

A Quadratic Programming Approach to Modular Robot Control and Motion Planning

Chao Liu

GRASP Lab and

Department of Mechanical Engineering

and Applied Mechanics,

University of Pennsylvania

Philadelphia, PA 19104

Email: chaoliu@seas.upenn.edu

Mark Yim

GRASP Lab and

Department of Mechanical Engineering

and Applied Mechanics,

University of Pennsylvania

Philadelphia, PA 19104

Email: yim@seas.upenn.edu

Abstract—Modular robotic systems consist of multiple modules that can be transformed into different configurations with respect to different needs. Different from robots with fixed geometry or configurations, the kinematics model of a modular robotic system can alter as the robot reconfigures itself. Since modular robotic systems are usually highly redundant for versatility, developing a generic approach for control and motion planning is difficult, especially when multiple motion goals are coupled. A new framework in terms of control and motion planning is developed. The problem is formulated as a linearly constrained quadratic program (QP) that can be solved efficiently. Some constraints can be incorporated into this QP, including a novel way to approximate environment obstacles. This solution can be used directly for real-time applications and it is validated and demonstrated on the CKBot and SMORES-EP modular robot platforms.

I. INTRODUCTION

Modular self-reconfigurable robot systems are usually composed of a small set of building blocks with uniform docking interfaces that allow the transfer of mechanical forces and moments, electrical power, and communication throughout the robot [1]. These platforms are designed to be versatile and adaptable with respect to different tasks, environments, functions or activities.

A single module in a modular robotic system usually has one or more degrees of freedom (DoFs). Combining many modules to form versatile systems results in robots requiring representations with high dimension. This dimensionality makes control and motion planning difficult. That the system is not a single structure but can take a very large number of configurations (typically exponential in the number of modules) requires an approach that can be applied to arbitrary configurations. For example, a modular robot configuration built by PolyBot [2] modules is shown in Fig. 1 which has multiple serial kinematic chains. This is different from common multi-limb systems with a single base. These systems can be modeled such that chains are decoupled. However, in chain-type modular robotic systems the chains often share many DoFs leading to a more complicated control problem.

In this paper, we present a new approach for modular robotic systems in terms of control and motion planning. In



Fig. 1: A modular robot configuration built by PolyBot modules is composed of multiple chains [2].

order to solve the problem in general, an universal kinematics model is required for arbitrary configurations and the approach is meant to be easily extended for more motion goals. We propose a quadratic programming approach that can be solved efficiently for real-time application. This requires that system stability, hardware limits, and motion constraints can be incorporated into this quadratic program (QP) as linear constraints. One advantage of this approach is that the large variety of configurations and kinematic structures can be represented easily as linear constraints, including situations where multiple portions of the robot may have different simultaneous goals. The framework is tested and evaluated on CKBot [3] and SMORES-EP [4] in the end.

The paper is organized as follows. Sec. II reviews relevant and previous works. Sec. III introduces the details to derive the kinematics model required to describe the motion of any modular robotic configuration. Sec. IV discusses the approach to control and motion planning for given tasks. Some experiments are validated in Sec. V with some analysis. Sec. VI includes the conclusions and future work.

II. RELATED WORK

There are a variety of methods for control and motion planning in locomotion with modular robots, e.g. [5] [6]. Whereas locomotion gaits usually feature repeated actions, manipulation typically focuses on the precise control of the end-effector. Motion planning for variable topology truss systems is presented in [7] but these robots are in a different morphology.

This paper addresses the manipulation and shape-morphing tasks for modular robots in tree topologies that are constructed from multiple serial chain configurations, and is particularly advantageous where some DoFs are shared among several branches. Work related to manipulation of modular robot systems includes inverse kinematics for highly redundant chain using PolyBot [2] [8], and constrained optimization techniques with nonlinear constraints [9]. Due to complicated constraints in these approaches, real-time applications for large systems cannot be guaranteed and numerical issue have to be addressed when solving the optimization problem in the presence of obstacles. Some quadratic programming approaches have been presented to deal with the inverse kinematics problem for redundant manipulators [10] considering joint limit constraints, and whole body manipulation planning [11] [12] with complex models for collision check requiring iterative computation. Our work also differs from [12] in how the motion goals are incorporated into the objective functions. In [12] the goals are in the objective function along with the weighted 2-norm of all joint velocities. However, there is no guarantee on the tracking performance because the magnitude of each part in the objective function can be different. Another part of works related are about controller design for modular robots, such as an adaptive control approach using a neural network architecture [13], a virtual decomposition control approach [14], a distributed control method with torque sensing [15], and a centralized controller [16]. These approaches only consider the control problem in a free environment and extra motion planning is required to fully control the system in a complex environment. Sampling-based planning approaches have been shown to be effective for high dimensional problem [17] [18] but tend to be slow and hard to integrate with motion constraints.

A new approach is presented in this paper to solve this problem. A more general solution to build kinematics models of modular robotic systems is introduced. Then the motion planning and control problem can be formulated as a linearly constrained quadratic program (QP) that can be solved efficiently for real-time applications.

III. KINEMATICS FOR MODULAR ROBOTS

A. Kinematics Graph

The representation of a modular robot configuration is discussed in [19] which is an undirected graph $G = (V, E)$. Each vertex $v \in V$ represents a module and each edge $e \in E$ represents the connection between two modules.

