Alexander Spinos[1]
Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104
e-mail: spinos@seas.upenn.edu

Devin Carroll
Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104
e-mail: cdevin@seas.upenn.edu

Terry Kientz
Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104
e-mail: tkientz@seas.upenn.edu

Mark Yim
Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, PA 19104
e-mail: yim@seas.upenn.edu

Topological Reconfiguration Planning for a Variable Topology Truss

This paper introduces a new class of self-reconfigurable robot: the variable topology truss (VTT), which is an extension of an existing class of robots: the variable geometry truss (VGT). Variable topology trusses have the additional capability to change the topology of the truss through self-reconfiguration by merging and splitting the nodes of the truss. We first introduce a hardware platform that enables this reconfigurability. We give a procedure to compute all possible distinct reconfiguration actions for a given robot topology. We show that 18 members are required for a minimal reconfigurable VTT, and we exhaustively enumerate all possible reconfigurable topologies for VTTs up to 29 members. Lastly, we introduce the topology network, which describes the relationship between these reconfigurable topologies. The topology network concept enables some high-level planning and provides insights into the design of truss topologies. [DOI: 10.1115/1.4050530]

Keywords: mechanism design, parallel platforms, robot design, theoretical kinematics

1 Introduction

In many applications, trusses are ideal due to their high structural efficiency, as they consist of members that are only in pure tension or compression, resulting in reduced maximum stress [1]. The class of robots commonly known as variable geometry trusses (VGTs) can be obtained by replacing some or all of the members in a truss with linear actuators [2]. There are many applications for this type of robot, including parallel manipulators [3], long-chain actuators, collapsible structures for space applications [4], and locomotion platforms. Figure 1 shows a VGT with a high expansion/compression ratio for octahedral locomotion.

A variable topology truss (VTT) starts from the same truss framework as a VGT: linear actuators form the members (beam elements in the truss) and passive spherical joints connect the members at the nodes in the truss. However, the truss has the additional capability to self-reconfigure, changing its topology by merging or splitting nodes. That is, two separate nodes in the truss can dock to form one single node which connects all of the involved members. Similarly, a single node with a sufficient amount of members can undock into a pair of nodes. In this way, a VTT can be thought of as a chain-type self-reconfigurable robot [5] consisting only of linearly actuated elements and freely movable spherical joints.

While VGTs only have control over the shape or geometry of the truss, a VTT can additionally change the topology of the underlying truss. This brings benefits typically associated with reconfigurable robots; the VTT can select the topology which is most suited to the task at hand. For example, a single VTT may have the capability to self-reconfigure to suit many applications: a collapsible topology for storage, a multi-limbed topology for locomotion, or a long thin topology for an extended antenna application. This task flexibility is not possible with the single fixed topology of a VGT. An example of two structures which might be realizable from one system is shown in Fig. 2.

The additional reconfigurability of a VTT could have applications in space missions as a lightweight adaptable structure, or as versatile structural reinforcement to assist first responders during disaster scenarios. Our prior work [6] explores the challenges of the latter application using an earlier VTT concept. In other related work, we have developed methods for fast local path planning for VTTs. These methods use sampling-based planning [7], grid-based planning [8], or cell decomposition methods to move single nodes [9] or pairs of nodes [10] at a time. In this work, we introduce the hardware design and investigate the topological reconfiguration constraints. This allows us to answer questions

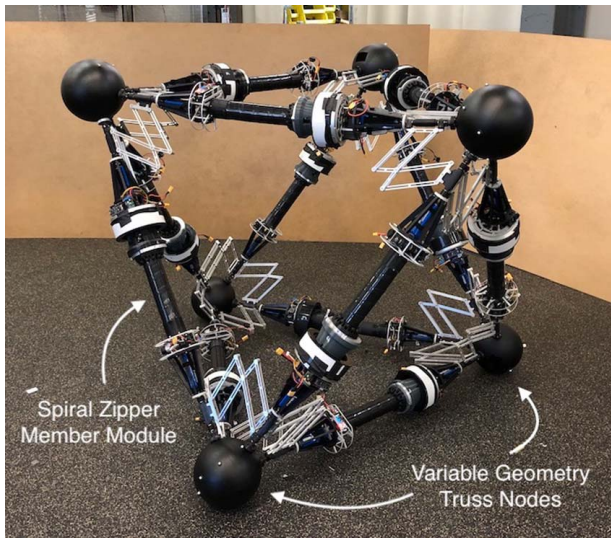

Fig. 1 An octahedral variable geometry truss constructed using our Spiral Zipper actuators

[1]Corresponding author.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the Journal of Mechanisms and Robotics. Manuscript received November 25, 2020; final manuscript received March 6, 2021; published online April 9, 2021. Assoc. Editor: Clement Gosselin.

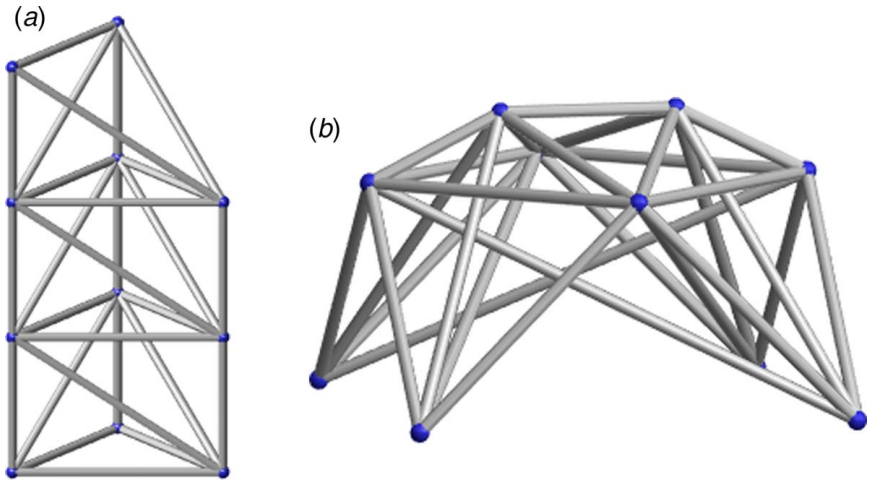

Fig. 2 A (*a*) tower and (*b*) dome configuration each with 27 members. We are interested in determining whether it is possible to reconfigure from one to the other.

about what truss topologies are achievable for a given VTT system and what intermediate topologies the system must pass through in order to reconfigure from one topology to another.

This paper extends a previous conference version of our work [11]. Whereas the previous work focuses on the mechanical design of the members and nodes (shown in Sec. 3) and introduces the concept of a topology network (Sec. 4); after the introduction and background section (Secs. 1 and 2), this new paper adds an analysis on the computational complexity for reconfiguration with explicit enumeration of truss topologies with up to 29 members (Sec. 5). In addition, in the prior conference version, we presented a six-bar VTT joint with a peg-in-hole latching design. In Sec. 3, we present the prototyped twelve-bar VTT joint with a pivoting catch latching system that has a larger area of acceptance.

2 Background

The Stewart platform is one example of a VGT; it can be thought of as an octahedral truss with six-actuated members [2]. Rhodes and Mikulas investigated the use of VGTs as a collapsable, controllable beam for space applications [12]. The Tetrobot system is another example of a VGT with a variety of configurations, including a six-legged walker [13]. Lee and Sanderson presented dynamic rolling locomotion with an icosahedral Tetrobot [14]. The dynamics and control of general VGTs have been studied as well [15].

Many existing VGTs and families of VGTs are constructed from basic unit cells [2,16,17]. For example, Bolker and Roth claim to enumerate all possible VGTs, specifying that they must be statically determinate and composed of convex polyhedral cells. However, these criteria are too restrictive, as they eliminate many interesting structures. For example, the complete bipartite graph $K_{5,5}$ is generically rigid in 3D space [18], but is neither statically determinate nor composed of polyhedral cells. By removing a single edge, it can be made statically determinate; yet, it is still not composed of polyhedral cells. The only criterion we specify for a VTT in this paper is that the truss is a rigid framework. We consider all suitably rigid bar and joint structures, including statically indeterminate ones. Statically indeterminate structures are necessary to allow reconfiguration of the truss, since merging two nodes in a statically determinate truss results in an overdetermined structure.

A critical design parameter in improving the usefulness of VGT-type systems is the range of motion of the prismatic joints. In the past work referenced above, all of the actuators were standard single-stage linear actuators with an extension/compression ratio of less than two. Larger extension ratios will allow the systems to have larger workspaces or smaller storage volumes as well as higher dexterity over these larger workspaces.

3 Mechanical Design and Validation

VGTs and VTTs achieve geometric shape change by actuating member lengths. In a truss with variable member lengths, the nodes must consist of passive revolute joints. In the case of a VTT, where nodes may merge or split, the joints must also be chainable to support the connection and disconnection of an arbitrary number of members at a single node.

