Task-Specific Design Optimization and Fabrication for Inflated-Beam Soft Robots with Growable Discrete Joints

Ioannis Exarchos[1], Karen Wang[1], Brian H. Do[2], Fabio Stroppa[3], Margaret M. Coad[4], Allison M. Okamura[2], and C. Karen Liu[1]

***Abstract*— Soft robot serial chain manipulators with the capability for growth, stiffness control, and discrete joints have the potential to approach the dexterity of traditional robot arms, while improving safety, lowering cost, and providing an increased workspace, with potential application in home environments. This paper presents an approach for design optimization of such robots to reach specified targets while minimizing the number of discrete joints and thus construction and actuation costs. We define a maximum number of allowable joints, as well as hardware constraints imposed by the materials and actuation available for soft growing robots, and we formulate and solve an optimization problem to output a planar robot design, i.e., the total number of potential joints and their locations along the robot body, which reaches all the desired targets, avoids known obstacles, and maximizes the workspace. We demonstrate a process to rapidly construct the resulting soft growing robot design. Finally, we use our algorithm to evaluate the ability of this design to reach new targets and demonstrate the algorithm's utility as a design tool to explore robot capabilities given various constraints and objectives.**

I. Introduction

Soft robots are often created with the goal of offering compliant physical interactions between a robot and its environment, which can be safer and more readily accommodate uncertainty compared to traditional rigid robots. Moreover, the materials and actuation technologies used for soft robots can be less expensive and easier to manufacture or assemble than those used for traditional rigid robots. Soft robots have the potential to be designed with both mechanical properties and geometries appropriate for various tasks and environments. However, soft robots are typically limited in their dexterity because continuum arms lack the localized, large angle changes of traditional joints.

An attractive vision for soft robots is that regular people (i.e., not professional robot designers) can effectively create new robots to help them with tasks. This "robot for every task," approach is motivated by the goal of democratizing the technology and its benefits across a wide spectrum of users [1], as well as developing robotic devices that are uniquely suited to the needs of particular users [2]. Design optimization can automate the process of new robot creation.

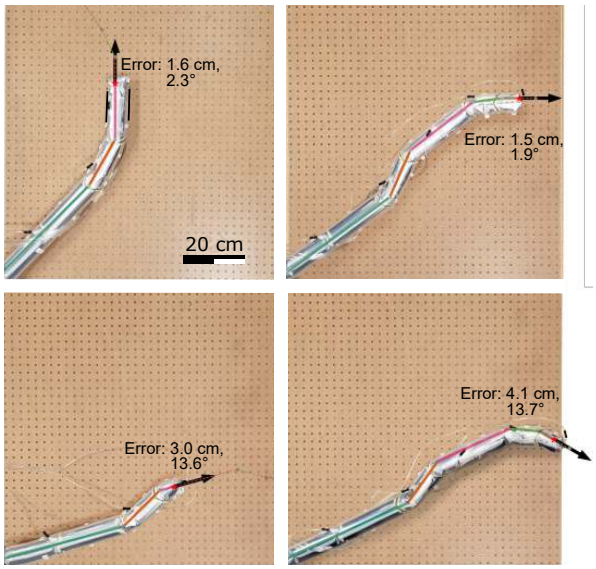

Fig. 1. A soft growing vine robot design with four discrete joints, optimized to achieve four different targets, each at a different orientation. Each colored segment indicates a link, and the joints are between the links. All robots are grown from a rotating base (not shown). Due to growth, only the necessary joints are exposed for each target. Here, a *single* robot design reaches all targets with their specified orientations. Difference between the computed linear and angular positions of the end effector and those achieved by the hardware is listed for each target as "Error".

Toward the goal of creating soft robot manipulators that approach dexterity similar to that of traditional rigid robot arms while bringing advantages in compliance, low cost, and increased workspace, here we develop a new design optimization process coupled with the unique properties of soft growing "vine" robots [3] with stiffening capabilities and discrete joints [4] in order to create custom soft robot arms that can reach a set of specific targets, avoid known obstacles, maximize the workspace, while minimizing construction and actuation cost. The outcome of the design optimization is the specification for a soft serial-chain manipulator, which is then rapidly constructed and demonstrated to achieve the design goals, as shown in Fig. 1.

A. Related Work

1) Soft, inflated-beam, growing vine robots: Continuum robots characterized by tip extension, significant length change, and directional control are termed "vine robots," due

This work was supported in part by National Science Foundation grants 1953008 and 2024247, a National Science Foundation Graduate Research Fellowship, and an ARCS Foundation Fellowship.

[1]I. Exarchos, K. Wang and C. K. Liu are with the Department of Computer Science, Stanford University, Stanford, CA 94305, USA. exarchos@stanford.edu, karenw24@stanford.edu, karenliu@cs.stanford.edu

[2]B. H. Do, and A. M. Okamura are with the Department of Mechanical Engineering, Stanford University, Stanford, CA 94305, USA. brianhdo@stanford.edu, aokamura@stanford.edu

[3]F. Stroppa is with the Faculty of Computer Engineering, Kadir Has University, Turkey. fabio.stroppa@khas.edu.tr

[4]M. M. Coad is with the Department of Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA. mcoad@nd.edu

to their similar behavior to plants with the growth habit of trailing. In our instantiation, vine robots are inflated beams that extend in length from the tip using internal air pressure to pass the material of a flexible, tubular body through its center and turn it inside out at the tip (a process called eversion) [3], [5]. These robots can be made of low-cost materials, such as low-density polyethylene (LDPE) plastic, and can grow to very long lengths from a compact base, giving them a large workspace. They can also use active, reversible steering mechanisms based on cable tendons [6], [7], [8] or pneumatic artificial muscles [7], [9], [10], [11], [12] that are compatible with the growing mechanism. A number of researchers have recognized the advantages of tip growth for movement and manipulation in cluttered and constrained environments, because the robot body does not slide with respect to the environment [13], [14], [15], [16].