We use a *module graph* to model a module's topology among its all connectors and joints. A **module graph** is a directed graph $G_m = (V_m, E_m)$: each vertex is a rigid body in the module which is either a connector or the module body, and each edge shows how two adjacent rigid bodies are connected. The transformations among all rigid bodies are determined by its joint set and geometry. For example, a CKBot UBar module in Fig. 2a is a one DoF module as well as four connectors (TOP Face or \mathcal{T} , BOTTOM Face or \mathcal{B} , LEFT Face or \mathcal{L} , and RIGHT Face or \mathcal{R}). For simplicity, when the module joint is in its zero position, all rigid bodies

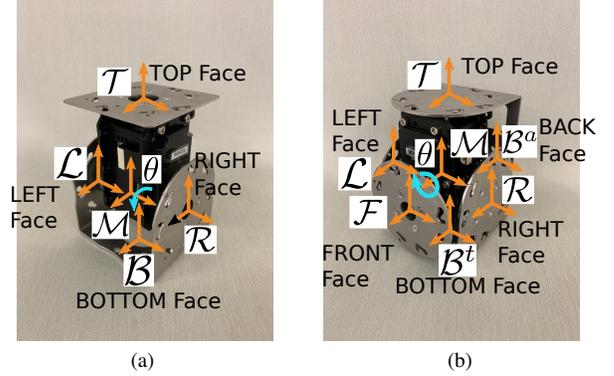


Fig. 2: (a) A CKBot UBar module has one DoF and four connectors; (b) A CKBot CR module has one DoF and six connectors.

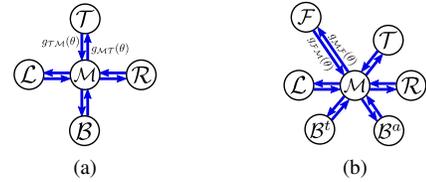


Fig. 3: (a) Module graph of a CKBot UBar module in which $g_{\mathcal{M}\mathcal{B}}$, $g_{\mathcal{B}\mathcal{M}}$, $g_{\mathcal{M}\mathcal{L}}$, $g_{\mathcal{L}\mathcal{M}}$, $g_{\mathcal{M}\mathcal{R}}$, and $g_{\mathcal{R}\mathcal{M}}$ are invariant of θ , and (b) module graph of a CKBot CR module in which $g_{\mathcal{M}\mathcal{B}^a}$, $g_{\mathcal{B}^a\mathcal{M}}$, $g_{\mathcal{M}\mathcal{B}^t}$, $g_{\mathcal{B}^t\mathcal{M}}$, $g_{\mathcal{M}\mathcal{T}}$, $g_{\mathcal{T}\mathcal{M}}$, $g_{\mathcal{M}\mathcal{L}}$, $g_{\mathcal{L}\mathcal{M}}$, $g_{\mathcal{M}\mathcal{R}}$, and $g_{\mathcal{R}\mathcal{M}}$ are invariant of θ .

are in the same orientation and the translation offsets among them are determined by the module geometry. Let \mathcal{B} be fixed in \mathcal{M} , then the homogeneous transformations among \mathcal{M} , \mathcal{L} , and \mathcal{R} are invariant of joint parameter θ because they are rigidly connected. Only the homogeneous transformation between \mathcal{M} and \mathcal{T} is not invariant to θ . This relationship can be fully represented in a directed graph shown in 3a. The edge direction denotes the direction of the corresponding forward kinematics. \mathbf{G}_m is the set of unique module graphs G_m for a modular robotic system since some systems have more than one type of module (e.g. CKBot in Fig. 2).

In general, given a module m with connector set \mathcal{C} and joint set Θ , a frame \mathcal{C} is attached to each connector $c \in \mathcal{C}$ and frame \mathcal{M} is attached to the module body. Let mapping $g_{\mathcal{F}_1\mathcal{F}_2} : Q \rightarrow SE(3)$ describe the forward kinematics from \mathcal{F}_1 to \mathcal{F}_2 in joint space Q , then $\forall c \in \mathcal{C}$, $g_{\mathcal{M}\mathcal{C}}$ and $g_{\mathcal{C}\mathcal{M}}$ can be defined with respect to Θ . The results for CKBot CR modules and SMORES modules are shown in Fig. 3b and Fig. 4. With module graph model, we can easily obtain the **kinematics graph** $G_K = (V_K, E_K)$ for a modular robot configuration which is constructed by composing the modules by connecting connectors. A directed edge is used to denote each connection and the transformation between the two mating connectors is fixed since they are rigidly connected. Using this kinematics graph, a *kinematics chain* from frame \mathcal{F}_1 to frame \mathcal{F}_2 can

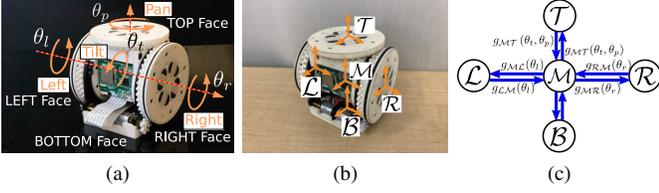


Fig. 4: (a) A SMORES-EP module has four DoFs and four connectors; (b) The frames of all rigid bodies are shown and \mathcal{B} is fixed in \mathcal{M} ; (c) Module graph of a SMORES module in which g_{MB} and g_{BM} are invariant of $\Theta = (\theta_l, \theta_r, \theta_p, \theta_t)$.