The *member module* pictured in Fig. 3 is the fundamental unit of the VTT. It consists of an active prismatic joint with two passive chainable spherical joints, one attached at each end. Figure 1 has twelve of these member modules joined together with passive (non-reconfigurable) sphere nodes.

3.1 Active Prismatic Joint. The ideal prismatic joint is one with a large extension/compression ratio. Collins and Yim have developed such an actuator called the Spiral Zipper [19], which can be seen in Fig. 3 as the main column of the member module. In other applications, the Spiral Zipper has demonstrated extension ratios in excess of 14:1.

The Spiral Zipper operates both as a prismatic actuator and a structural component. It extends and retracts a rigid, lightweight tube by nesting the features in the top of a band to the matching features on the bottom. The extension ratio is only limited in principle by the amount of stored band and the material strength. The band forms a circular tube which is the optimal shape for stiffness-to-weight ratio under buckling loads, the most likely mode of failure for slender truss beams. The zipping feature interface is strongest in compression, although it can support small torsions and moments. To support tension, the actuator has a reinforcing winched cable on the inside of the tube. Experimentally, we have shown strength-to-weight ratios of approximately 10:1 for a prismatic actuator capable of extending 1 m with a plastic band. Spiral Zipper tubes made of acetal plastic, 1.5 m long, have been shown to support 530 N. We have also shown the tube can be extended as fast as 0.45 m/s.

In the member module, we use enough band to give the column an 8:1 extension ratio. The spherical joints on each end add some additional fixed length to the member, resulting in an overall member extension ratio of about 2.5:1.

3.2 Passive Chainable Spherical Joints. To maintain the advantageous decoupling of moments and forces, the system needs passive spherical joints at the nodes of the truss. The joints should satisfy the following:

(1) All members joined at a node must be constrained so that their longitudinal axes pass through the node center, but

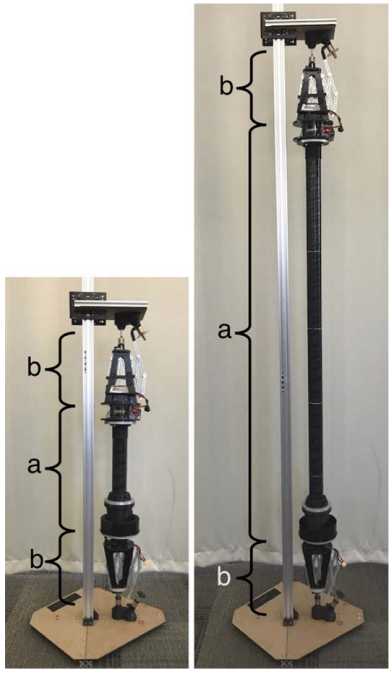

Fig. 3 A truss member module in a vertical test rig. The (a) spiral zipper actuator is the length-changing element in the member module and (b) chainable spherical joints are attached at both ends.

the members should be free to rotate in any of the degrees of rotational freedom about the node center.

(2) The node must be able to constrain an arbitrary number of truss members. The number of members at a node may change during topological reconfiguration.

(3) The nodes must minimize interference between members as the angles between members change.

To satisfy the first condition in practice, the truss members themselves do not need to physically occupy the node center as long as they share a common virtual center of rotation. Universal joints and spherical ball joints are typical candidates for a truss node; however, they are not chainable to support merging of nodes. Instead, a spherical link chain, made up of links of revolute joints which rotate about the node center, can provide this behavior [20].

A mechanism similar to a concentric multilink spherical (CMS) joint first developed by Hamlin [21] for the Tetrobot Modular Robot satisfies the design, in particular reducing interference. To provide the ability to attach and detach multiple joints at a single node, which was not part of the CMS joint, we add an attachment post and a pivoting catch which can dock to this attachment post. We call this type of spherical joint a *VTT joint*, it is illustrated in Fig. 4.

There is one additional practical concern that is relevant for terrestrial VTT applications, where the truss may need to apply forces to the environment. It is desirable to have the nodes that touch the environment do so in such a way that all external forces pass through the node center. The simplest way to achieve this is to place a physical sphere in the node center. However, a reconfiguring VTT cannot have a sphere in the center of every node, since this would prohibit merging two nodes that both have spheres. The placement of physical spheres in the truss nodes depends on the application and expected motion plan for a VTT.

The CMS joint is characterized by an offset planar hinge that forms a six-bar linkage. In Tetrobot, this offset allows multiple joints to be connected together manually to a single truss member using a shoulder screw. In the VTT joint, shown in Fig. 4, one end of the linkage is permanently attached to the truss member, along with an alignment post on the longitudinal axis of the member. The distal end of the linkage is a catch that can dock with the alignment posts on other members. When the catch of one member is docked with the alignment post of another member, the two members are constrained to move around the same spherical center. When the endpoints of multiple member modules are brought together, the spherical joints can dock to form a chain of arbitrary length with a free alignment post and catch at either end of the chain. Then, all the members at a node are constrained to move about the same spherical center. Figure 5 shows a chain of three VTT joints around a sphere.

To latch two member modules together, the pivot catch on the end of one linkage approaches the central post on the other member module. Once contact is made, guiding features on the

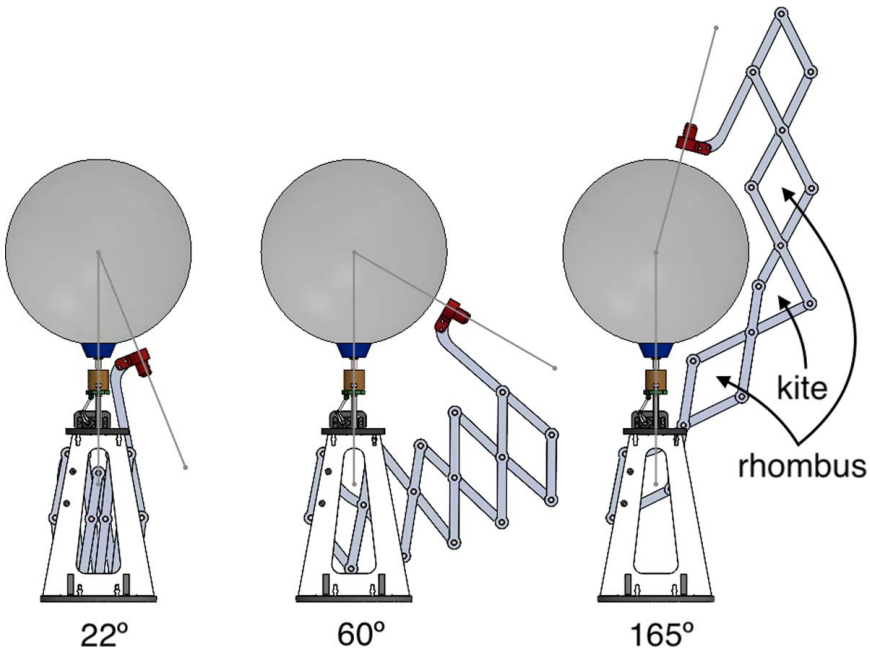

Fig. 4 The VTT joint allows members to move freely over the surface of a virtual sphere. Minimum, intermediate, and maximum joint positions are shown with a virtual member axis to show minimal and maximal angles.

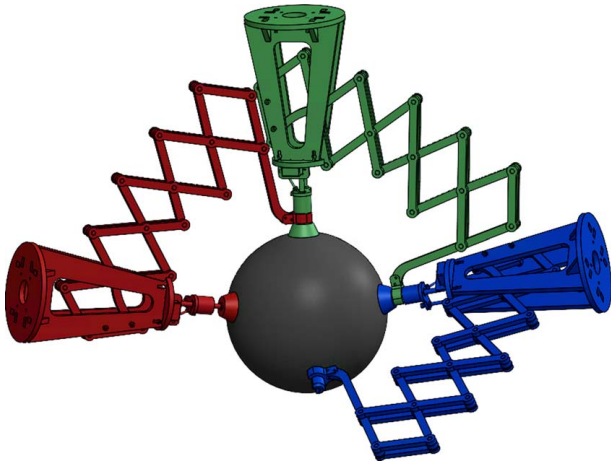

Fig. 5 Chaining of three VTT joints. The truss members are constrained to move over a virtual sphere about the virtual center. If this node needs to interact with the environment, a real sphere can be added. The rightmost VTT joint has a free pivot catch which can chain to another VTT joint.

catch help align the linkage plane with the axis of the post, increasing the area of acceptance. Next, the linkage extends until the catch is fully aligned with the post, so precise positioning is not required on the part of the linkage actuator. Finally, a sleeve slides upwards and captures the catch. As a result, the catch is free to pivot about the central post, but the post axis is held in the correct position for the truss constraints. This process is shown in Fig. 6.

