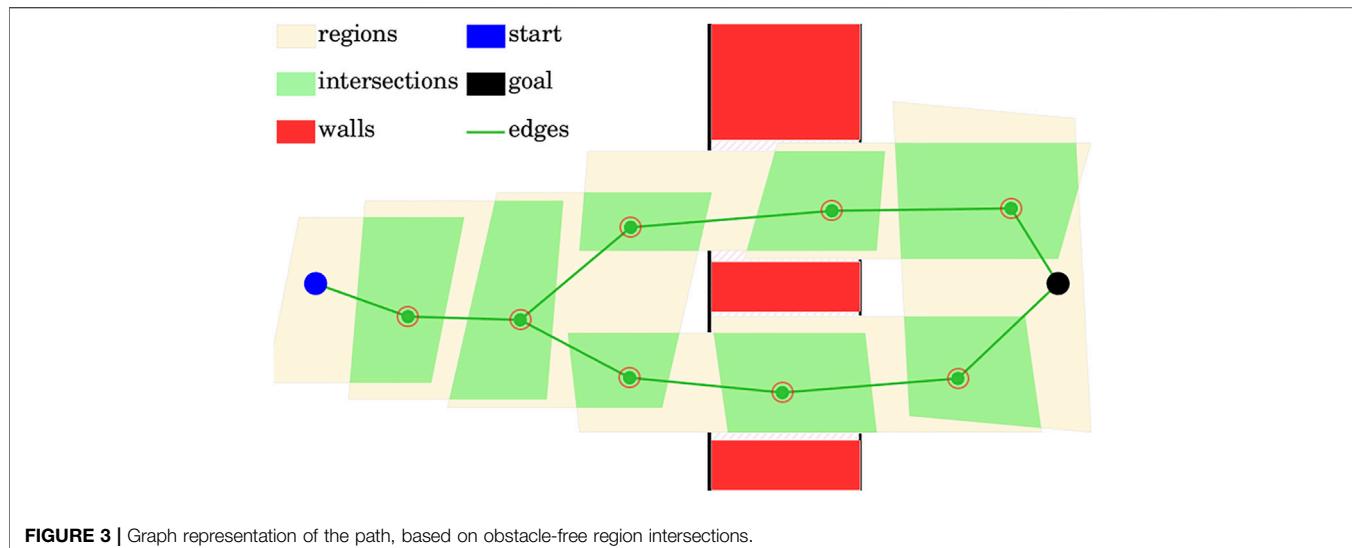
Then, finding the configuration that corresponds to the smallest ν we find the desired map. Note that the map is not designed to find the closest possible configuration to the given ellipsoid, and instead it tries to find the one with the least deformation that can fit in it.

5.3 Drone-Specific Considerations

The use of a tensegrity structure as a frame of a drone entails a large number of problems. This paper does not aim to present a comprehensive study or give solutions to these problems; however, we can list a few important considerations connected with the proposed dataset generation method that need to be addressed when using the method for planning a sequence of deformation for a tensegrity drone.

Firstly, depending on the number of motors and on how they are attached and oriented, some configurations of the tensegrity structure might be impossible to stabilize in the air. These should be identified and removed. The process of identifying these configurations depends both on the position of the motors and the control algorithm designed for the drone.

Secondly, the geometry of the propellers needs to be taken into account when the configuration of the tensegrity is being evaluated as admissible or not. If the volume covered by the propellers can be approximated as a convex polytope, and if the position of this volume can be approximated as a linear form of the positions of the nodes of the structure, then the problem of finding self-intersections involving the propellers can also be cast as a quadratic problem. The same can be said about other structural elements of the drone: the batteries, the controller and transmission boards, etc.



6 QUASI-STATIC DEFORMATION-ENABLED MOTION PLANNING

The goal of the quasi-static deformation-enabled motion planning is to provide a sequence of positions and deformations of the robot, as it moves from the initial to the goal point, such that no constraints are violated. Here we only consider constraints associated with containment in the obstacle-free regions.