In prior work, vine robot tip orientation has not been independent from its position [17]. Here we build upon recent results [4], [18], [19] that lock the robot's shape or use the environment to allow the robot tip to reach targets at different orientations. In particular, we use layer jamming to control the stiffness of vine robot sections [4] in order to create discrete joints at selected locations along the length of the robot. It is also possible to stiffen regions of the robot to improve shape control and payload handling, although this is beyond the scope this work [4].

2) Design optimization of soft robots: Due to their infinite degrees of freedom and large design space that is linked to their physical properties, soft robots are prime candidates for design optimization. Design can be interpreted as the set of parameters and characteristics used to build a soft robot, which might include lengths, actuators, joints, material properties, and local shape. Many methods for design optimization exist, including greedy algorithms [20], sensitivity analysis [21], finite element analysis [22], supervised learning for sensor placement [23], geometric modeling to achieve desired deformations [24], genetic algorithms for dexterity [25], gait parameters [26], and counterweight balance [27]. Furthermore, optimization has been used for control of soft robots, for example using reinforcement learning [28], [29].

Although design planning has been performed for passive, pre-shaped, everting tubes that exploit environmental contacts [30], design optimization of actively steered vine robots has not yet been approached. In this work, we propose methods to balance competing demands on a robot design, including the limitations of actuation, while also considering geometric constraints that define the proposed task.

II. Problem Statement

Fig. 2 shows a schematic of the vine robot considered in this work. The robot originates from a base from which it can grow from its tip or retract. The base can rotate and thus acts as an active position-controlled joint. The robot body contains pre-fabricated sections whose stiffness can be independently controlled using layer jamming. By stiffening all sections except one and pulling on one of two cables in the planar case, the robot body bends at the proximal end

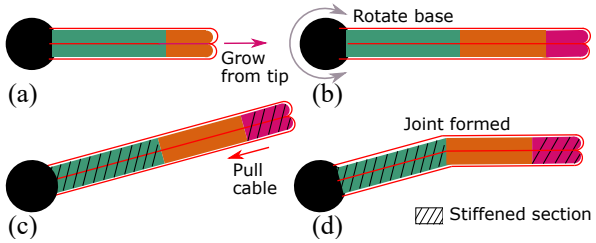

Fig. 2. The vine robot has three distinct actuation modes. (a) Through pressure-driven eversion, the robot can add material to its tip. The robot can also shorten in length by retracting from the tip. (b) The material for this growth comes from a spool in a fixed based, which is free to rotate. (c) By stiffening certain sections while leaving others soft, a joint can be formed by pulling on a cable. (d) The angle of the joint is determined by the cable lengths and can be subsequently preserved by stiffening that section. Repeating (c) and (d) allows other joints to bend.

of the softened segment, creating an effective revolute joint. When that section is then stiffened, the bend is held in place, then a different section is softened, and a cable is pulled to create a new bend. In general, we produce n' discrete joints by fabricating n' robot sections. Because of the ability to grow and retract, the vine robot can grow a different number of joints depending on how many have been everted, unlike conventional robots that have a constant number of joints. Those joints that have not been everted do not contribute to the robot shape.

Rotation by cable pulling introduces some mechanical limitations. Joint rotations near the tip of the robot are challenging due to small applied moments. Additionally, there may be a maximum bending angle less than self-intersection. Bending an inflated beam results in spring-like behavior where the beam produces a resistance torque proportional to the angle it is bent [31]. Depending on the maximum beam wall stiffness, the maximum torque the jammed layers can resist may depend on the angle of the joint. For our demonstrations, we choose $[-30°, 30°]$ as an estimate for the range of controllable angles.

A greater number of joints results in greater dexterity and a larger workspace. However, there is a trade-off between these attributes and the time and complexity of robot fabrication, as well as the complexity of controlling the robot once fabricated. For many applications with a predictable set of tasks, the locations of targets may be known ahead of time. Thus, our primary goal is to minimize the fabrication time and cost by producing a robot with the fewest number of joints while still reaching all targets. However, a fixed design is capable of reaching many more targets than those it was explicitly designed for, and situations may arise where we wish to reach targets whose locations are not known ahead of time. Thus, for a given fixed design, we also evaluate whether additional target positions/orientations can be achieved.

III. Design Optimization

Given the vine robot's base location, a set of target end effector locations and orientations, and a set of known obstacles, we wish to obtain an optimal design of the vine robot that reaches all targets, avoids all fixed obstacles, covers the largest target space, and satisfies all hardware

constraints. The design parameters include the total number of the potential joints and their locations along the vine robot, which also determine the length of each link of the robot. These choices need to be set before the robot is fabricated.