Two nodes can be brought together and merged by docking their joint chains, and a single node can split into two by undocking in the middle of the joint chain. The order of the members in the chain can be rearranged by docking the ends of the chain to form a closed loop, then undocking at a different point in the loop.

When all of the VTT joints at a node are docked into a loop, the joints are completely passive. They constrain the truss members by virtue of their geometry, and all of the degrees-of-freedom in the VTT are controlled by actuating the lengths of the members. However, in order to autonomously perform the docking process, lightweight actuators are needed to control the position of the free linkage of an open chain. Before a chain is undocked, the actuators would connect to the linkage using a clutch mechanism to control the motion of the linkage. Once docking is complete, these actuators disconnect to return the joint to its typical passive state. We omit these actuators for simplicity in our current prototype and move the free linkages by hand as a proof-of-concept.

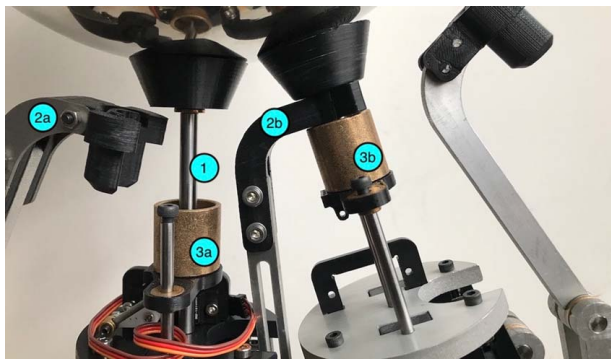

Fig. 6 Three VTT joints demonstrate the docking procedure. A VTT joint out of frame to the left is preparing to dock with the central VTT joint, which itself is already docked to the VTT joint on the right. A pivot catch (2a) on the end of a linkage (out of frame to the left) approaches the alignment post (1). The sleeve (3a) is in the down position, in preparation for latching. The second pivot catch (2b) in already docked in place by the sleeve (3b); it can pivot about its post but is otherwise constrained.

The other distinguishing feature of the VTT joint is the number of bars in the linkage. The main limitation of the six-bar linkage in the CMS joint is the relationship between the maximum angle and the size of the physical sphere that could fit at the virtual center. The sphere diameter needs to be at least as large as the actuator diameter to be a useful environmental contact, and with a six-bar linkage, the central pin joint collides with the sphere at moderate angles. Our novel 12-bar linkage has the same spherical motion and offset properties as the six-bar linkage, but the additional links allow a larger range of motion over the sphere surface with shorter link lengths. Figure 4 shows the VTT joint in its fully extended collision-free position at 165 deg. If there is no physical sphere, the joint can extend further, to about 175 deg. In fact, any even number of additional links can be used to extend the six-bar mechanism with a repeated series of rhombuses, in a similar way to a pantograph or scissor mechanism. If the total number of links is a multiple of four, then it is possible to replace the central rhombus section with a kite-shaped section while still preserving the spherical motion property. This kite-shaped section can be most clearly seen in Fig. 4, and it greatly improves the clearance over the sphere.

3.3 Reconfiguration Demonstration. We have built a simple VTT, shown in Fig. 7, to demonstrate this reconfiguration capability. This truss has 18 members, 12 of which form an outer octahedron. The remaining six members form two internal tetrahedra. The internal nodes associated with these six members can merge and split according to the process described in Sec. 3.2.

To reduce the manufacturing burden, some of the external members use a fixed-length aluminum extrusion instead of the typical Spiral Zipper. The internal nodes use the reconfigurable VTT joints, while the external nodes use non-reconfigurable CMS joints for simplicity. Since these external nodes need to contact the environment and do not reconfigure, we place a physical sphere at each of their centers. To keep the physical spheres in place, we pin them to one arbitrary member at each node.

4 Robot Topological Analysis

A VTT has the capability to change from one truss topology to another by merging or splitting nodes. Yet, there are limitations on which topologies can be reached by a given VTT. For example, trusses shown in Figs. 13(*a*) and 13(*b*) have the same number of nodes and members, but there is no way to reconfigure from one into the other. This section discusses the rules that govern possible reconfiguration actions.

4.1 Robot Configuration and Topology. We give the term *configuration* the standard definition in robotics: a set of parameters that describes the complete specification of all of the points in a robot system. A VTT consists of some fixed number of member modules, so the configuration must encode both the spatial locations of all of the members as well as a description of how the members are connected to one another. We call the description of the connections between the members the *connectivity* of the truss. The natural choice to describe the connectivity is a simple graph with labeled edges, with each graph edge corresponding to a truss member. Then, to fully specify the physical locations of the members, one choice is to provide an *embedding* of this graph, which is a specification of the point in 3D space of each node in the graph. The member lengths can be unambiguously determined from the node locations, while the reverse is not true. Using a graph embedding to represent the full configuration space greatly simplifies the kinematics.

When performing high-level planning and design, we often would like to disregard the labels from the labeled graph representation of

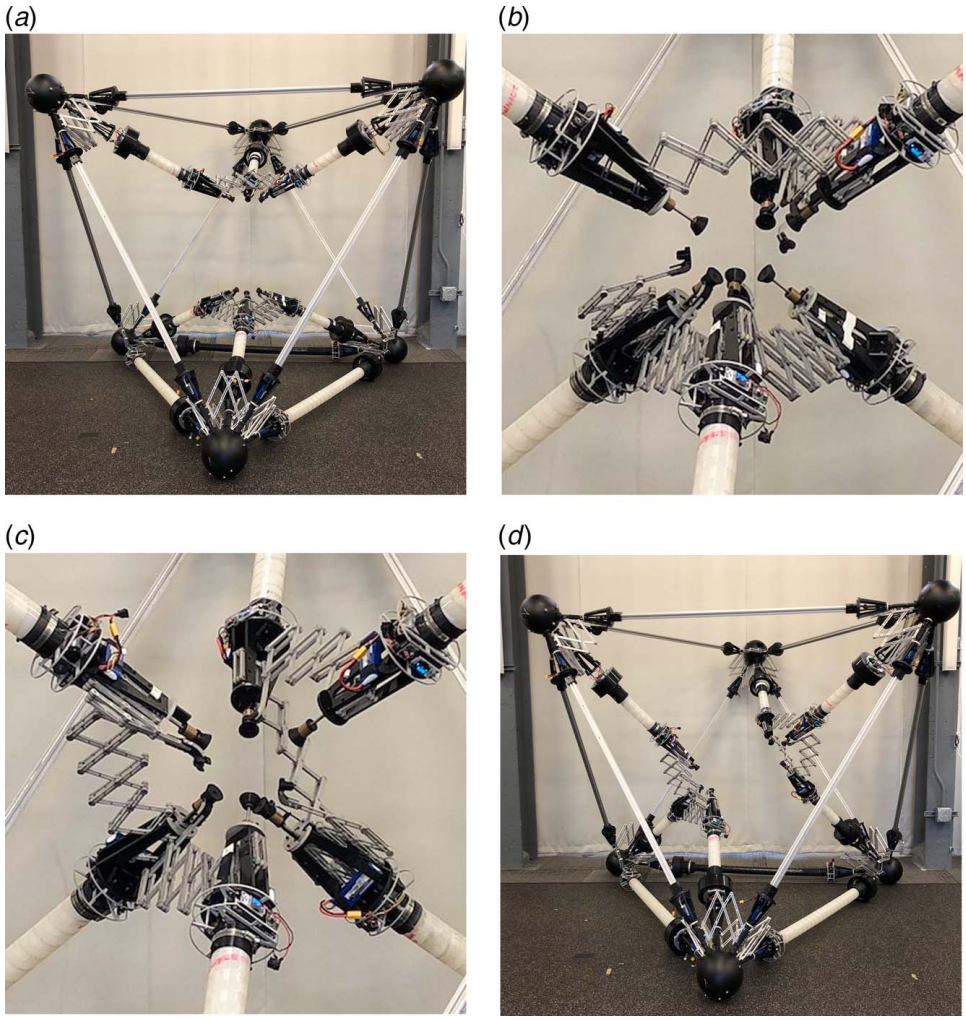

Fig. 7 A demonstration of merge and split reconfiguration. (*a*) We start with an octahedron with two internal nodes. (*b*) The internal nodes move to have a coincident virtual center and undock their chains. (*c*) The joints rearrange their connections to form two different nodes. (*d*) The two new nodes move apart to show that the truss connectivity has changed.

the connectivity. This yields the *topology* of the truss, which is an isomorphism class of truss connectivity. The truss topology can be represented by an unlabeled graph. In other words, two trusses are said to have the same topology if there is a *graph isomorphism* between their graph representations. The truss topology provides the most general specification that corresponds to the tasks a particular robot configuration is suited to perform.