6.1 Obstacle-free Region Intersections

To simplify computations we make a conservative assumption that in order for the robot to be able to pass from one region to another, it has to be able to fit in the intersection of the two regions. This assumption can be prohibitive if the free space is tessellated into non-intersecting obstacle-free regions. However, it can be made less conservative if maximum-volume convex obstacle-free regions are generated from seed points [as was done in Deits and Tedrake (2015); Savin (2017)], and the free space is densely populated by the seed points.

Note that the requirement that two regions have a sufficiently large intersection in order for the robot to be able to pass from one to the other allows treating the obstacle-free regions as graph edges, and intersections as graph vertices. Therefore a path from the region containing the initial point to the region containing the goal point is also a path along this graph. Thus the path can be found by graph search methods, such as A* algorithm. The heuristic for the search can be based on the distances between the centers of the region intersections that share a region, as well as the width of the region; both can be approximated by finding the inner Löwner-John ellipsoid (5), for the intersection of the regions with \mathbf{x} giving the ellipsoid's center acting as an approximation of the region's center, and the width of the ellipsoid acting as an approximation of the region's width.

Figure 3 illustrates the graph representation of the path, based on obstacle-free region intersections. The graph nodes and graph edges are shown, generating at least two paths from the starting point to the goal point, passing through different openings in the wall (its intersection shown in red). Note that on the picture the obstacle-free space is not densely populated with large obstacle-free regions, and hence their intersections are unnecessarily small, illustrating the conservativeness of the requirement in this case.

6.2 Inflation-Based Deformation Planning

In this subsection we assume that a single path from the initial point to the goal point is given in form of a sequence of regions $\mathcal{S}_1, \dots, \mathcal{S}_n$, and the robot needs to pass through intersections of all consecutive regions in that sequence. The proposed inflation-based deformation planning methods consists of two steps: 1) find the biggest possible ellipsoid that fits in each region and in each intersection that needs to be passed, 2) for each ellipsoid, find a robot configuration that can fit in it with the *least deformation* v .

To check if the ellipsoid simultaneously lies in two regions $\mathcal{S}_p = \{\mathbf{r}: \mathbf{S}_p \mathbf{r} \leq \mathbf{h}_p\}$ and $\mathcal{S}_{p+1} = \{\mathbf{r}: \mathbf{S}_{p+1} \mathbf{r} \leq \mathbf{h}_{p+1}\}$, where $\mathbf{S}_p = [\mathbf{s}_{1,p} \dots \mathbf{s}_{m_1,p}]^\top \in \mathbb{R}^{m_1 \times 3}$ and $\mathbf{S}_{p+1} = [\mathbf{s}_{1,p+1} \dots \mathbf{s}_{m_2,p+1}]^\top \in \mathbb{R}^{m_2 \times 3}$, it is sufficient to check if it lies in their intersection, whose H-representation is constructed by concatenation: $\mathcal{S}_p \cap \mathcal{S}_{p+1} = \{\mathbf{r}: [\mathbf{S}_p^\top \mathbf{S}_{p+1}^\top]^\top \mathbf{r} \leq [\mathbf{h}_p^\top \mathbf{h}_{p+1}^\top]^\top\}$. The problem (5) then takes form:

$$\begin{aligned} & \underset{\mathbf{x}_{p,p+1}, \mathbf{x}_{p+1}}{\text{minimize}} && \log \det \mathbf{X}_{p,p+1}^{-1}, \\ & \text{subject to} && \begin{cases} \|\mathbf{X}_{p,p+1} \mathbf{s}_{j,p}\| + \mathbf{s}_{j,p}^\top \mathbf{x}_{p,p+1} \leq h_{j,p}, & 1 \leq j \leq m_1, \\ \|\mathbf{X}_{p,p+1} \mathbf{s}_{k,p+1}\| + \mathbf{s}_{k,p+1}^\top \mathbf{x}_{p,p+1} \leq h_{k,p+1}, & 1 \leq k \leq m_2. \end{cases} \end{aligned} \quad (13)$$

Note that problem (5) can be solved independently for each region \mathcal{S}_p and each region intersection $\mathcal{S}_p \cap \mathcal{S}_{p+1}$. This alleviates numerical problems associated with the growth in the number of variables in the planning problem.