Unlike rigid robot arms for which the design space can be simply represented by the length of each link, the vine robot's design space needs to consider a set of unique characteristics owing to its ability to grow and retract: (1) Different targets might require different numbers of links to reach. (2) Any interior link has a fixed length across different targets while the length of the terminal link can vary for different targets, though not exceeding its maximum length (which is equal to the one it possesses as an interior link). (3) A link can be interior or terminal for different targets. (4) The number of links required for a given set of targets is unknown.

Figure 1 illustrates one possible design that can reach four targets with the desired orientations. This design requires four potential joints to be placed at 0.44, 0.61, 0.88 and 0.98 meters from the base of the vine robot. For each target, the vine robot needs to expose a different number of joints to reach. The length of each link is fixed if the link is not the terminal one (the most distal one). For the terminal link, it can grow or shrink arbitrarily within the bounds.

A. Formulating the Optimization

Given a budget of n links of a vine robot, we wish to determine their lengths $\boldsymbol{\ell} \in \boldsymbol{R}_+^n$ and a set of bending angles $\boldsymbol{Q} = [\boldsymbol{q}_1^T, \boldsymbol{q}_2^T, \ldots, \boldsymbol{q}_m^T]$, such that m pairs of target location and orientation, $\boldsymbol{T} = [(\boldsymbol{t}_1, \phi_1), \ldots, (\boldsymbol{t}_m, \phi_m)]$ can be reached with the same design while avoiding fixed obstacles $\boldsymbol{o} \in \mathcal{O}$ and maximizing the workspace. A design is defined solely by $\boldsymbol{\ell}$. A particular configuration $\boldsymbol{q}$ to reach a given target is not part of the design. The variables of optimization are $\boldsymbol{\ell}$ (shared across all targets) and $\boldsymbol{Q} = [\boldsymbol{q}_1^T, \boldsymbol{q}_2^T, \ldots, \boldsymbol{q}_m^T]$, the collection of all configurations, different for each target.

At first glance, this problem involves optimizing continuous variables (i.e., link lengths and joint angles) and solving for a discrete variable—how many joints do we need for each target? We can bypass answering this discrete question, thereby avoiding solving a challenging mixed-integer program, if we exploit the unique aspects of soft growing robots. Instead of following the conventional rigid robot design, in which the end-effector is always at the tip of the robot arm, we allow any point along the vine robot to be the end-effector. If a point $\boldsymbol{p}$ on the vine robot can reach the target with the desired orientation, we consider it a successful design for this target and discard all the materials and joints beyond $\boldsymbol{p}$. The fact that the vine robot has the ability to retract makes our formulation possible. Concretely, we solve for a vector $\boldsymbol{\ell} \in \boldsymbol{R}_+^n$, but the optimizer might end up only using $n' < n$ links to reach all m targets and yielding a design with only n' links. Our method results in a generic continuous optimization which automatically determines the optimal number of links required for each target.

1) Constraints: The hardware constraints are specified by a maximum bending angle at any given joint in the robot: with the exception of the root joint (i.e., the base of the robot) which is allowed to rotate freely, bending angles of all other joints are restricted to $[-30°, 30°]$. Another constraint is that the minimum possible length of a link is 0.1 m, while we do not consider links longer than 1 m.

2) Objective function: The geometric interpretation of a feasible solution requires targets to lie on the line segment corresponding to any one of those links, with the link orientation matching the prescribed target orientation and no collisions between any link and obstacle. In addition, we would like the design to maximize the workspace.

a) Target location: The (minimum) distance of the target to the line segment represented by the link. The minimum distance from a target $\boldsymbol{t}_j$ to link i of the current estimate of design is

$$d(\boldsymbol{\ell}, \boldsymbol{q}_j, \boldsymbol{t}_j, i) = \|\boldsymbol{t}_j - \boldsymbol{t}^*(\boldsymbol{\ell}, \boldsymbol{q}_j, i)\|,$$

where $\boldsymbol{t}^*$ is either $\boldsymbol{t}_j$'s projection on link i, or one of link i's endpoints, if the projection lies outside the link. Note that to discourage solutions with targets lying very close to a link node (which would lead to very small end-effector links), we prefer the target to lie between $30 - 90\%$ of the link length.

b) Target orientation: The absolute difference between link orientation and prescribed target orientation ϕ_j, namely,

$$o(\boldsymbol{q}_j, \phi_j, i) = |\phi_j - \sum_{k=1}^{i} \boldsymbol{q}_j(k)|,$$

where i is the link index we are considering and $\boldsymbol{q}_j$ is the current estimate of robot configuration for target j.

These two quantities are weighted by β to balance the difference in units. The weighted sum represents the cost for each individual link i, for a given target $(\boldsymbol{t}_j, \phi_j)$,

$$L_{tar}(\boldsymbol{\ell}, \boldsymbol{Q}) = \sum_{j=1}^{m} \frac{\min_i \left(d(\boldsymbol{\ell}, \boldsymbol{q}_j, \boldsymbol{t}_j, i) + \beta\, o(\boldsymbol{q}_j, \phi_j, i) \right)}{\|\boldsymbol{t}_j\|^2}. \tag{1}$$

Taking the minimum across links yields the cost of the "active end-effector" link, i.e., the particular link that will be considered towards reaching the target. These procedures are repeated across all targets, producing a vector of costs. Those costs are then weighted by the inverse of the target squared distance from the origin (thus giving priority to closer targets which are more difficult to satisfy; this is because the bending limits force us to use fewer links to reach close targets, thus fewer degrees of freedom) and summed up to furnish the total cost. Note that the "active end-effector" link may change during the process of optimization.