We call the motion that only moves the position of the nodes a *geometric reconfiguration*, and a motion that merges or splits nodes a *topological reconfiguration*. Figure 8 shows a simple 2D example. Any single topological reconfiguration action necessarily changes the number of nodes in the graph representation, but the total number of edges must remain the same. The edges correspond to physical actuators; they cannot vanish or reappear, nor can two actuators occupy the same location. Consequently, nodes that are first-order neighbors—nodes directly connected by an edge—cannot merge. If first-order neighbors in a graph were to combine, the edge between them would be deleted. Similarly, nodes that are second-order neighbors—the neighbor of a neighbor—also cannot merge. If second-order neighbors were to combine, two edges would collapse to a single edge. Therefore, the valid list of node merges for a given connectivity is restricted to the list of third or greater order neighbors.

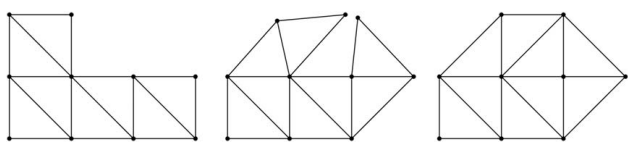

Fig. 8 This 2D truss demonstrates geometric reconfiguration as it moves from the first configuration to the second, and then topological reconfiguration as it merges two nodes to reach the third configuration. Notice that even in 2D, nodes must be third- or higher-order neighbors to merge.

4.2 Rigidity. A VTT configuration can be fully represented by a graph $G(V, E)$ and an embedding function $P(V) \mapsto \mathbb{R}^3$ where V is the set of truss nodes and E is the set of members in the truss. We use $n = |V|$ as the total number of truss nodes and $m = |E|$ as the total number of members in the truss. The embedding function P specifies the location of each truss node in 3D space. This combination of graph and embedding (G, P) is typically called a *framework* [22].

A VTT is required to be a rigid structure to maintain its shape. This means at all times, the truss must be a *rigid framework*. In order to maintain controllability of every node on a VTT robot, it is necessary for the truss to satisfy the stronger condition of *infinitesimal rigidity*. Infinitesimal rigidity is the property that no infinitesimal motions can be assigned to the nodes of the truss without violating the distance constraints imposed by the member lengths,

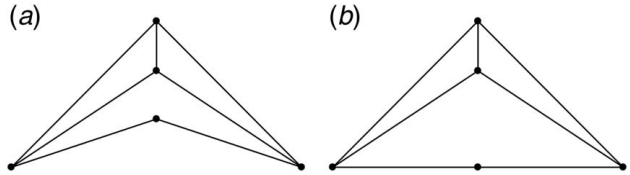

Fig. 9 An example of a generically rigid graph in 2D. Configuration (*a*) is infinitesimally rigid, and all nearby configurations are also infinitesimally rigid. Configuration (*b*) is a singular configuration which is not infinitesimally rigid. Although no nodes can move a finite distance without violating distance constraints, an infinitesimal motion in the vertical direction can be applied to the bottom center node.

aside from the motions that correspond to the six degrees-of-freedom of a rigid body [22]. However infinitesimal rigidity is a property that depends on the specific embedding of a framework. A more general property is *generic rigidity*, a property of the graph G alone (or equivalently, a property of the topology alone). A generically rigid graph is rigid "almost everywhere," and if it is rigid with one embedding, it is rigid in some neighborhood of that embedding [22]. Similarly, a graph that is generically non-rigid will be flexible "almost everywhere." There may be embeddings where a generically rigid graph becomes infinitesimally flexible, and these correspond to singularity configurations of a VTT. Figure 9 gives one such example in 2D space.

In 2D, there are combinatorial methods for determining whether a graph is generically rigid. Finding a combinatorial method for 3D graphs is still an open problem in rigidity theory. However, checking the infinitesimal rigidity of a 3D framework is straightforward. The infinitesimal motions of a framework are captured by the *rigidity matrix* [23]. Given a framework (G, P), we have the following m constraints on the node positions:

$$\|P(u) - P(v)\| = l_{(u,v)}, \quad \forall (u, v) \in E \tag{1}$$

where $l_{(u,v)}$ is the length of the member connecting nodes u and v. The rigidity matrix is the $m \times 3n$ Jacobian matrix of these constraints with the partial derivatives taken with respect to the node positions. The kernel of the rigidity matrix corresponds to the infinitesimal node velocities which preserve the length constraints. In 3D space, the rigidity matrix of a rigid framework will have a six-dimensional null space corresponding to the six degrees-of-freedom of a rigid body, whereas a non-rigid framework will have a larger null space corresponding to the additional degrees-of-freedom. To determine whether a given framework is rigid, we simply construct this matrix and calculate its rank. If the rank is $3n-6$, then the framework is infinitesimally rigid in the given embedding.

To achieve a rank of $3n-6$, the rigidity matrix must have at least $3n-6$ rows. This implies the following necessary condition on the number of members in a rigid framework:

$$m \geq 3n - 6 \tag{2}$$

Since this condition is a necessary condition for infinitesimal rigidity that does not rely on the embedding, it is also a necessary condition for generic rigidity.

We use the fact that if a graph has at least one infinitesimally rigid configuration, then the graph is generically rigid. A random embedding is assigned to a graph, and infinitesimal rigidity is checked by computing the rank of the rigidity matrix. If the graph is not infinitesimally rigid, then the embedding is perturbed to ensure its position is not singular. Since generic (non-singular) configurations occur "almost everywhere," this method works well in practice.

Truss topologies that are not generically rigid have no infinitesimally rigid configurations, so these topologies must be avoided during topological reconfiguration. Any merging or splitting of nodes must preserve the generic rigidity of structure. Additionally, singular configurations of generically rigid topologies must be avoided during geometric reconfiguration.

Minimally rigid structures are frameworks that have precisely the number of members to guarantee rigidity; the removal of any member results in flexibility. The above constraint inequality becomes an equality, and in this case, the number of members must be a multiple of three. These structures are statically determinate and as such are much easier to control than overconstrained structures.

Every three members we add to a minimally rigid truss allows us to add one extra node. As we will see in Sec. 4.5, the complexity of a graph is heavily dependent on the number of nodes, and the number of possible truss topologies will grow rapidly for every three members added.

4.3 Possible Reconfiguration Actions. Given a truss topology, we can enumerate all of the possible reconfiguration actions that preserve the generic rigidity of the structure. There are two basic reconfiguration actions: merging two nodes, and splitting a node. In this section, we will describe how to enumerate all possible ways of performing a merge or a split.

To enumerate the possible ways to merge two nodes, we need to generate all possible pairs of nodes on the truss and determine which are third- or greater order neighbors. First-order neighbors can be determined from the adjacency matrix, which can be built in $O(n^2)$ time, where n is the number of nodes. The second-order neighbors can be determined from nonzero entries of the square of the adjacency matrix, which takes $O(n^3)$ time via matrix multiplication. Any remaining pairs of nodes will be third-order or greater, and there may be $O(n^2)$ such pairs in general. We need to check that the result of merging each of them is rigid. Updating the graph data structure to merge two nodes takes $O(n)$ time, which is negligible compared to checking rigidity. We check rigidity by calculating the rank of the rigidity matrix, which takes $O(mn)$ time. Checking rigidity for each candidate pair of nodes is the most computationally complex step, so the whole procedure is $O(mn^3)$.

To enumerate the possible ways to split a node in a truss, we need to enumerate all of the ways of picking a set of members from each node. Since it is a necessary condition that a rigid graph has minimum node degree three, we only need to be concerned about splitting nodes that have six or more members, and we only need to pick sets of minimum size three. Additionally, we can ignore any trusses that are minimally rigid. In an n-node truss, any given node may have a maximum degree $d = n-1$. Since choosing a set of members is identical to choosing its complement, we can designate one of the members to always be in the set. There are at most $\binom{d-1}{3} + \binom{d-1}{4} + \cdots + \binom{d-1}{d-3}$ choices, which is $O(2^n)$. Once again, checking rigidity takes $O(mn)$ time, so the total time is $O(mn2^n)$. Although this term is exponential, for the trusses investigated in this paper n is small enough that this procedure is fast in practice.

This process enumerates all of the possible reconfiguration actions, however, we are often only interested in the non-isomorphic results. Since we have a large number of graphs, pairwise isomorphism testing is inefficient. Instead, for each resulting graph, we compute a canonical representation of the graph. Isomorphic graphs will have identical canonical representations, so it serves as an identifier that can be directly compared. Although computing a canonical representation is very difficult in general, the well-known `nauty` program is very efficient for small graphs (<100 nodes) [24].