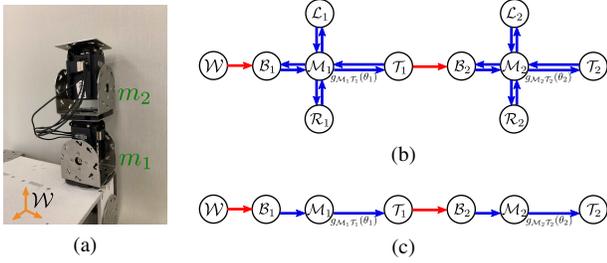


Fig. 5: (a) A configuration by two CKBot UBar modules, (b) its kinematics graph model, and (c) kinematics chain from \mathcal{W} to \mathcal{T}_2 .

be derived by following the shortest path $G_K : \mathcal{F}_1 \rightsquigarrow \mathcal{F}_2$. This creates a graph with no loops. A simple configuration built by two CKBot UBar modules is shown in Fig. 5. Frame \mathcal{W} is the world frame and module m_1 is fixed to it via its BOTTOM Face. The kinematics graph is shown in Fig. 5b and the kinematics chain from \mathcal{W} to \mathcal{T}_2 can be obtained shown in Fig. 5c. All the edges have fixed homogeneous transformations except for edge $(\mathcal{M}_1, \mathcal{T}_1)$ and edge $(\mathcal{M}_2, \mathcal{T}_2)$, and we can conclude that the forward kinematics mapping is $g_{\mathcal{W}\mathcal{T}_2} : \mathbb{T}^2 \rightarrow SE(3)$ where \mathbb{T}^p represents the p -torus. However, we can also see that all the edges in the shortest path from \mathcal{W} to \mathcal{L}_1 have fixed homogeneous transformations, so \mathcal{L}_1 is fixed in \mathcal{W} .

Similar to the configuration discovery algorithm in [19], the kinematics graph can be built by visiting modules in breadth-first-search order. The given configuration is traversed from the module fixed to the world frame \mathcal{W} . When visiting a new module m , denoting its parent via its connector c as \tilde{m} and the mating connector of \tilde{m} as \tilde{c} , record the fixed homogeneous transformation $g_{\mathcal{C}\tilde{c}}$ in which frame \mathcal{C} and frame $\tilde{\mathcal{C}}$ are attached to c and \tilde{c} respectively. Not until all modules are visited, is the $G_K = (V_K, E_K)$ of the given configuration constructed. With this structure, there is no need for case-by-case derivation of the kinematics as long as the kinematics for each type of module and connection are defined.

B. Kinematics for Modules

Recall that given a module m with connector set \mathcal{C} and joint set Θ , a frame \mathcal{C} is attached to each connector $c \in \mathcal{C}$ and frame \mathcal{M} is attached to the module body. For a joint

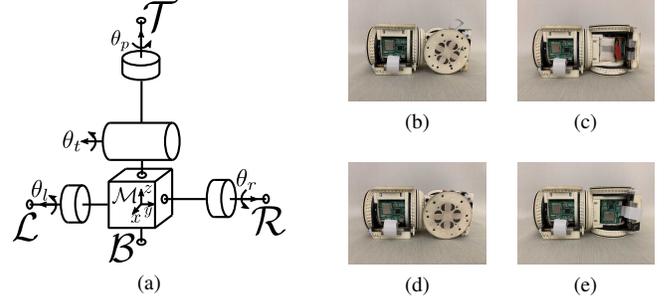


Fig. 6: (a) Kinematics for SMORES modules and (b) — (e) four cases to connect \mathcal{R} and \mathcal{T} .

$\theta \in \Theta$, a twist $\hat{\xi}_\theta \in se(3)$ can be defined with respect to \mathcal{M} in which $\xi_\theta = (v_\theta, \omega_\theta) \in \mathbb{R}^6$ is the twist coordinates for $\hat{\xi}_\theta^1$, and ξ is the set of the twist associated with each joint. For homogeneous transformation g_{MC} , it is straightforward to have

$$g_{MC} = g_{MC}(\Theta^C) = \prod_i \exp(\hat{\xi}_{\Theta_i}^C \Theta_i^C) g_{MC}(0) \quad (1)$$

in which Θ^C denote the parameter vector in joint space of the kinematics chain from \mathcal{M} to \mathcal{C} . If no joints are involved in the kinematics chain from \mathcal{M} to \mathcal{C} , then \mathcal{C} is fixed in \mathcal{M} and g_{MC} is a constant determined by the geometry of the module. g_{CM} is just the inverse of g_{MC} .

C. Kinematics for Chains

A kinematics chain from frame \mathcal{S} to \mathcal{F} can be obtained as $G_K : \mathcal{S} \rightsquigarrow \mathcal{F}$ where \mathcal{S} and \mathcal{F} are two vertices of G_K . In this kinematics chain, all the homogeneous transformations between connectors (e.g. $g_{\mathcal{T}_1\mathcal{B}_2}$ in Fig. 5c) are fixed and can be easily computed. The relative orientation between connectors needs considered that is determined by the connector design. For example, there are four cases to connect SMORES-EP modules shown in Fig. 6b — Fig. 6e due to the ring arrangement of the magnets on the connector. Then the homogeneous transformation $g_{\mathcal{S}\mathcal{F}}$ can be computed by multiplying the homogeneous transformation of each edge of path $G_K : \mathcal{S} \rightsquigarrow \mathcal{F}$ in order. In particular, let \mathcal{S} be world frame \mathcal{W} , if module m_1, m_2, \dots, m_N are involved in this chain, then the position of the origin of \mathcal{F} in \mathcal{W} is given by