c) Obstacle Violation: When attempting to reach a desired target, a cost will incur if any part of the robot intersects with an obstacle. We implement a collision checking function $b(\boldsymbol{\ell}, \boldsymbol{Q}, i, j, k)$ against two types of obstacles, polygons and circles. b is a binary function that returns 1 if link i at pose j is intersecting with obstacle k, and returns 0 otherwise. The loss function for the obstacle violation can be defined as:

$$L_{obs}(\boldsymbol{\ell}, \boldsymbol{Q}) = \begin{cases} c, & \text{if } \exists\ i, j, k \ \text{ s.t. } b(\boldsymbol{\ell}, \boldsymbol{Q}, i, j, k) = 1 \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where c is a large constant, indicating the penalty of obstacle violation. We set $c = 20$ in our implementation.

d) Coverage of workspace: The design should also maximize the coverage space. Since an analytical solution for coverage is difficult to obtain, we opt for a Monte Carlo approximation, where we sample uniformly in the 3D target space of (t_x, t_y, ϕ) and check whether each sampled target is reachable. We define the coverage of a design, $g(\boldsymbol{\ell})$, as the ratio of reachable targets to the total sampled targets. Since we need to check reachability for a large number of targets every time we evaluate $g(\boldsymbol{\ell})$, we need a very fast routine to check whether a design can reach a sampled target.

Algorithm 1 checks the reachability using the first n' links, where n' iterates from 2 to $|\boldsymbol{\ell}|$, the total number of links in the design. For each iteration, we determine a target segment $\overline{\boldsymbol{tp}}$, where $\boldsymbol{p}$ is a point on the line extended backward from $\boldsymbol{t}$ aligned with ϕ, subject to $\|\boldsymbol{t} - \boldsymbol{p}\| = \sum_{i=n'}^{|\boldsymbol{\ell}|} \ell_i$. We first check whether the most straight configuration ($\boldsymbol{q} = \mathbf{0}$) using the first $n' - 1$ can intersect $\overline{\boldsymbol{tp}}$. If so, adding n'-th link from the intersection point to $\boldsymbol{t}$ will allow the robot to reach the target at the desired orientation ϕ. We still will have to check if the intersection angle is within the joint limit. If not, we bend the first $n' - 1$ links of the robot to the joint limits ($\boldsymbol{q} = \mathbf{30}^\circ or -\mathbf{30}^\circ$) and check intersection angle with $\overline{\boldsymbol{tp}}$. If the most bent configuration no longer intersects $\overline{\boldsymbol{tp}}$, we unbend the links one by one starting with the first link. If at any point, the intersection angle is within the acceptable bending range, Alg. 1 returns `true` and exits.

The cost function can be summarized by:

$$L(\boldsymbol{\ell}, \boldsymbol{Q}) = w_t\, L_{tar}(\boldsymbol{\ell}, \boldsymbol{Q}) + w_b\, L_{obs}(\boldsymbol{\ell}, \boldsymbol{Q}) - w_g\, g(\boldsymbol{\ell}). \quad (3)$$

Algorithm 1: Reachable Check

Input: target $(\boldsymbol{t}, \phi)$ and link lengths $\boldsymbol{\ell}$
if $|t| \le \ell_0$ **then**
  return $|\phi| < \epsilon$
for $n' = 2 : |\boldsymbol{\ell}|$ **do**
  $\boldsymbol{p} = (t_x - \sum_{i=n'}^{|\boldsymbol{\ell}|} \ell_i \cdot \cos\phi, t_y - \sum_{i=n}^{|\boldsymbol{\ell}|} \ell_i \cdot \sin\phi)$
  $\boldsymbol{q} = \mathbf{0}$
  (b, θ) = CheckIntersect$(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{\ell}, \boldsymbol{q}, n')$
  if $b ==$ *false* **then**
    continue
  if $-30^\circ < \theta < 30^\circ$ **then**
    return *true*
  if $\theta > 30^\circ$ **then**
    $\boldsymbol{q} = \mathbf{30}^\circ$
  else
    $\boldsymbol{q} = -\mathbf{30}^\circ$
  for $j = 1 : n' - 1$ **do**
    (b, θ) = CheckIntersect$(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{\ell}, \boldsymbol{q}, n')$
    if $b ==$ *true and* $|\theta| < 30^\circ$ **then**
      return *true*
    $q_j = 0$
return *false*

Algorithm 2: CheckIntersect

Input: target $(\boldsymbol{t}, \phi)$, point $\boldsymbol{p}$, link lengths $\boldsymbol{\ell}$, link configurations $\boldsymbol{q}$, and number of links used n'
Output: existence of intersection b and angle of intersection θ
$\boldsymbol{r} = (\sum_{i=0}^{|n'-1|} \ell_i \cdot \cos\phi, \sum_{i=0}^{|n'-1|} \ell_i \cdot \sin\phi)$
$\mathcal{S} = \overline{\boldsymbol{tp}} \cap \text{circle}(\mathbf{0}, |\boldsymbol{r}|)$
if $\mathcal{S} = \emptyset$ **then**
  return *(false, 0)*
$\boldsymbol{s}_0$ = closest$(\boldsymbol{t}, \mathcal{S})$
$\theta = \arccos(\boldsymbol{s}_0, \boldsymbol{t} - \boldsymbol{p})$
return *(true, θ)*