In Sec. 4.5, we enumerate a list of all reconfigurable truss topologies with up to 29 members. From this list, we can summarize their reconfiguration capabilities with the statistics given in Table 1. The table gives the maximum and average number of merges and splits for m-member reconfigurable truss topologies. For example, among all 24-member trusses, the maximum amount of merges that any given truss topology that can perform is six. The average given is the median among all trusses that can perform such an action. For example, among all 24-member truss topologies that can perform

Table 1 Number of reconfiguration actions for truss sizes

m	Max(Avg) # merges	Max(Avg) # Non-iso merges	Max(Avg) # splits	Max(Avg) # Non-iso splits
18	1 (1)	1 (1)	40 (25)	5 (3)
19	1 (1)	1 (1)	40 (35)	3 (3)
20	1 (1)	1 (1)	50 (50)	2 (2)
21	3 (1)	3 (1)	90 (40)	54 (13)
22	3 (1)	2 (1)	140 (62)	65 (15)
23	2 (1)	2 (1)	140 (90)	37 (15)
24	6 (1)	5 (1)	210 (60)	146 (40)
25	5 (1)	5 (1)	262 (90)	171 (56)
26	5 (1)	4 (1)	364 (125)	206 (66)
27	10 (1)	9 (1)	420 (88)	308 (75)
28	9 (1)	8 (1)	570 (125)	389 (110)
29	9 (1)	7 (1)	641 (166)	426 (140)

a merge, in the median case, there is only one way to perform a merge. This table shows both the total number of ways to apply a reconfiguration action, which results in many different truss connectivities, and how many of these actions result in distinct (non-isomorphic) truss topologies. Using 24-member trusses as an example again: although there are trusses with six different ways of picking pairs of nodes to merge, these choices never result in more than five non-isomorphic truss topologies. This table can be thought of as the true "worst cases" and "average cases" of the theoretical bounds given above.

We can see that for most small trusses, if a truss topology can perform a merge, there is typically only one way to merge two nodes. On the other hand, if a truss topology can perform a split, there are typically many distinct ways to split a node. In Sec. 5, we will see the implications of this on the space of reachable topologies for any given VTT.

4.4 Smallest Reconfigurable Trusses. From the restrictions on reconfiguration, we can determine the minimal reconfigurable trusses.

THEOREM 1. *A minimum of 18 members are necessary to admit reconfiguration in a VTT.*

Proof. Since topological reconfiguration steps are reversible, any reconfigurable VTT must have some configuration which is capable of merging two nodes. One necessary condition for generic rigidity in 3D is that the graph representation of the truss must have a minimum node degree of three or more (with the exception of the special cases of a single bar and single triangle). Since the two merging nodes must be no closer than third-order neighbors, each of the merging nodes must have three unshared neighbors to satisfy rigidity. Therefore, the total number of nodes in this configuration is at least eight. A second necessary condition for generic rigidity in 3D is the constraint inequality (2) derived in Sec. 4.2. With eight nodes, $m \geq 18$; therefore, the minimum number of truss members required to admit reconfiguration is 18. ∎

As we will see in Sec. 4.5 where we enumerate all reconfigurable topologies with 18 members, only trusses with seven or eight nodes are possible, since the complete graph with six nodes only contains 15 members, and a truss of 9 nodes would violate the constraint inequality (2). There are five non-isomorphic graphs with seven nodes and 18 edges [25], and they are pictured in Fig. 10. All of them are generically rigid in 3D. One of the five graphs (Fig. 10 (1)) corresponds to the complete graph with six nodes with one additional node connected by three edges. This graph is incapable of splitting any node without losing generic rigidity. The remaining four graphs correspond to the four 18-member VTTs that are capable of topological reconfiguration.

4.5 Enumerating Reconfigurable Graphs. Since reconfiguration is achieved by node splitting and merging, the number of members is preserved throughout topological reconfiguration of a VTT. Given a prescribed number of members with which to construct a VTT, what choices exist for putting them together such that they can reconfigure? For relatively small trusses, it is feasible to enumerate all possible truss topologies that can reconfigure in some way. In this paper, we consider trusses with up to 29 members in detail.

Non-isomorphic connected graphs with a specified number of nodes can be generated efficiently with programs such as `geng` [24]. From the list of all non-isomorphic connected graphs up to certain size, we can check each graph for rigidity and check whether it has any reconfiguration actions available. The results can be found in Table 2. Since the number of members is preserved throughout reconfiguration actions, the table is organized by number of members. Recall, for an n-node truss, the minimum number of members is $3n-6$ to satisfy the rigidity condition, and the maximum is $n(n-1)/2$ for the complete graph. In the table, combinations of m and n that result in minimally rigid trusses are marked with an asterisk.

For 27- to 29-member trusses, there are too many 11-node graphs to check all of them for rigidity in a reasonable amount of computer time. In these cases, we start with the much smaller number of reconfigurable ten-node graphs and enumerate all of the non-isomorphic ways to perform a node split. This gives the correct result for the number of reconfigurable 11-node graphs, but does not reveal how many non-reconfigurable connected graphs are rigid or flexible.

We can see some interesting results in Table 2. The total number of connected simple graphs increases roughly exponentially in the number of nodes. Similarly, the number of rigid graphs and reconfigurable graphs grow exponentially in the number of nodes. This explains the large jump in the number of reconfigurable graphs every three members, since this results in an additional possible node. As the trusses get larger, the proportion of rigid trusses that are reconfigurable increases. Additionally, trusses that are overconstrained by three or more members are highly likely to be reconfigurable. This trend can be seen in Fig. 11, which combines the exhaustive data in Table 2 with additional sampled data for

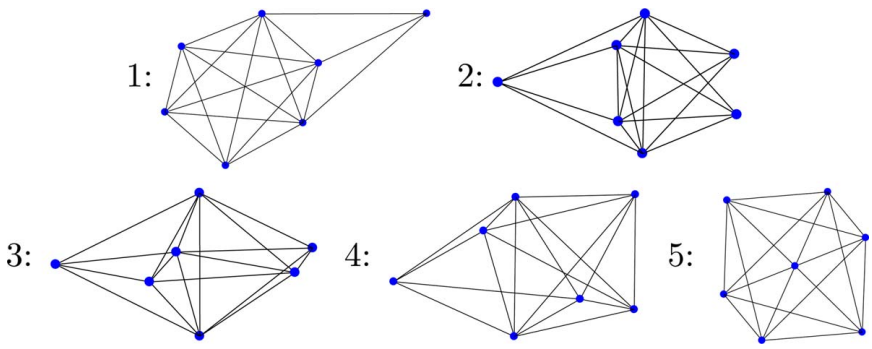

Fig. 10 The five topologies with seven nodes and 18 members

Table 2 Number of non-isomorphic graphs for truss sizes

m	n	Connected	Rigid	Reconfig.
18	7	5	5	4
	8[a]	658	374	14
19	7	2	2	2
	8	400	297	6
20	7	1	1	1
	8	220	185	2
21	7	1	1	1
	8	114	103	97
	9[a]	21,817	11,487	1493
22	8	56	52	52
	9	15,558	10,790	902
23	8	24	23	23
	9	10,096	8041	405
24	8	11	11	11
	9	5984	5177	5095
	10[a]	1,245,369	612,884	199,285
25	8	5	5	5
	9	3247	2954	2954
	10	1,057,896	685,028	163,761
26	8	2	2	2
	9	1635	1534	1534
	10	827,086	622,816	103,926
27	8	1	1	1
	9	770	737	737
	10	595,418	493,332	491,310
	11[a]	105,723,785	?	28,164,762
28	8	1	1	1
	9	344	333	333
	10	394,820	348,250	348,250
	11	105,925,685	?	30,547,662
29	9	148	144	144
	10	241,428	221,954	221,954
	11	98,945,882	?	26,502,523

[a]minimally rigid truss.

trusses that are too large to exhaustively enumerate. Thus far, sampling results indicate that all rigid trusses overconstrained by four or more members are reconfigurable, and all rigid trusses with at least 48 members (or equivalently, at least 18 nodes) are reconfigurable. This is promising for the practical application of VTT systems. If we have a way to generate desired goal topologies, then for larger systems, they will always or almost always be reconfigurable.

5 Topology Networks

To describe the reconfiguration capability of a VTT, we introduce the *topology network* of a VTT. The topology network can be thought of as a "graph describing the relationships of graphs." In the topology network, each vertex corresponds to a truss topology. This correspondence is one-to-one; a network vertex can be uniquely labeled by the canonical graph representation of the truss topology. Two vertices in the network are connected by an edge if a single topological reconfiguration action can take a VTT configuration in the first topology to the second topology. Figure 12 shows an example network of three truss topologies, which captures the reconfiguration capability of the VTT described in Sec. 3.3.

To keep the terminology of a network distinct from that of an individual truss, we refer to the elements of a *topology network* as *vertices* and *edges*. In contrast, the topology of a truss is a *graph* describing an arrangement of *nodes* and *members*. We avoid the use of the term "graph" to describe the topology network to reduce ambiguity.