$$p_{\mathcal{F}}^{\mathcal{W}} = g_{\mathcal{W}\mathcal{F}} [0 \ 0 \ 0 \ 1]^T \quad (2)$$

then the instantaneous spatial velocity of \mathcal{F} is given by the twist

$$\hat{V}_{\mathcal{W}\mathcal{F}}^s = \sum_{i=1}^N \sum_{j=1}^{N_i} \left(\frac{\partial g_{\mathcal{W}\mathcal{F}}}{\partial \theta_{ij}} g_{\mathcal{W}\mathcal{F}}^{-1} \right) \dot{\theta}_{ij} \quad (3)$$

in which θ_{ij} is the j th joint parameter of module m_i involved in this chain and the number of joints of module m_i involved in this chain is N_i . Rewrite Eq. (3) in twist coordinates as

$$V_{\mathcal{W}\mathcal{F}}^s = J_{\mathcal{W}\mathcal{F}}^s \dot{\Theta}^{\mathcal{W}\mathcal{F}} \quad (4)$$

¹Refer to [20] for background

in which

$$\Theta^{\mathcal{WF}} = [\theta_{11} \cdots \theta_{1N_1} \theta_{21} \cdots \theta_{2N_1} \cdots \theta_{N_1} \cdots \theta_{NN_n}]^\top \quad (5)$$

$$J_{\mathcal{WF}}^s = [J_1 \quad J_2 \quad \cdots \quad J_N] \quad (6)$$

$$J_i = \left[\left(\frac{\partial g_{\mathcal{WF}}}{\partial \theta_{i1}} g_{\mathcal{WF}}^{-1} \right)^\vee \quad \left(\frac{\partial g_{\mathcal{WF}}}{\partial \theta_{i2}} g_{\mathcal{WF}}^{-1} \right)^\vee \quad \cdots \quad \left(\frac{\partial g_{\mathcal{WF}}}{\partial \theta_{iN_i}} g_{\mathcal{WF}}^{-1} \right)^\vee \right] \quad (7)$$

and $J_{\mathcal{WF}}^s$ is the *spatial chain Jacobian*.

Define the twist of j th joint of module m_i with respect to \mathcal{W} as ξ'_{ij} that is

$$\xi'_{ij} = \left(\frac{\partial g_{\mathcal{WF}}}{\partial \theta_{ij}} g_{\mathcal{WF}}^{-1} \right)^\vee = \text{Ad}_{g_{\mathcal{WM}_i}} \xi_{ij}$$

in which $\text{Ad}_{g_{\mathcal{WM}_i}}$ is the adjoint transformation² and ξ_{ij} is defined in Sec. III-B for each joint in a module with respect to its module body frame. Then J_i becomes

$$J_i = [\xi'_{i1} \quad \xi'_{i2} \quad \cdots \quad \xi'_{iN_i}] \quad (8)$$

With this spatial chain Jacobian, the velocity of the origin of the frame \mathcal{F} is

$$v_{\mathcal{F}}^s = \widehat{V}_{\mathcal{WF}}^s p_{\mathcal{F}}^{\mathcal{W}} = (J_{\mathcal{WF}}^s \dot{\Theta}^{\mathcal{WF}})^\wedge p_{\mathcal{F}}^{\mathcal{W}} \quad (9)$$

For a module m_i in the kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}$ (\mathcal{M}_i is a vertex in the corresponding path), a sub-kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{M}_i$ can be defined with joint parameter vector $\Theta^{\mathcal{WM}_i} = [\theta_{11}, \theta_{12}, \cdots, \theta_{\bar{j}_i}]^\top$ where $\theta_{\bar{j}_i}$ is the parameter of \bar{j}_i th joint of module $m_{\bar{i}}$. For example, take the sub-kinematics chain from \mathcal{W} to \mathcal{M}_2 in Fig. 5c, then $i = 2$, $\bar{i} = 1$, $\bar{j}_i = 1$, since there is only one joint between \mathcal{W} and \mathcal{M}_2 which is the 1st joint of module m_1 . Then the *spatial module Jacobian* $J_{\mathcal{WM}_i}^s$ or $J_{\mathcal{M}_i}^s$ for simplicity can be defined as

$$J_{\mathcal{M}_i}^s = [\xi'_{i1} \quad \xi'_{i2} \quad \cdots \quad \xi'_{i\bar{j}_i}] \quad (10)$$

and the velocity of the origin of \mathcal{M}_i is

$$v_{\mathcal{M}_i}^s = (J_{\mathcal{M}_i}^s \dot{\Theta}^{\mathcal{WM}_i})^\wedge p_{\mathcal{M}_i}^{\mathcal{W}} \quad (11)$$

By replacing all the twists associated with joints after \bar{j}_i th joint of module $m_{\bar{i}}$ in the spatial chain Jacobian of chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}$ with 6×1 zero vectors, spatial module Jacobian can also be written as

$$J_{\mathcal{M}_i}^s = [\xi'_{i1} \quad \xi'_{i2} \quad \cdots \quad \xi'_{i\bar{j}_i} \quad 0_{6 \times 1} \quad \cdots \quad 0_{6 \times 1}] \quad (12)$$

then the velocity of the origin of \mathcal{M}_i is represented as

$$v_{\mathcal{M}_i}^s = (J_{\mathcal{M}_i}^s \dot{\Theta}^{\mathcal{WF}})^\wedge p_{\mathcal{M}_i}^{\mathcal{W}} \quad (13)$$

IV. CONTROL AND MOTION PLANNING

A. Control

Given the kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}$, the goal of the control task is to move $p_{\mathcal{F}}^{\mathcal{W}}$ (or $p_{\mathcal{F}}$ for simplicity) — the position of \mathcal{F} — to follow a desired trajectory.