Algorithm 3: Adaptive Stochastic Search for Design

Input: objective function $L(\boldsymbol{x})$, where $\boldsymbol{x} = [\boldsymbol{\ell}, \boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_m]$, initial guess $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, max link budget n_{max}, learning rate α, shape function S, number of samples K, number of iterations N, sampling variance lower bound $\varepsilon = 10^{-3}$
for $n = 2, \ldots, n_{max}$ *or feasible design found* **do**
  Initialize $\boldsymbol{\ell}$ to be size n
  for $i = 1, \ldots, N$ *or convergence criterion* **do**
    for $k = 1, \ldots, K$ **do**
      $\boldsymbol{x}^k = \boldsymbol{\mu}_i + \Delta\boldsymbol{x}^k, \quad \Delta\boldsymbol{x}^k \sim \mathcal{N}(0, \boldsymbol{\sigma}_i^2)$
        `/* σ² means element-wise multiplication */`
      Clip $\boldsymbol{x}^k$ for constraints
      $L^k = -L(\boldsymbol{x}^k)$;
    $L_{min} = \min_k L^k$, $L_{max} = \max_k L^k$
    for $k = 1, \ldots, K$ **do**
      $L^k = \frac{L^k - L_{min}}{L_{max} - L_{min}}$;
      $S^k = S(L^k)$
    $\boldsymbol{\mu}_{i+1} = \boldsymbol{\mu}_i + \alpha \frac{\sum_{k=1}^K S^k \Delta\boldsymbol{x}^k}{\sum_{k=1}^K S^k}$
    $\boldsymbol{\sigma}_{i+1} = \sqrt{(\sum_{k=1}^K S^k (\Delta\boldsymbol{x}^k)^2 + \boldsymbol{\varepsilon})}$ Clip $\boldsymbol{\mu}_{i+1}$ for constraints
  $\boldsymbol{x}^* = \boldsymbol{\mu}_N$
return $\boldsymbol{x}^*$

B. Solving the Optimization

The nature of the cost is highly nonlinear, non-convex, non-differentiable, and with a plethora of local minima. For these reasons, we opted for gradient-free, sampling-based optimization, specifically Adaptive Stochastic Search [32]. Adaptive stochastic search is a sampling-based method within stochastic optimization that transforms the original optimization problem via a probabilistic approximation. The core concept behind this algorithm is approximating the gradient of the objective function by evaluating random perturbations around some nominal value of the variable of optimization, a concept that also appears under the name Stochastic Variational Optimization and shares many similarities with natural evolution strategies and the Cross Entropy

Method [33], [34]. Given the objective function $L(\cdot)$ and an initial guess solution $\boldsymbol{x}$, where $\boldsymbol{x} = [\boldsymbol{\ell}, \boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_m]$, we sample K solutions using mean $\boldsymbol{\mu} = \boldsymbol{x}$ and variance $\boldsymbol{\sigma}^2$ (we abuse the notation here to indicate $\boldsymbol{\sigma}$ multiplying by itself element-wise), evaluate the objective function for each one of them. and map its value using a shape function $S(\cdot)$; in our implementation we used $S(L) = \exp(10L)$. We then normalize the values $S(L)$ in $[0, 1]$ and update the mean and variance of our sampling through an S-weighted combination of the sampled solutions; thus, the better a sampled solution is, the more it influences the newly calculated mean and variance. To satisfy the constraints, samples and updated $\boldsymbol{\mu}$'s are clipped to the allowable range of values. The procedure is summarized in Alg. 3. While [32] was developed to accommodate sampling from any distribution of the exponential family, we opted for Gaussian sampling for its simplicity. Furthermore, we applied clipping to ensure constraints are satisfied, and performed normalization of L to improve convergence of the algorithm. However, there are no mathematical guarantees of convergence, as addressed in [32]. For our problem that has a plethora of local minima, while the sampling nature of Alg. 3 helps overcome some of these local minima, there is not theoretic guarantee for global optimality.

The link budget n is also a discrete variable, but we can avoid formulating a mixed-integer program using a simple 1-D search. That is, we optimize over n by repeating the aforementioned optimization procedure starting from $n = 2$ and increasing the budget until a solution that satisfies all targets/orientations is obtained (see the outer loop in Alg. 3). This is practical because, due to the low dimensionality of the optimization problem for each n, a solution can be obtained relatively quickly. The code is available at `https://github.com/iexarchos/SoftRobotDesOpt.git`

IV. Evaluation

A. Fabrication of an Optimal Design

We demonstrated the entire process of prototyping a vine robot from design optimization to hardware fabrication. We considered a base at $(0.0, 0.0)$ and four targets located at $(0.4, 0.65)$, $(0.8, 0.6)$, $(0.9, 0.4)$, and $(0.6, 0.25)$ (all units in meters), with corresponding prescribed end-effector orientations of $90°$, $0°$, $-30°$, and $15°$, respectively. The result of this optimization is shown later in Fig. 6(a), in comparison to designs with different constraints.