The topology network for a VTT can be generated by starting with some initial truss topology, computing its canonical form, and adding it as the first unexplored vertex in the topology network. For each unexplored vertex, we can discover the adjacent vertices by applying all valid topological reconfiguration actions as described in Sec. 4.3. After applying a valid topological reconfiguration action to the current truss, the resulting truss connectivity is put into its canonical form to check if it is isomorphic to any of the existing truss topologies in the topology network. If it is a new truss topology, then it is added as a new unexplored vertex to the topology network. An edge is then added between the current vertex and the vertex corresponding to the new truss topology. This process is repeated for any new unexplored vertices until the graph is fully explored.

Figure 13 shows two more complicated examples of VTTs and their associated topology networks. We can see that these networks feature dense clusters of vertices with a small number of paths connecting the clusters. This corresponds to the results we discussed in Sec. 4.3. Recall, trusses that are overconstrained enough to split a node can typically do so in many different ways to arrive at many different topologies. This also matches our expectations from Sec. 4.5; splitting a node increases the number of nodes in a truss, and in general, the number of unique truss topologies increases exponentially with the number of nodes.

However, any truss topologies that have the maximum number of nodes for their member count can only reconfigure by merging two nodes. As we saw in Sec. 4.3, for moderate numbers of members, most of these truss topologies only have one non-isomorphic way to merge two nodes. This implies that if we arrived at this truss topology by splitting a truss node, the only thing to do is merge that truss node right back together again. These truss topologies are peripheral vertices in the topology networks, forming the clusters around the more overconstrained trusses, which have fewer nodes. It is the smaller number of truss topologies that have multiple ways to merge truss nodes that form the "bridge" vertices in the topology networks. When designing search algorithms for VTT topological reconfiguration, it may be beneficial to prune these peripheral vertices, since they won't be part of any shortest path unless they correspond to the start or goal topology. As the total number of members (and hence nodes) increases, we expect the average truss to have more ways of merging those nodes. For large trusses, their topology networks will likely have a larger proportion of bridge nodes to fewer peripheral nodes.

In Fig. 13, these particular topology networks only contain topologies with 8 or 9 truss nodes, with the eight-node trusses forming the centers of clusters of nine-node trusses. For VTTs with more members, there may be more hierarchical levels of clustering. For example, the topology network of the 24-member VTT in Fig. 15 contains truss topologies with 8, 9, or 10 truss nodes. In that case, the eight-node trusses form the center of clusters of nine-node trusses, which in turn form centers of clusters of ten-node trusses.

Any given topology network can also be thought of as a single connected component in a *universal topology network*, an infinite graph that describes the relationship between all truss topologies. If we start the above procedure with all of the enumerated reconfigurable graphs with m-members (generated using the procedure from Sec. 4.5), we can generate the complete topology network for all m-member trusses. This procedure only takes a single pass through the list of starting trusses to build the topology network, because the result of any reconfiguration action is guaranteed to already be in the starting set. We could also consider all rigid trusses to be part of the starting set, in which case any rigid but non-reconfigurable trusses would form isolated vertices in the topology network.

Given start and goal truss configurations, the topology network can be used to quickly eliminate unreachable goal configurations. If the topology of the goal configuration does not lie in the same connected component as the topology of the start configuration, then there is no path through configuration space that connects

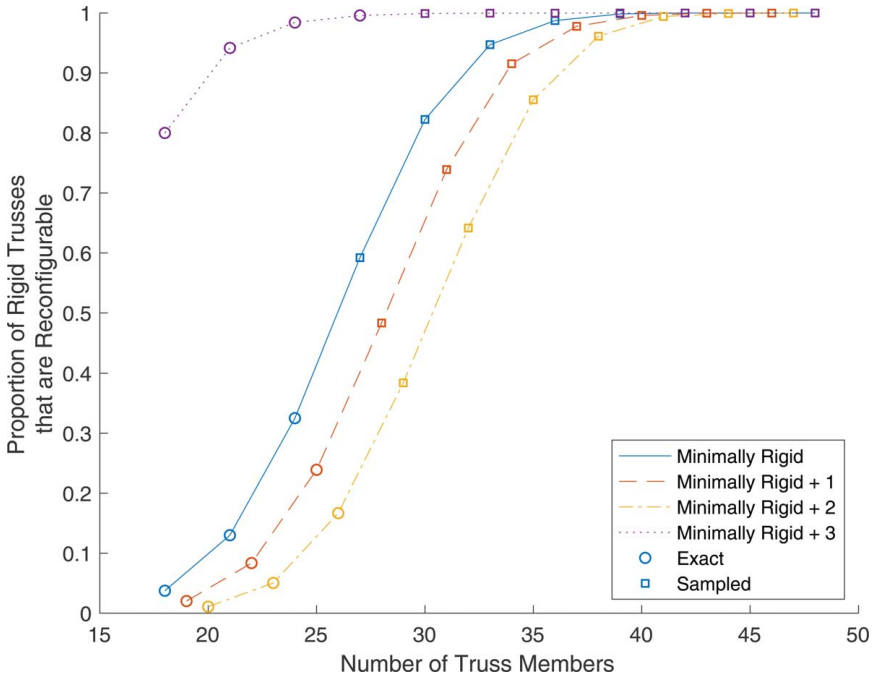

Fig. 11 The proportion of reconfigurable trusses increases with the number of members

the start and goal configurations. However, these graphs do not consider self-collisions or actuator constraints, so the existence of a path through the topology network does not guarantee that a path through configuration space exists. A more sophisticated planning procedure would be required to find a path that satisfies all of the physical constraints.

In prior work, we have proposed several constraint-aware methods for local VTT reconfiguration planning [7–10]. These methods are effective for finding shorter paths with a few topological reconfiguration actions between the start and the goal. However, these methods are not aware of the structure of the topology network. By combining the local planners with some guidance from the topology network, it may be possible to achieve effective path planning for more complicated reconfiguration sequences.

5.1 Topology Networks of 18–20 Member Trusses. As a practical matter, it is easier to manufacture VTTs with smaller numbers of members, so it is useful to explore the smaller number of member systems exhaustively. Recall the five non-isomorphic graphs with seven nodes and 18 edges from Fig. 10, which include the four smallest reconfigurable trusses. Figure 14 shows the topology networks of the five trusses. They form five separate connected components, which implies none of these trusses can reconfigure into any of the other four. The 14 additional vertices (6–19) in the network represent the topologically unique trusses that can be obtained by splitting nodes from the 18 member and seven-node trusses. This is also reflected in our enumeration results from Table 2. There are 658 non-isomorphic connected graphs with eight nodes and 18 members. Of those, 284 are non-rigid, 360 are rigid but incapable of reconfiguration, and the 14 shown here are reconfigurable.

A similar process can be applied for 19- and 20-member structures. There are two 19-member topologies with seven nodes, each of which can reach three eight-node topologies. There is only one 20-member topology with seven nodes, and it can reach two eight-node topologies.

5.2 Topology Networks of Larger Trusses. With 21 members, rigid nine-node trusses become possible, and the connected components of the topology networks begin to exhibit

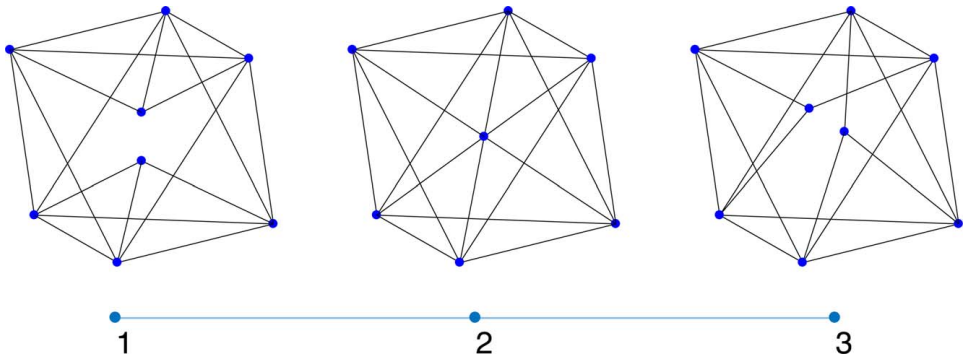

Fig. 12 An example of a topology network for an 18-member VTT. The (1) left truss, an octahedron with two interior tetrahedral cells, can merge its nodes to form the (2) center truss. (3) This truss can split the center node in a topologically distinct way, shown on the right. This is represented in the topology network: truss topologies 1 and 3 can reach truss topology 2 in a single action, but cannot directly reconfigure into each other.