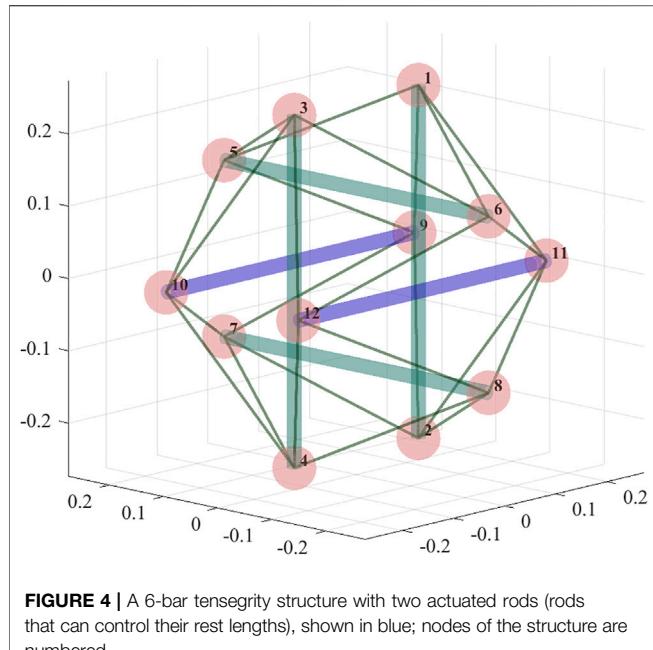


FIGURE 4 | A 6-bar tensegrity structure with two actuated rods (rods that can control their rest lengths), shown in blue; nodes of the structure are numbered.

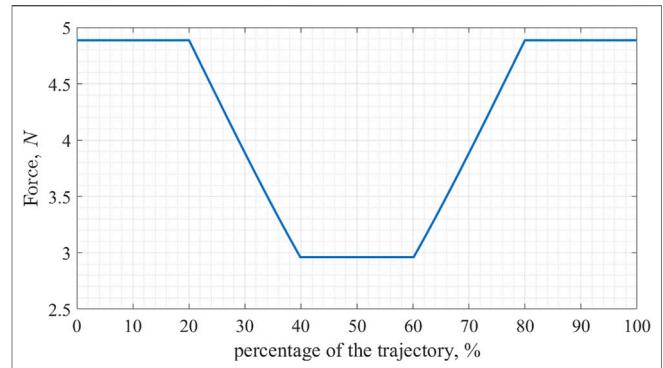


FIGURE 5 | Evolution of the magnitude of tensile forces for two cables; cable 1 connects nodes 10 and 3, and cable 2 connects nodes 8 and 12.

The second step of the process consists in mapping found ellipsoids to configurations from the dataset. On this step we go through found ellipsoids $E_p = \{\mathbf{X}_p \mathbf{r} + \mathbf{x}_p; \|\mathbf{r}\| \leq 1\}$ and $E_{p,p+1} = \{\mathbf{X}_{p,p+1} \mathbf{r} + \mathbf{x}_{p,p+1}; \|\mathbf{r}\| \leq 1\}$ sequentially, and for each of them we use (10) and (11) to determine the deformation value v of each configuration in the dataset; the configuration with the smallest v is chosen as the image of the respective ellipsoid. This process is described in the Algorithm 2.