The vine robot consists of a main body containing embedded pouches and a base through which the robot is actuated. Two cable tendons, on the left and right sides, are routed along the vine robot length and connected to motorized spools. The eversion and retraction of the robot are controlled using a motorized spool in the base. The base is connected to an air supply, which provides the pressure for growth. The robot body is fabricated from two LDPE tubes. Chevron-shaped pouches in the body are formed by heat sealing the tubes together. Layer stacks were then placed into the pouches and secured to the vine robot body using double-sided tape. A 1/8 inch outer diameter plastic tube was routed

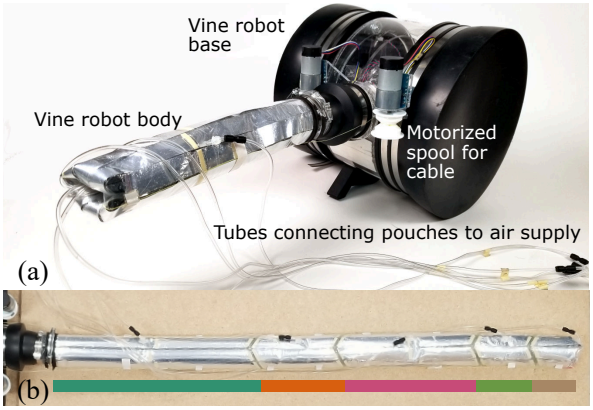

Fig. 3. (a) The vine robot is actuated by two cables running along its length and controlled via two motorized spools. Pouch pressure is controlled via tubes connected to each pouch. (b) Vine robot segment lengths were fabricated using the optimized $\boldsymbol{\ell}$. Each color corresponds to a link ℓ_i.

to each pouch, allowing for control of each pouch pressure. By connecting the pouch to the vine robot pressure source, pouch pressure equaled the vine robot body pressure and the layers were unjammed. By disconnecting the tube from the body pressure source, the layers were jammed.

For a specified link budget n, the optimization returns a sequence of n' link lengths $\boldsymbol{\ell}$ to construct. These link lengths correspond directly to the lengths of the layer stacks and pouches to be fabricated. Layers are cut into parallelograms and assembled into stacks of 14 layers each. We laser cut the layers from 0.05 mm thick aluminum-sputtered polyester film. Each pouch has eight layer stacks arranged circumferentially. The optimization in this example yielded a design consisting of five links, with link lengths 0.44, 0.17, 0.27, 0.10, and 0.11 meters, ordered from the base to the tip. Fig. 3(b) shows the fabricated vine robot. Fig. 1 shows how the fabricated robot achieves four targets. One source of error is how joints on the actual robot, which are formed through buckling, may occur at different locations than the modeled revolute joints. Another is difficulty in precise control of joint angles and last link length. Further quantitative performance analysis will be performed in future work.

B. Target Reachability Analysis

To assess the algorithm performance in optimizing robot design and configurations, we tested it on randomly sampled targets and orientations. Specifically, we varied both the number of targets, m, and the link budget, n. This introduced the following difficulty: when picking a set of targets and orientations randomly, it is unclear whether a failure of the algorithm to find a solution given a link budget is due to the algorithm's performance, or because a feasible solution satisfying the constraints does not exist for the given link budget. To avoid this problem, we sampled m targets and orientations by first sampling a random five-link solution with m different configurations, and then obtaining the targets and configurations by selecting points along any of the links of the robot, along with their respective link orientation. In this way a feasible solution is guaranteed to

TABLE I

DESIGN OPTIMIZATION SUCCESS RATE

Link budget (n)	# targets (m)				
	2	3	4	5	6
2	0.5	0.33	0.5	0.2	0.33
3	1.	1.	0.5	0.8	0.67
4	1.	1.	0.75	1.	1.

Fraction of targets satisfied for a given link budget and number of targets.

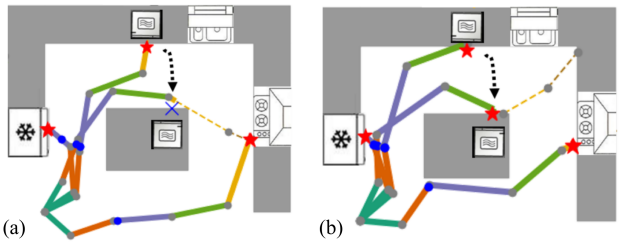

Fig. 4. Coverage differences between two designs. (a) Without the coverage function (i.e. removing the last term in Equation 3), if the microwave is moved from the counter to the island, it is no longer a reachable target. (b) With the addition of the coverage function, the design has a greater coverage and can still reach the microwave even if it is moved.

exist as long as the link budget during optimization is $n \geq 5$. The results of this analysis are shown in Table I. For the link budget above 4, our algorithm has a 100% success rate in obtaining a solution for a feasible target.

C. Workspace Coverage Analysis

We now demonstrate example workspaces for given robot designs. We randomly sampled two thousand points (t_x, t_y, ϕ) in $[0,1] \times [0,1] \times [-90°, 90°]$ and computed the ratio of reachable targets to total number of sampled points in the target space. The coverage computation yields a 33.35% success rate. The design optimized without the coverage function yields a lower success rate, 27.15%. Fig. 4 illustrates the difference between two designs in an environment representing a kitchen, including an island as an obstacle.

D. Multiple Sequential Targets

Once the design is optimized and fabricated, the robot can reach a sequence of targets through a series of retracting, growing, and bending without resetting to the base. However, different target orderings may result in different strategies due to obstacles. We implement a simple method that first interpolates the two configurations $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ solved independently for two consecutive targets. We check all collision points between obstacles and interpolated configurations and select the closet collision point $\boldsymbol{p}_c$ from the home base. We then retract $\boldsymbol{q}_A$ until the robot is within $|\boldsymbol{p}_c|$-radius from the home base. Once the robot retracts and bends over to configuration $\boldsymbol{q}_B$, it can then grow as needed to reach the next target. Fig. 5 shows a 2D visualization of a kitchen space, along with the configurations to reach each target.