Let $\tilde{p}_{\mathcal{F}} = \tilde{p}_{\mathcal{F}}(t)$ be the desired trajectory for the robot to track and $\tilde{v}_{\mathcal{F}}^s$ (or $\tilde{v}_{\mathcal{F}}$ for simplicity) is the derivative of $\tilde{p}_{\mathcal{F}}$, and the error and its derivative are defined as

$$e = \tilde{p}_{\mathcal{F}} - p_{\mathcal{F}} \quad \dot{e} = \dot{\tilde{p}_{\mathcal{F}}} - \dot{p}_{\mathcal{F}} = \tilde{v}_{\mathcal{F}} - v_{\mathcal{F}}$$

The error e can converge exponentially to zero as long as it satisfies

$$\dot{e} + Ke = 0 \quad (14)$$

in which K is positive definite. Substitute e and \dot{e}

$$\tilde{v}_{\mathcal{F}}^s - v_{\mathcal{F}}^s + K(\tilde{p}_{\mathcal{F}} - p_{\mathcal{F}}) = 0 \quad (15)$$

With Eq. (9), Eq. (15) can be rewritten as

$$(J_{\mathcal{WF}}^s \dot{\Theta}^{\mathcal{WF}})^\wedge p_{\mathcal{F}} = \tilde{v}_{\mathcal{F}}^s + K(\tilde{p}_{\mathcal{F}} - p_{\mathcal{F}}) \quad (16)$$

Eq. (16) is the control law to control the position of frame \mathcal{F} , namely $\dot{\Theta}^{\mathcal{WF}}$ (or $\dot{\Theta}^{\mathcal{F}}$ for simplicity) — the velocity of each involved joints that satisfies this equation — can move $p_{\mathcal{F}}$ to $\tilde{p}_{\mathcal{F}}$ in exponential time.

Suppose there are α motion goals $\tilde{p}_{\mathcal{F}_1}, \tilde{p}_{\mathcal{F}_2}, \cdots, \tilde{p}_{\mathcal{F}_\alpha}$, then the control law for all motion goals can be written as

$$\mathbf{JP} = \tilde{\mathbf{V}} + \mathbf{K}(\tilde{\mathbf{P}} - \mathbf{P}) \quad (17)$$

which is the stack of Eq. (16) for each motion goal. This makes the control problem for multiple motion goals easier without considering the fact that some motion goals may be coupled, that is some kinematic chains share DoFs. All we have to do is to build Eq. (16) for each individual motion goal and then stack them as linear constraints. Building a specific model for different combinations of motion goals is not necessary.

Recall that a modular robotic system is usually redundant so that there can be infinite number of solutions to Eq. (17) and this problem is formulated as a quadratic program

$$\begin{aligned} & \text{minimize} \quad \frac{1}{2} \dot{\Theta}^\top \dot{\Theta} \\ & \text{subject to} \quad \mathbf{JP} = \tilde{\mathbf{V}} + \mathbf{K}(\tilde{\mathbf{P}} - \mathbf{P}) \end{aligned} \quad (18)$$

where Θ is the set of joint parameters in kinematics chains $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_1, G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_2, \cdots, G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_\alpha$. Then solving (18) yields the minimum norm solution of joint velocities at every moment.

Joint position and velocity limits can be added to the quadratic program as inequality constraints

$$\frac{\Theta_{\min} - \Theta}{\Delta t} \leq \dot{\Theta} \leq \frac{\Theta_{\max} - \Theta}{\Delta t} \quad (19)$$

$$\dot{\Theta}_{\min} \leq \dot{\Theta} \leq \dot{\Theta}_{\max} \quad (20)$$

in which Δt is the time duration for the current step. Due to these two constraints, K cannot be too aggressive or there may not be solutions.

This optimization approach is helpful for many types of motion task. The controller can be used to move $p_{\mathcal{F}}$ to a desired position $\tilde{p}_{\mathcal{F}}$ by setting $\tilde{v}_{\mathcal{F}}^s = 0$, and it can also control $p_{\mathcal{F}}$ to move at a desired velocity by increasing $\tilde{p}_{\mathcal{F}}$ by $\tilde{v}_{\mathcal{F}} \Delta t$ for every time step.

²Refer to Chapter 2 in [20] for adjoint transformation definition

B. Motion Planning

The goal of the motion planning task is to enable a cluster of modules to navigate collision-free in an environment with obstacles.

1) *Frame Boundaries*: The cluster of modules can be kept in any polyhedral region in space which is defined by the boundaries of the environment. For a module m_i in the kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}$, let \hat{s}_{ij} be the unit direction vector from $p_{\mathcal{M}_i}^{\mathcal{W}}$ (or $p_{\mathcal{M}_i}$ for simplicity) — the origin of \mathcal{M}_i in world frame \mathcal{W} — to the j th face of the environment polyhedron perpendicular with distance d_{ij} , then if we enforce the constraint

$$v_{\mathcal{M}_i}^s \bullet \hat{s}_{ij} = (J_{\mathcal{M}_i}^s \dot{\Theta})^\wedge p_{\mathcal{M}_i} \bullet \hat{s}_{ij} \leq d_{ij} \quad (21)$$

for every side of the environment polyhedron, $p_{\mathcal{M}_i}$ will never cross the boundary of the environment as long as this kinematics chain follows the velocity for much less than 1 second. Using a sphere with radius r_i to approximate the geometry size of module m_i , then the constraint

$$v_{\mathcal{M}_i}^s \bullet \hat{s}_{ij} = (J_{\mathcal{M}_i}^s \dot{\Theta})^\wedge p_{\mathcal{M}_i} \bullet \hat{s}_{ij} \leq d_{ij} - r_i \quad (22)$$

will ensure that the module body will always be inside the environment boundaries (Fig. 7a). Thus, applying constraint (22) to all modules in the kinematics chain will ensure the chain will stay inside the environment.