Algorithm 2. Ellipsoids mapping.

```

Input: Dataset,  $\mathbf{X}_{1,2}^{-1}, \dots, \mathbf{X}_{n-1,n}^{-1}$ 
Output:  $\mathbf{r}_p, \rho_p$ 
1: while  $p \leq n - 1$  do
2:   while  $i \leq \text{Dataset.length}$  do
3:      $(\mathbf{r}_i, \rho_i, \mathbf{V}_i, \mathbf{D}_i, \nu_i) := \text{Dataset.retrieve}(i)$ 
4:      $\mathbf{W}_i := \mathbf{X}_{p,p+1}^{-1} \mathbf{V}_i \mathbf{D}_i$ 
5:      $v_i := \nu_i + \sum_{j=1}^3 \phi \left( 1 - \mathbf{e}_j^\top \mathbf{W}_i^\top \mathbf{W}_i \mathbf{e}_j \right)$ 
6:      $i \leftarrow i + 1$ 
7:   end while
8:    $I = \text{argmin}(v_i)$ 
9:    $\mathbf{r}_p = \mathbf{r}_I, \rho_p = \rho_I$ 
10:   $p \leftarrow p + 1$ 
11: end while

```

The described above algorithm can be used for planning deformations of a drone, as it passes through narrow spaces. However, let us note that the algorithm only provides the desired deformations (and corresponding node positions and cable rest lengths) for a few trajectory nodes; hence, the evolution of the nodes and the evolution of the cable rest lengths need to be determined separately, e.g., via interpolation. With that in mind it is of further interest to study resulting trajectories. Such a study is presented in the following section.

Additionally, let us remark that if the additional safety margins are required, the ellipsoids representing obstacle-free regions can be scaled down by a given amount. Alternatively, the required safety margins can be checked directly on the configurations found by the algorithm; this might be preferable, as the use of ellipsoids to approximate obstacle-free spaces already makes the algorithm conservative.

7 SIMULATION RESULTS

In this section, we provide simulation results, which illustrate the work of the algorithm. As was mentioned in the last section, the algorithm is providing a sequence of deformations, but between each pair of consecutive deformations, the rest lengths of the elastic elements should be determined separately. Here we take the simplest case when the linear interpolation is used. We study the resulting trajectories in terms of the mechanical properties of the structure as it undergoes deformation.

In order to provide a higher range of deformation, while maintaining the orientation of the rods (which is preferable for a drone where rotors are rigidly attached to the rods) we choose to actuate symmetrically placed rods. In Figure 4 the actuated rods are shown in blue, while the unactuated cables and rods are shown in green. Actuated rods connect nodes 9 and 10, and nodes 11 and 12, as shown in Figure 4.

Note that the practicality of such actuation pattern depends on the ability of the drone to effectively change and maintain its orientation; if the drone can be relied on to maneuver itself in such a way as to face the narrow opening with the actuated horizontal rods being orthogonal to the drone trajectory as it traverses the opening, then the actuation pattern is justified, as the other pair of horizontal rods would be oriented tangentially to the trajectory, not influencing the ability of the robot to fit through the opening. If the opening is horizontal rather than vertical (wide and short rather than narrow and tall), the vertical rods would also require to be actuated. In this section, we consider the simpler case for a cleaner presentation.

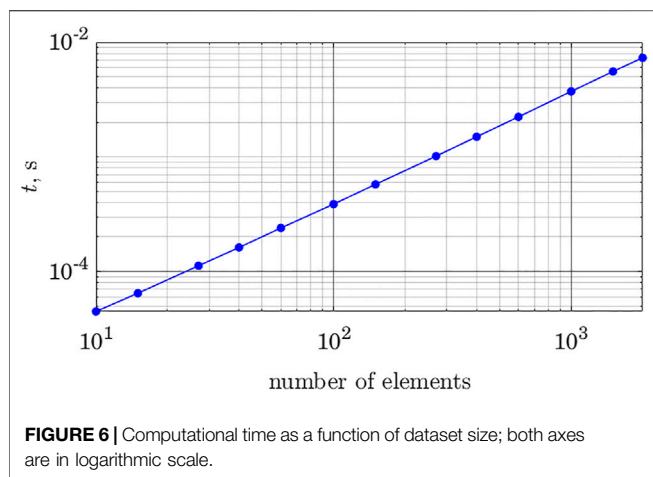


FIGURE 6 | Computational time as a function of dataset size; both axes are in logarithmic scale.