E. Trade-off in Design

To demonstrate the utility of the algorithm as a tool to understand the various designs that could be built given different constraints, we showcase different designs for the same set of targets as those introduced in Section IV-A. The various designs are shown in Fig. 6. Specifically, we compare our original design (Fig. 6(a)) to the resulting design if the link budget, n, is increased from 5 to 8 (Fig. 6(b)) and find that the solution is approximately the same, and only 5 of the 8 links are used. In Fig. 6(c), a design is shown for a robot that is able to bend $45°$ instead of $30°$; the solution requires only 3 links. Conversely, a robot being able to bend only $15°$ would require 6 links, as shown in Fig. 6(d).

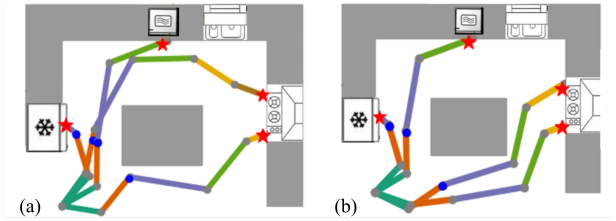

Fig. 5. Different sequences of reaching the four targets. The point of retraction is marked by the blue dot. (a) Fridge, top of stove, bottom of stove, microwave. (b) Bottom of stove, top of stove, microwave, fridge.

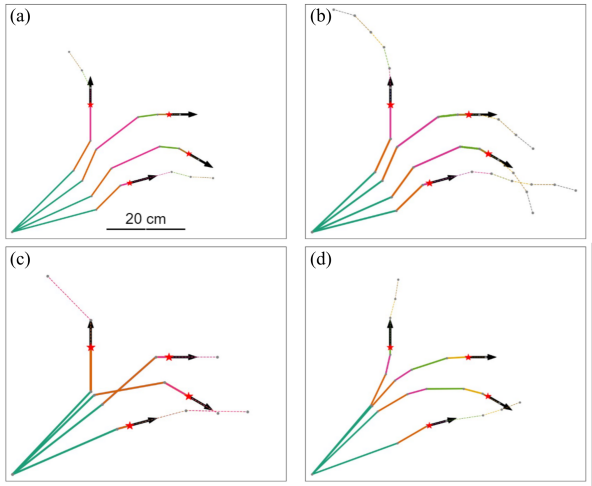

Fig. 6. Alternative designs for varying hardware constraints and link budgets. (a) The original design, described in Sec. IV-A. (b) Using a budget of 8 links yields approximately the same solution: only 5 of them are used. (c) If the robot were able to bend $45°$, only 3 links are necessary. (d) If the robot were only able to bend $15°$, 6 links are necessary.

V. CONCLUSIONS AND FUTURE WORK

In this work, we presented a method to optimize the design of a soft growing vine robot to reach a set of given targets at specified approach angles, while avoiding the obstacles and maximizing the space coverage, such that the design can also be used to achieve other targets not originally specified. A sample problem of 4 targets generated a design with 5 links, which was fabricated and demonstrated to achieve the task. These results are a key step toward low-cost, bespoke soft robots for user-defined tasks in home environments.

In the future, we plan to achieve the following objectives: extend the optimizer to 3D scenarios; exploit other optimization methods to explore the research space; implement advanced obstacle-avoidance algorithms for navigation in cluttered environments; and improve the fabrication process to enable a faster, tighter loop between robot design and fabrication.

REFERENCES

[1] A. Schulz, C. Sung, A. Spielberg, W. Zhao, R. Cheng, E. Grinspun, D. Rus, and W. Matusik, "Interactive robogami: An end-to-end system for design of robots with ground locomotion," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1131–1147, 2017.

[2] T. K. Morimoto, J. D. Greer, E. W. Hawkes, M. H. Hsieh, and A. M. Okamura, "Toward the design of personalized continuum surgical robots," *Annals of Biomedical Engineering*, vol. 46, no. 10, pp. 1522–1533, 2018.

[3] E. W. Hawkes, L. H. Blumenschein, J. D. Greer, and A. M. Okamura, "A soft robot that navigates its environment through growth," *Science Robotics*, vol. 2, no. 8, p. eaan3028, 2017.

[4] B. H. Do, V. Banashek, and A. M. Okamura, "Dynamically reconfigurable discrete distributed stiffness for inflated beam robots," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 9050–9056.

[5] L. H. Blumenschein, A. M. Okamura, and E. W. Hawkes, "Modeling of bioinspired apical extension in a soft robot," in *Biomimetic and Biohybrid Systems. Living Machines 2017*, ser. Lecture Notes in Computer Science, M. Mangan, M. Cutkosky, A. Mura, P. Verschure, T. Prescott, and N. Lepora, Eds. Springer, 2017, vol. 10384, pp. 522–531.

[6] L. H. Blumenschein, L. Gan, J. Fan, A. M. Okamura, and E. W. Hawkes, "A tip-extending soft robot enables reconfigurable and deployable antennas," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 949–956, 2018.

[7] L. H. Blumenschein, N. S. Usevitch, B. Do, E. W. Hawkes, and A. M. Okamura, "Helical actuation on a soft inflated robot body," in *IEEE International Conference on Soft Robotics*, 2018, pp. 245–252.