2) *Obstacle Avoidance*: It is hard to represent the collision-free space analytically in joint space due to the high-DoF of modular robotic systems. Here we propose an alternative. The obstacles can be approximated by a set of spheres using a sphere-tree construction algorithm [21]. Similar ideas have been explored in [9] and [22] which model this constraint as the distance between every sphere approximating the robot and every sphere approximating the obstacles. These constraints are not suitable for real-time applications of large systems due to numerical issues. In this paper, the obstacle avoidance requirement is modeled as linear constraints which are efficient to solve stably. For a module m_i in the kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}$, let \tilde{s}_{ij} be the unit direction vector from $p_{\mathcal{M}_i}$ to the center of the j th obstacle sphere o_j in world frame \mathcal{W} with radius r_j^o . Imaging a plane with \tilde{s}_{ij} as its normal vector and o'_j being the point of tangency to this sphere, then if we enforce the constraint

$$v_{\mathcal{M}_i}^s \bullet \tilde{s}_{ij} = (J_{\mathcal{M}_i}^s \dot{\Theta})^\wedge p_{\mathcal{M}_i} \bullet \tilde{s}_{ij} \leq \|o'_j - p_{\mathcal{M}_i}\| - r_i \quad (23)$$

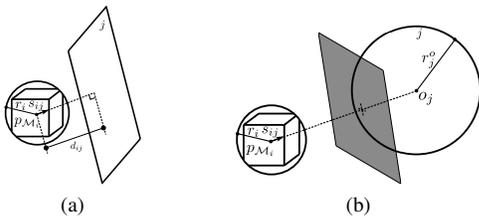


Fig. 7: (a) Environment boundary and (b) sphere obstacle avoidance.

in which $o'_j = o_j - r_j^o \tilde{s}_{ij}$ for every obstacle sphere, $p_{\mathcal{M}_i}$ will never touch an obstacle (Fig. 7b).³ In order to enable the system to safely navigate the environment, we can apply this constraint for every module.

C. Integrated Control and Motion Planning

With the control law in Sec. IV-A and motion constraints in Sec. IV-B, we can formalize the control and motion planning problem for multiple kinematics chains $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_i$ $i = 1, 2, \dots, \alpha$ as the following quadratic program with linear constraints

$$\begin{aligned} & \text{minimize} && \frac{1}{2} \dot{\Theta}^\top \dot{\Theta} \\ & \text{subject to} && \mathbf{J}\mathbf{P} = \tilde{\mathbf{V}} + \mathbf{K}(\tilde{\mathbf{P}} - \mathbf{P}) \\ & && \frac{\Theta_{\min} - \Theta}{\Delta t} \leq \dot{\Theta} \leq \frac{\Theta_{\max} - \Theta}{\Delta t} \\ & && \dot{\Theta}_{\min} \leq \dot{\Theta} \leq \dot{\Theta}_{\max} \\ & && (J_{\mathcal{M}_i}^s \dot{\Theta})^\wedge p_{\mathcal{M}_i} \bullet \hat{s}_{ij} \leq d_{ij} - r_i \\ & && \forall (\mathcal{M}_i, f_j) \in V_K \times F \\ & && (J_{\mathcal{M}_i}^s \dot{\Theta})^\wedge p_{\mathcal{M}_i} \bullet \tilde{s}_{ik} \geq r_i + r_k^o \\ & && \forall (\mathcal{M}_i, S_k) \in V_K \times \mathbf{S} \end{aligned} \quad (24)$$

in which F is the set of all faces of environment polyhedron and f_j is the j th face, and \mathbf{S} is the set of all spheres approximating the environment obstacles and S_k is the k th sphere in \mathbf{S} . By solving this quadratic program, the minimum norm solution that satisfies the hardware limits, control requirement, and motion constraints can be obtained for the current time step given the current state of every kinematics chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_i$ $i = 1, 2, \dots, \alpha$, the desired velocity, and the position of the origin of each frame \mathcal{F}_i .

D. Iterative Algorithm for Modular Robots

The set of module graph \mathbf{G}_m described in Sec. III-A and the twist set ξ described in III-B associated with all the joints in different type of modules are computed and stored. For a modular robot configuration G , assuming the base module \tilde{m} and how it is attached to the world frame \mathcal{W} as well as the motion goals $\tilde{p}_{\mathcal{F}_1}, \tilde{p}_{\mathcal{F}_2}, \dots, \tilde{p}_{\mathcal{F}_\alpha}$ for frame $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\alpha$ respectively are known, the set of all faces of the environment polyhedron is F and the set of all spheres approximating environmental obstacles is \mathbf{S} , the full algorithm framework is shown in Algorithm 1 with following functions:

- *BFS*($G, \mathbf{G}_m, \tilde{m}$): Traverse a modular robotic configuration G in breadth-first-search order starting from \tilde{m} to construct the kinematics graph $G_K = (V_K, E_K)$;
- *GetChain*(G_K, \mathcal{F}): Return the kinematics chain from \mathcal{W} to \mathcal{F} in G_K ;
- *SolveQP*($G_K, \mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\alpha, \tilde{\mathbf{P}}(t), \tilde{\mathbf{V}}(t), \mathbf{P}, \mathbf{K}, \Delta t$): Construct and try to solve the quadratic program described in Eq. (24). If failed to solve this program, then return Null as an invalid solution.