We consider the case when the structure has to pass through a tall opening with a width of 0.35 m. The undeformed width of the structure is 0.488 m. Two possible concerns with the resulting trajectory are 1) cable slack, and 2) excessive tension in some of the cables.

Firstly, we study the evolution of the tensile forces in the structure's cables. As a representative example we choose the cable between nodes 10 and 3 (connecting one of the actuated rods with one of the vertical ones) denoted as cable 1, and a cable between nodes 8 and 12, denoted as cable 2, using the notation in **Figure 4**. **Figure 5** shows the evolution of the tensile forces in the cables along the trajectory. The first 20% of the trajectory the structure maintains its shape, next 20% it deforms, then it maintains the deformed shape (during which period the structure is required to pass through the opening), then the structure deforms back into its original shape and maintains it until the end of the trajectory. Note that the timing is chosen arbitrarily; since the deformation planning requires purely geometric information, the timing of the trajectory can be planned separately taking into account its dynamical properties.

As we can see in **Figure 5** the tensile force in the cable goes down during the deformation, and then back up afterward. At its lowest point, it goes down by 39.4% for the cable 1 and by 60% for the cable 2. With the trajectory planned for this experiment, no cable loses more than 60% of its initial tensile force value, and only cables connecting nodes 6, 8, 11, and 12 do so.

We should note that while in this experiment the tensile forces were not increasing throughout the motion and the relative decrease of these forces was limited to 60% of their initial values, the dataset used in generating the trajectory does include a higher range of force changes. Overall in the dataset of 200 elements, the highest recorded tensile force exceeds the nominal value by 55.9%, while the smallest is only 4.78% of the nominal value. Whether or not these values can lead to mechanical damage of the structure, the loaded elements, or the cables themselves depends on a choice of materials and the design of mechanical parts (e.g., connectors between the cables and struts).

Presented here study can serve as an illustration of how the analysis of the trajectory quality can be done in practice. In our

case we can be satisfied that the structure is capable of following the found trajectory since the tensile forces are clearly within the nominal range, but for arbitrary trajectories generated with the same dataset, we need to check the ability of the cables to withstand a 55.9% increase in the tensile force magnitudes. If such an increase of tensile force magnitude is undesirable, the corresponding elements can be removed from the dataset, or a re-design of the cables can be suggested.

8 COMPUTATIONAL COST

In order to assess the computational cost of the algorithm, we need to recognize that the number of optimization programs (13) to be executed is equal to the number of intersections in the path. The number of variables in the problem formulation is independent of the problem geometry, but the number of constraints can change, subject to the shape of the region intersections.

Another source of growth in the computational time relates to the dataset search. Algorithm 2 requires checking ellipsoids found by the solver against each entry in the dataset, making the number of checks proportional to the dataset size. **Figure 6** demonstrates how the computational time depends on the number of entries in the dataset. The time cost is computed for the algorithm run on a PC with Intel i7 9700K processor, with 32 Gb memory and the dataset stored on an SSD.

The graph was produced by running the algorithm 10^3 times and taking an average, displayed as the data point. As implied by Algorithm 2, the number of operations (matrix multiplications and computation of logarithms) depends linearly on the number of entries in the dataset.

9 CONCLUSION

In this paper we presented a method for deformation-aware trajectory planning for deformable tensegrity drones. Employing two conservative assumptions (over-approximating the shape of tensegrity drone as an ellipsoid, and assuming that in order to pass between two regions, the drone needs to pass through their intersection) the proposed method takes advantage of the convexity of the resulting problem, as well as of the possibility to separate it into a number of independent optimization problems, avoiding the growth in the number of decision variables with the increase in the size of the path to be planned. Pre-computing Löwner-John ellipsoids and their eigendecomposition allow to keeping computational cost of the algorithm relatively low, with the linear growth in the number of elements in the dataset.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

SS: original idea, code development, numerical studies; AK: analysis, graphics, editing.