[8] L. T. Gan, L. H. Blumenschein, Z. Huang, A. M. Okamura, E. W. Hawkes, and J. A. Fan, "3d electromagnetic reconfiguration enabled by soft continuum robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1704–1711, April 2020.

[9] J. D. Greer, T. K. Morimoto, A. M. Okamura, and E. W. Hawkes, "Series pneumatic artificial muscles (sPAMs) and application to a soft continuum robot," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 5503–5510.

[10] ——, "A soft, steerable continuum robot that grows via tip extension," *Soft Robotics*, vol. 6, no. 1, 2019.

[11] E. W. Hawkes, D. L. Christensen, and A. M. Okamura, "Design and implementation of a 300% strain soft artificial muscle," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 4022–4029.

[12] N. D. Naclerio and E. W. Hawkes, "Simple, low-hysteresis, foldable, fabric pneumatic artificial muscle," *IEEE Robotics and Automation Letters, Accepted*, 2020.

[13] D. A. Haggerty, N. D. Naclerio, and E. W. Hawkes, "Characterizing environmental interactions for soft growing robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 3335–3342.

[14] N. D. Naclerio, C. M. Hubicki, Y. O. Aydin, D. I. Goldman, and E. W. Hawkes, "Soft robotic burrowing device with tip-extension and granular fluidization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 5918–5923.

[15] Y. Ozkan-Aydin, M. Murray-Cooper, E. Aydin, E. N. McCaskey, N. Naclerio, E. W. Hawkes, and D. I. Goldman, "Nutation aids heterogeneous substrate exploration in a robophysical root," in *IEEE International Conference on Soft Robotics*, 2019, pp. 172–177.

[16] A. Sadeghi, A. Mondini, and B. Mazzolai, "Toward self-growing soft robots inspired by plant roots and based on additive manufacturing technologies," *Soft Robotics*, vol. 4, no. 3, pp. 211–223, 2017.

[17] M. M. Coad, L. H. Blumenschein, S. Cutler, J. A. R. Zepeda, N. D. Naclerio, H. El-Hussieny, U. Mehmood, J.-H. Ryu, E. W. Hawkes, and A. M. Okamura, "Vine robots: Design, teleoperation, and deployment for navigation and exploration," *IEEE Robotics and Automation Magazine*, vol. 27, no. 3, pp. 120–132, 2020.

[18] S. Wang, R. Zhang, D. A. Haggerty, N. D. Naclerio, and E. W. Hawkes, "A dexterous tip-extending robot with variable-length shape-locking," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 9035–9041.

[19] M. Selvaggio, L. A. Ramirez, N. D. Naclerio, B. Siciliano, and E. W. Hawkes, "An obstacle-interaction planning method for navigation of actuated vine robots," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 3227–3233.

[20] M. Koehler, N. Usevitch, and A. M. Okamura, "Model-based design of a soft 3D haptic shape display," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 613–628, 2020.

[21] V. Megaro, J. Zehnder, M. Bächer, S. Coros, M. H. Gross, and B. Thomaszewski, "A computational design tool for compliant mechanisms." *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 82–1, 2017.

[22] M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, and M. Gross, "Computational design of actuated deformable characters," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 1–10, 2013.

[23] A. Spielberg, A. Amini, L. Chin, W. Matusik, and D. Rus, "Co-learning of task and sensor placement for soft robotics," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1208–1215, 2021.

[24] L.-K. Ma, Y. Zhang, Y. Liu, K. Zhou, and X. Tong, "Computational design and fabrication of soft pneumatic objects with desired deformations," *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–12, 2017.

[25] D. M. Bodily, T. F. Allen, and M. D. Killpack, "Multi-objective design optimization of a soft, pneumatic robot," in *IEEE International Conference on Robotics and Automation*. IEEE, 2017, pp. 1864–1871.

[26] M. Tesch, J. Schneider, and H. Choset, "Expensive multiobjective optimization for robotics," in *IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 973–980.

[27] C. A. C. Coello, A. D. Christiansen, and A. H. Aguirre, "Using a new ga-based multiobjective optimization technique for the design of robot arms," *Robotica*, vol. 16, no. 4, pp. 401–414, 1998.

[28] Y. Ansari, M. Manti, E. Falotico, M. Cianchetti, and C. Laschi, "Multiobjective optimization for stiffness and position control in a soft robot arm module," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 108–115, 2017.

[29] E. Coevoet, A. Escande, and C. Duriez, "Optimization-based inverse model of soft robots with contact handling," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1413–1419, 2017.

[30] J. D. Greer, L. H. Blumenschein, R. Alterovitz, E. W. Hawkes, and A. M. Okamura, "Robust navigation of a soft growing robot by exploiting contact with the environment," *The International Journal of Robotics Research*, vol. 39, no. 14, pp. 1724–1738, 2020.

[31] C. R. Nesler, T. A. Swift, and E. J. Rouse, "Initial design and experimental evaluation of a pneumatic interference actuator," *Soft Robotics*, vol. 5, no. 2, pp. 138–148, 2018.

[32] E. Zhou and J. Hu, "Gradient-based adaptive stochastic search for non-differentiable optimization," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1818–1832, 2014.

[33] R. Y. Rubinstein, "Combinatorial optimization, cross-entropy, ants and rare events," in *Stochastic Optimization: Algorithms and Applications*. Springer, 2001, pp. 303–363.

[34] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.