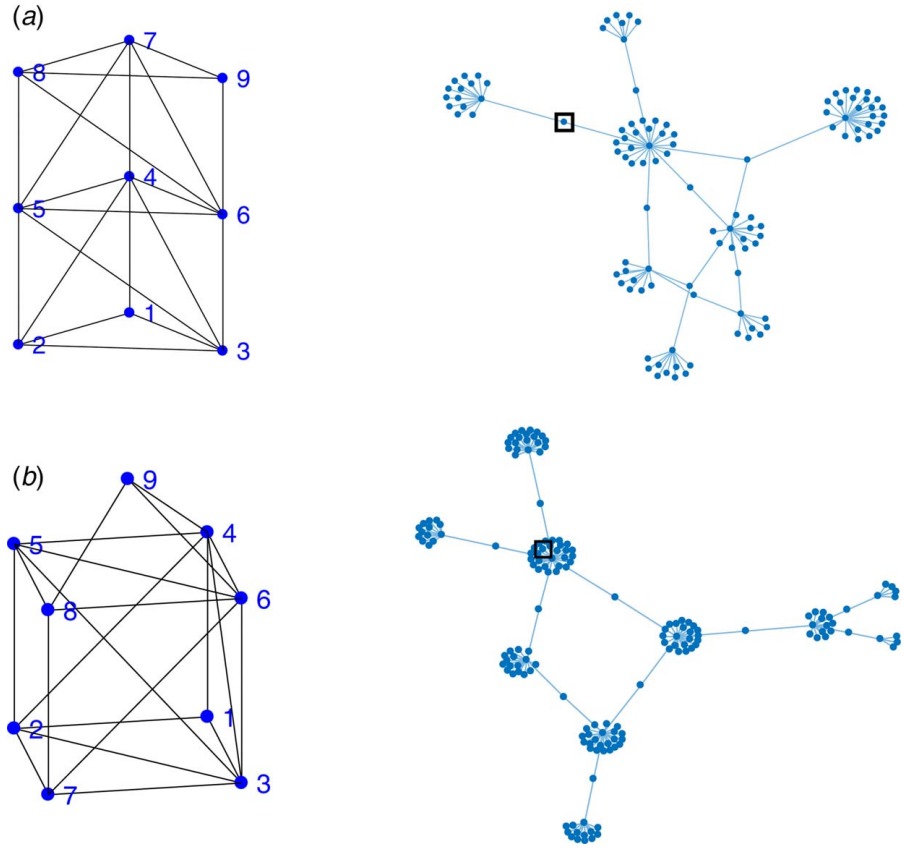

Fig. 13 Two 21-member VTTs and their associated topology networks. The black box shows the location of the pictured topology in the topology network. The topology network of truss (*a*) has 111 topologies, while the topology network of truss (*b*) has 165 topologies.

more interesting organization. Figure 13 shows two such examples. There is only one 21-member truss topology with seven nodes, which corresponds to K_7—the complete graph on seven nodes. It can reconfigure by splitting into an eight-node topology, but this eight-node topology cannot reconfigure further. Therefore, all other reconfigurable trusses with 21, 22, or 23 members involve merges and splits between eight and nine-node topologies.

Rigid ten-node truss topologies become possible with 24 members, and again, there is a jump in complexity of the topology networks. The maximum size of the topology networks increases rapidly with the maximum number of nodes in the truss. Figure 15 gives an example of a VTT with 24 members and ten nodes. Here, the reachable connected component has 38,217 topologies.

We have applied this same procedure to generate the topology network for all VTTs up to 29 members. Beyond this becomes expensive to compute. There are several interesting properties that emerge, summarized in Table 3.

First is the number of connected components in the topology network, and the size of the largest one. We can see that the number of connected components is correlated with the total number of reconfigurable trusses, but it does not increase as quickly. For larger trusses, these connected components are very non-uniform in size. There are many small components with only a handful of topologies, but the vast majority of truss topologies belong to a small number of connected components. Additionally, it seems that these largest connected components tend to coalesce into a single component as the number of members increases. For example, for 24-, 25-, and 26-member trusses, about half of the truss topologies are in the largest connected component. For 27-, 28-, and 29-member trusses, over 99.9% of reconfigurable truss topologies are in the largest connected component. This is promising for planning, since it is likely that two randomly selected reconfigurable trusses with the same number of members have a connecting path. Table 3 lists the probability of this occurring, up to 29 members.

We might imagine that at some point, the number of connected components in the topology network will begin to decrease, and perhaps all sufficiently large VTTs are in the same component—provided they have the same number of members. Since the number of truss topologies increases exponentially, it is unlikely we will ever arrive at this answer by exhaustively computing the topology networks.

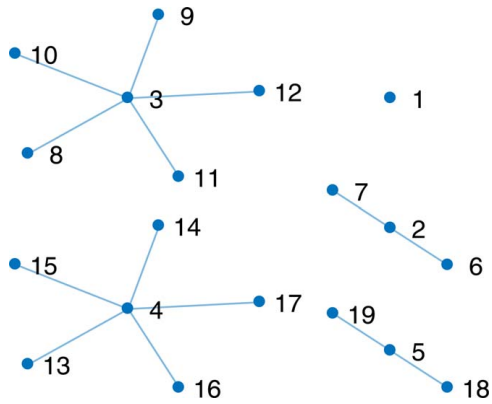

Fig. 14 The topology networks of the five topologies with seven nodes and 18 edges, where numbers 1–5 above correspond to numbers 1–5 in Fig. 10. They form five connected components.

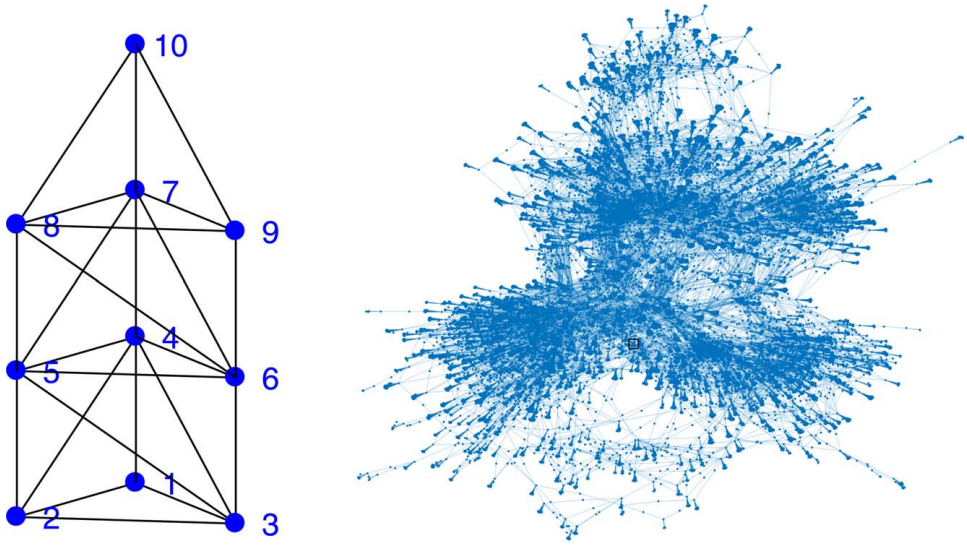

Fig. 15 A 24-member VTT and its associated topology network. It belongs to the third largest connected component for 24-member trusses, which contains 38,217 topologies.

Table 3 Connected component (CC) grouping statistics

m	Reconf. trusses	Num CCs	Size Max CC	Same CC chance	Graph diam.
18	18	4	6	0.235	2
19	8	2	4	0.429	2
20	3	1	3	1	2
21	1591	47	258	0.0691	10
22	954	32	145	0.0718	10
23	428	18	75	0.0894	6
24	204,391	83	96,288	0.341	22
25	166,720	16	69,871	0.333	20
26	105,462	12	44,131	0.326	20
27	28,656,810	206	28,654,819	0.99986	30–34
28	30,896,246	12	30,896,166	0.999995	31–32
29	26,724,621	2	26,724,611	0.9999993	30–32

Another interesting property is the graph diameter of the connected components in the topology network, also listed in Table 3. This is the maximum length of the shortest path between every pair of vertices in the network. It can equivalently be considered as the maximum eccentricity, where the eccentricity of a vertex is the maximum length of the shortest path to every other vertex in the network. For 18- to 26-member trusses, the diameter was determined exactly by computing the eccentricity of every vertex. For 27- to 29-member trusses, we can only compute the eccentricity of a random sample of vertices. The diameter is bounded below by the maximum eccentricity of the sampled vertices and bounded above by twice the minimum eccentricity of the sampled vertices, so we provide upper and lower bounds for the true diameter. We can see that this increases very slowly with the size of the trusses. For example, although there are over 200,000 reconfigurable 24-member truss topologies, the worst-case diameter of any 24-member connected component in the topology network is 22. This means if two topologies are in the same connected component, then you can reconfigure from one to the other with only 22 reconfiguration actions. If we can find an effective search heuristic, then it may be possible to efficiently find paths between very large truss configurations, despite the large and highly connected search space.