³This is a correction of the published conference paper on IEEE Xplore.

Algorithm 1: Control and Motion Planning

Input: ξ , \mathbf{G}_m , \bar{m} , $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\alpha$,
 $\{\tilde{p}_{\mathcal{F}_i}(t) | 0 \leq t \leq T, i = 1, 2, \dots, \alpha\}$, F , \mathbf{S}

Output: result

```

1  $G_K = \text{BFS}(G, \mathbf{G}_m, \bar{m})$ ;
2  $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_i = \text{GetChain}(G_K, \mathcal{F}_i)$ ,  $i \in [1, \alpha]$ ;
3 Initialize  $\Theta$ ;
4 Initialize  $\mathbf{K}$  and  $\Delta t$ ;
5  $t \leftarrow 0$ ;
6 while  $\sum_{i=1}^{\alpha} \|p_{\mathcal{F}_i} - \tilde{p}_{\mathcal{F}_i}(T)\| \geq \epsilon$  do
7   Compute  $\hat{s}_{ij} \forall (\mathcal{M}_i, f_j) \in V_K \times F$ ;
8   Compute  $\hat{s}_{ik} \forall (\mathcal{M}_i, S_k) \in V_K \times \mathbf{S}$ ;
9    $\hat{\Theta} = \text{SolveQP}(G_K, \mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_\alpha$ ,
    $\hat{\mathbf{P}}(t), \hat{\mathbf{V}}(t), \mathbf{P}, \mathbf{K}, \Delta t)$ ;
10  if  $\hat{\Theta} = \text{Null}$  then
11    return result  $\leftarrow$  False;
12  end
13  Publish  $\hat{\Theta}$  to the system;
14   $t \leftarrow t + \Delta t$ ;
15 end
16 return result  $\leftarrow$  True;

```

After initializing all the parameters, compute the unit direction vector \hat{s}_{ij} between every $\mathcal{M}_i \in V_K$ and every face of the environment $f_j \in F$, and also compute the unit direction vector \hat{s}_{ik} between every $\mathcal{M}_i \in V_K$ and every obstacle sphere $S_k \in \mathbf{S}$. If there is no valid solution, the program should stop, or the program will continue until $p_{\mathcal{F}}$ is close enough to the destination $\tilde{p}_{\mathcal{F}}(T)$. If the trajectory $\tilde{p}_{\mathcal{F}}(t)$ is not specified and only $\tilde{p}_{\mathcal{F}}(T)$ where $T \rightarrow \infty$ is given, then this algorithm can automatically find a path for modules to navigate the environment.

V. EXPERIMENTS

The approach is verified by a couple of experiments. We first show that the framework is able to execute a motion task with guaranteed tracking and navigation performance, frame boundary constraint, and obstacle avoidance. Then the framework is validated on SMORES-EP which has more DoFs to show its universality. At last, a motion task with two chains involved is executed with our framework.

1) *CKBot Chain*: A configuration with four CKBot UBar modules is built in Fig. 8a. The base module $\bar{m} = m_1$ is attached to the world frame \mathcal{W} and \mathcal{F} is attached to connector \mathcal{T} of module m_4 . A virtual frame boundary is next to the right side of the base. The task is to control $p_{\mathcal{F}}$ to follow a given trajectory to the position shown in Fig. 8d. Another experiment setup with five CKBot UBar modules is shown in Fig. 9a. The black sphere is an obstacle, the base module $\bar{m} = m_1$ and frame \mathcal{F} is attached to connector \mathcal{T} of module m_5 . Two tasks are executed: control $p_{\mathcal{F}}$ to follow a given trajectory and control $p_{\mathcal{F}}$ to approach a specified destination with the final position of $p_{\mathcal{F}}$ as shown in Fig. 9d. The control loop runs

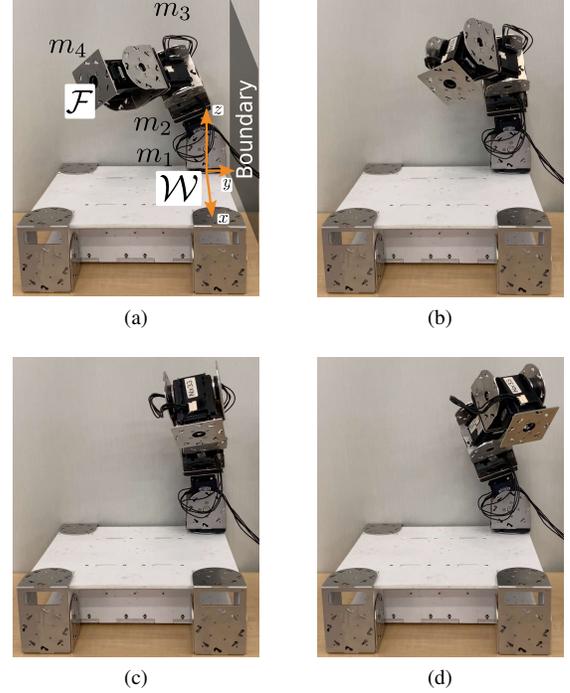


Fig. 8: Control $p_{\mathcal{F}}$ to follow a given trajectory along $+y$ -axis of \mathcal{W} by 15 cm from initial pose (a) to final pose (d): All the modules have to be on the left side of the boundary. m_1, m_2 , and m_3 have to approach the boundary first (b) and then move away from the boundary (c) to finish the task.