REFERENCES

- Anshelevich, E., Owens, S., Lamirault, F., and Kavraki, L. E. (2000). "Deformable Volumes in Path Planning Applications," in Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), San Francisco, CA, USA, April 24-28, 2000 (IEEE), 2290–2295. doi:10.1109/robot.2000.846368
- Bayazit, O. B., Lien, J.-M., and Amato, N. M. (2002). "Probabilistic Roadmap Motion Planning for Deformable Objects," in Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Washington, DC, USA, May 11-15, 2002 (IEEE), 2126–2133.
- Boyd, S., Boyd, S. P., and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge, United Kingdom: Cambridge University Press.
- Deits, R., and Tedrake, R. (2015). "Computing Large Convex Regions of Obstacle-free Space through Semidefinite Programming," in *Algorithmic Foundations of Robotics XI*. Heidelberg: Springer, 109–124. doi:10.1007/978-3-319-16595-0_7
- Deng, Y., Zhang, Y., He, X., Yang, S., Tong, Y., Zhang, M., et al. (2020). Soft Multicopter Control Using Neural Dynamics Identification. Available at: <https://arxiv.org/abs/2008.07689>. (November 10, 2021).
- Gayle, R., Lin, M. C., and Manocha, D. (2005). "Constraint-based Motion Planning of Deformable Robots," in Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, April 18-22, 2005 (IEEE), 1046–1053.
- Khafizov, R., and Savin, S. (2021). "Length-constrained Mixed-Integer Convex Programming-Based Generation of Tensegrity Structures," in 2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM), Chongqing, China, July 3-5, 2021 (IEEE), 125–131. doi:10.1109/icarm52023.2021.9536138
- Kimber, J., Ji, Z., Petridou, A., Sipple, T., Barhydt, K., Boggs, J., et al. (2019). "Low-cost Wireless Modular Soft Tensegrity Robots," in 2019 2nd IEEE International Conference on Soft Robotics (RoboSoft), Seoul, Korea, April 14-18, 2019 (IEEE), 88–93. doi:10.1109/robosoft.2019.8722723
- Koohestani, K. (2020). Innovative Numerical Form-Finding of Tensegrity Structures. *Int. J. Sol. Structures* 206, 304–313. doi:10.1016/j.ijsolstr.2020.09.034
- Lee, H., Jang, Y., Choe, J. K., Lee, S., Song, H., Lee, J. P., et al. (2020). 3d-printed Programmable Tensegrity for Soft Robotics. *Sci. Robot* 5, 61–64. doi:10.1126/scirobotics.ayy9024
- Liu, Y., Bi, Q., Yue, X., Wu, J., Yang, B., and Li, Y. (2022). A Review on Tensegrity Structures-Based Robots. *Mechanism Machine Theor.* 168, 104571. doi:10.1016/j.mechmachtheory.2021.104571
- Mahoney, A., Bross, J., and Johnson, D. (2010). "Deformable Robot Motion Planning in a Reduced-Dimension Configuration Space," in 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, May 3-7, 2010 (IEEE), 5133–5138. doi:10.1109/robot.2010.5509649
- Masic, M., Skelton, R. E., and Gill, P. E. (2005). Algebraic Tensegrity Form-Finding. *Int. J. Sol. Structures* 42, 4833–4858. doi:10.1016/j.ijsolstr.2005.01.014
- Mirletz, B. T., Park, I.-W., Flemons, T. E., Agogino, A. K., Quinn, R. D., and SunSpiral, V. (2014). "Design and Control of Modular Spine-like Tensegrity Structures," in The 6th World Conference of the International Association for Structural Control and Monitoring (6WCSCM), Barcelona, Spain, July 15-17, 2014, 4–10.
- Moll, M., and Kavraki, L. E. (2006). Path Planning for Deformable Linear Objects. *IEEE Trans. Robot.* 22, 625–636. doi:10.1109/tr.2006.878933
- Paul, C., Lipson, H., and Cuevas, F. J. V. (2005). "Evolutionary Form-Finding of Tensegrity Structures," in GECCO '05: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, June 2005, 3–10. doi:10.1145/1068009.1068011