Finally, we can return to the tower and dome example from Fig. 2. Both of these topologies have 27 members, and both are rigid and reconfigurable. Given the statistics from Table 3, it is no surprise to find that they are in the same connected component. A simple breadth-first search in the topology network starting at the

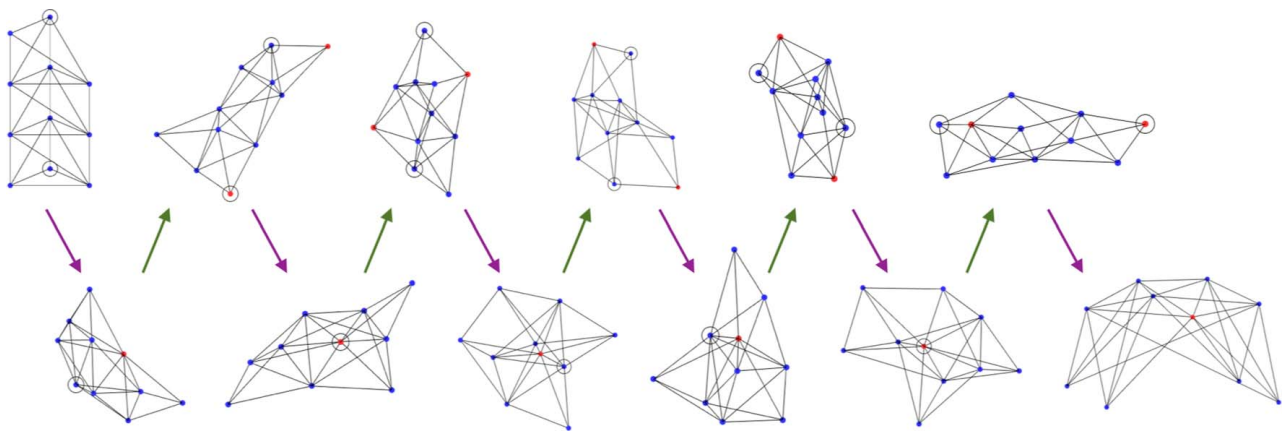

Fig. 16 A sequence of topologies to reconfigure from the tower configuration to the dome configuration from Fig. 2. The top row is all 11-node trusses, and the bottom row is all ten-node trusses, so the downward pointing arrows correspond to merging two nodes, while the upward pointing arrows correspond to splitting two nodes. The circled nodes are nodes which are about to merge or split, and the red (or lightly shaded) nodes are the resulting nodes from the previous merge or split.

tower topology gives a shortest path to the dome topology, which passes through ten intermediate topologies. This sequence is shown in full in Fig. 16. The embeddings of the intermediate steps were generated using a force-directed graph embedding algorithm, so the geometry of a real collision-free path may look very different.

6 Conclusion

Variable topology trusses are a novel extension to the class of robots known as variable geometry trusses. The ability to topologically reconfigure expands the capabilities of the robot system. A hardware design that implements the motion and reconfiguration of a VTT was presented. Constraints on reconfiguration and the allowable truss topologies were introduced, leading to the derivation that 18 members are required for topological reconfiguration and to a procedure for enumerating reconfigurable trusses. By exhaustively exploring reconfiguration options, the relationship between all of the topologically distinct reconfigurable structures for 18 up to 29 members were characterized. *Topology networks* were introduced as a way to visualize the potential for different configurations to reconfigure to each other and the steps to achieve this reconfiguration.

Future work includes performing a comprehensive study of structures with more members and identifying promising configurations for common robotics tasks. The motion and reconfiguration planning problems for VTTs also need to be further developed. These problems might be tackled by combining our prior work in local planning with the global methods described in this work. We suspect that topological reconfiguration can be used to move past singular configurations and avoid obstacles, expanding the workspace of VTTs.

Acknowledgment

This work was supported by the Air Force Office of Scientific Research (AFOSR) contract FA2386-17-1-4656 and the Korean Ministry of Trade, Industry, and Energy (MOTIE). The authors would like to thank our collaborators Frank Park, Jongwon Kim, Sumin Park, and Eugene Park at the Seoul National University and TaeWon Seo at Hanyang University, as well as previous Univ. of Penn. GRASP Lab members Thulani Tsabedze and Daniel Edgar.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Ananthanarayanan, A., Azadi, M., and Kim, S., 2012, "Towards a Bio-Inspired Leg Design for High-Speed Running," Bioinspir. Biomimet., **7**(4), p. 046005.
[2] Reinholtz, V. A., and Watson, L. T., 1990, "Enumeration and Analysis of Variable Geometry Truss Manipulators".
[3] Patel, Y., and George, P., 2012, "Parallel Manipulators Applicationsa Survey," Modern Mech. Eng., **2**(03), p. 57.
[4] Miura, K., Furuya, H., and Suzuki, K., 1985, "Variable Geometry Truss and Its Application to Deployable Truss and Space Crane Arm," Acta Astronautica, **12**(7–8), pp. 599–607.
[5] Yim, M., Duff, D. G., and Roufas, K. D., 2000, "Polybot: A Modular Reconfigurable Robot," IEEE International Conference on on Robotics and Automation, Vol. 1, IEEE, pp. 514–520.
[6] Spinos, A., and Yim, M., 2017, "Towards a Variable Topology Truss for Shoring," 14th International Conference on Ubiquitous Robots and Ambient Intelligence, IEEE.
[7] Jeong, S., Kim, B., Park, S., Park, E., Spinos, A., Carroll, D., Tsabedze, T., Weng, Y., Seo, T., Yim, M., Park, F. C., and Kim, J., 2018, "Variable Topology Truss: Hardware Overview, Reconfiguration Planning and Locomotion," 2018 15th International Conference on Ubiquitous Robots (UR), IEEE, 610–615.
[8] Liu, C., and Yim, M., 2019, "Reconfiguration Motion Planning for Variable Topology Truss," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 1941–1948.
[9] Liu, C., Yu, S., and Yim, M., 2020, "A Fast Configuration Space Algorithm for Variable Topology Truss Modular Robots," IEEE International Conference on Robotics and Automation (ICRA).
[10] Liu, C., Yu, S., and Yim, M., 2020, "Motion Planning for Variable Topology Truss Modular Robot," Proceedings of Robotics: Science and Systems.
[11] Spinos, A., Carroll, D., Kientz, T., and Yim, M., 2017, "Variable Topology Truss: Design and Analysis," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 2717–2722.
[12] Rhodes, M. D., and Mikulas, Jr, M., 1985, Deployable Controllable Geometry Truss Beam. Technical Report TM-86366, NASA Langley Research Center.
[13] Hamlin, G. J., and Sanderson, A. C., 1998, *Tetrobot: A Modular Approach to Reconfigurable Parallel Robotics*, Kluwer Academic Publishers, Norwell, MA.
[14] Lee, W. H., and Sanderson, A. C., 2000, "Dynamic Rolling, Locomotion Planning, and Control of An Icosahedral Modular Robot," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 3, IEEE, pp. 2178–2183.
[15] Boutin, B., Misra, A. K., and Modi, V., 1999, "Dynamics and Control of Variable-Geometry Truss Structures," Acta Astronautica, **45**(12), pp. 717–728.
[16] Padmanabhan, B., Arun, V., and Reinholtz, C., 1992, "Closed-Form Inverse Kinematic Analysis of Variable-Geometry Truss Manipulators," ASME J. Mech. Des., **114**(3), pp. 438–443.
[17] Neville, A. B., and Sanderson, A. C., 1996, "Tetrobot Family Tree: Modular Synthesis of Kinematic Structures for Parallel Robotics," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 2, IEEE, pp. 382–389.
[18] Bolker, E., and Roth, B., 1980, "When Is a Bipartite Graph a Rigid Framework?," Pacific J. Math., **90**(1), pp. 27–44.
[19] Collins, F., and Yim, M., 2016, "Design of a Spherical Robot Arm With the Spiral Zipper Prismatic Joint," IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 2137–2143.
[20] Yim, M., U.S. Patent App 62/245, 461, Oct. 2015. Reconfigurable Structural Member and System.
[21] Hamlin, G. J., and Sanderson, A. C., 1996, "Tetrobot Modular Robotics: Prototype and Experiments," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 2, IEEE, pp. 390–395.
[22] Graver, J. E., Servatius, B., and Servatius, H., 1993, *Combinatorial Rigidity*, Vol. 2, American Mathematical Society, Providence, RI.
[23] Krick, L., Broucke, M. E., and Francis, B. A., 2009, "Stabilisation of Infinitesimally Rigid Formations of Multi-Robot Networks," Int. J. Control., **82**(3), pp. 423–439.
[24] McKay, B. D., and Piperno, A., 2014, "Practical Graph Isomorphism, II," J. Symbolic Comput., **60**, pp. 94–112.
[25] Comtet, L., 1974, *Advanced Combinatorics: The Art of Finite and Infinite Expansions*, D. Reidel, Dordrecht.