- Rieffel, J., and Mouret, J.-B. (2018). Adaptive and Resilient Soft Tensegrity Robots. *Soft robotics* 5, 318–329. doi:10.1089/soro.2017.0066
- Roussel, O., Taix, M., and Bretl, T.. (2014). *Motion Planning for a Deformable Linear Object*. In *European Workshop on Deformable Object Manipulation*. Lyon, France, 153.
- Sabelhaus, A. P., Bruce, J., Caluwaerts, K., Manovi, P., Firooz, R. F., Dobi, S., et al. (2015a). "System Design and Locomotion of Superball, an Untethered Tensegrity Robot," in 2015 IEEE international conference on robotics and automation (ICRA), Seattle, WA, USA, May 26-30, 2015 (IEEE), 2867–2873. doi:10.1109/icra.2015.7139590
- Sabelhaus, A. P., Ji, H., Hylton, P., Madaan, Y., Yang, C., Agogino, A. M., et al. (2015b). "Mechanism Design and Simulation of the Ultra Spine: a Tensegrity Robot," in International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Boston, Massachusetts, USA, August 2-5, 2015 (American Society of Mechanical Engineers) 57120. V05AT08A059. doi:10.1115/detc2015-47583
- Savin, S. (2017). "An Algorithm for Generating Convex Obstacle-free Regions Based on Stereographic Projection," in 2017 International Siberian Conference on Control and Communications (SIBCON), Astana, Kazakhstan, June 29-30, 2017 (IEEE), 1–6. doi:10.1109/sibcon.2017.7998590
- Shah, A. J., and Shah, J. A. (2016). "Towards Manipulation Planning for Multiple Interlinked Deformable Linear Objects," in 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, May 16-21, 2016 (IEEE), 3908–3915. doi:10.1109/icra.2016.7487580
- Tibert, A. G., and Pellegrino, S. (2003). Review of Form-Finding Methods for Tensegrity Structures. *Int. J. Space Structures* 18, 209–223. doi:10.1260/026635103322987940
- Vespignani, M., Friesen, J. M., SunSpiral, V., and Bruce, J. (2018). "Design of Superball V2, a Compliant Tensegrity Robot for Absorbing Large Impacts," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, October 1-5, 2018 (IEEE), 2865–2871. doi:10.1109/iros.2018.8594374
- Xu, X., Wang, Y., and Luo, Y. (2018). Finding Member Connectivities and Nodal Positions of Tensegrity Structures Based on Force Density Method and Mixed Integer Nonlinear Programming. *Eng. Structures* 166, 240–250. doi:10.1016/j.engstruct.2018.03.063
- Zalyaev, E., Savin, S., and Vorochaeva, L. (2020). "Machine Learning Approach for Tensegrity Form Finding: Feature Extraction Problem," in 2020 4th Scientific School on Dynamics of Complex Networks and their Application in Intellectual Robotics (DCNAIR), Innopolis, Russia, September 7-9, 2020 (IEEE), 265–268. doi:10.1109/dcnair50402.2020.9216799
- Zappetti, D., Arandes, R., Ajanic, E., and Floreano, D. (2020). Variable-stiffness Tensegrity Spine. *Smart Mater. Struct.* 29, 075013. doi:10.1088/1361-665x/ab87e0
- Zhang, L., Maurin, B., and Motro, R. (2006). Form-finding of Nonregular Tensegrity Systems. *J. Struct. Eng.* 132, 1435–1440. doi:10.1061/(asce)0733-9445(2006)132:9(1435)
- Conflict of Interest:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.
- Publisher's Note:** All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.
- Copyright © 2022 Savin and Klimchik. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.

FUNDING

The research is supported by grant of the Russian Science Foundation (project No:19-79-10246).