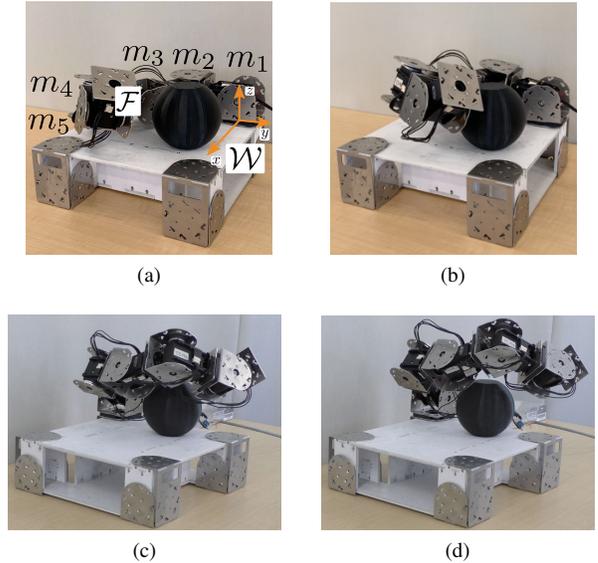


Fig. 9: Control $p_{\mathcal{F}}$ from its initial pose (a) to its final pose (d) by both following a given trajectory along $+y$ -axis of \mathcal{W} by 15 cm and navigating to the destination directly. The modules have to move around the sphere obstacle while executing these two tasks.

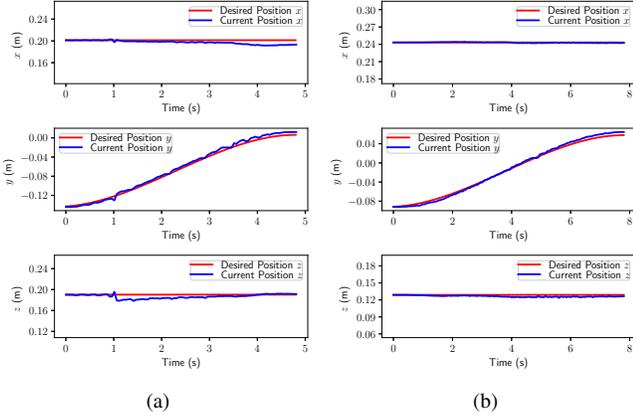


Fig. 10: The motion of $p_{\mathcal{F}}$ for (a) the four-module task and (b) the five-module trajectory following task.

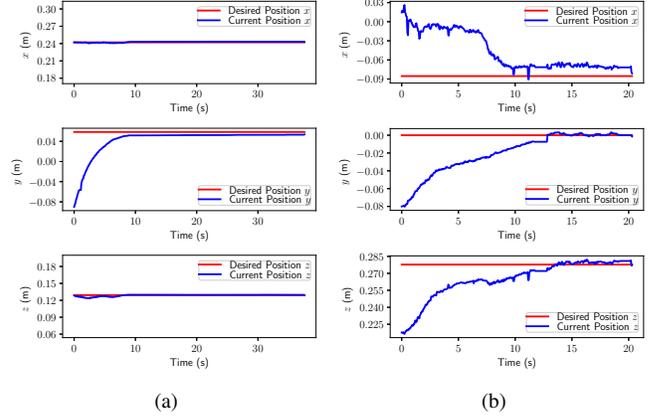


Fig. 12: The motion of $p_{\mathcal{F}}$ for (a) the CKBot five-module destination navigation task and (b) the SMORES-EP four-module chain destination navigation task.

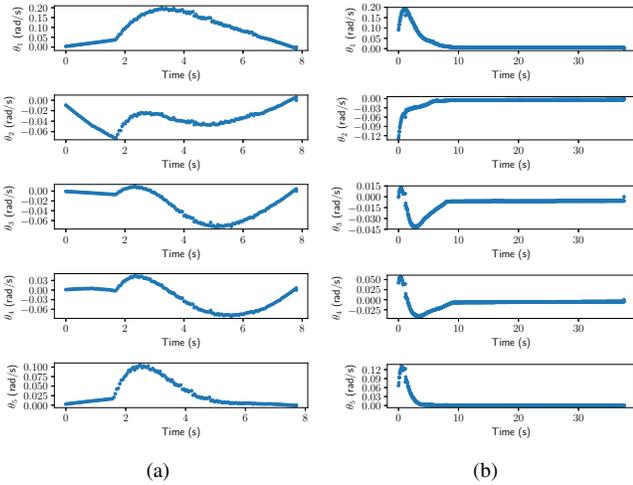


Fig. 11: The control input $\dot{\Theta}$ for five-module chain experiment: (a) trajectory following task and (b) destination navigation task.

at 20 Hz with gain $K = \text{diag}(1, 1, 1)$. Fig. 10 and Fig. 12a shows $p_{\mathcal{F}}(t)$ and $\hat{p}_{\mathcal{F}}$ of these three tests which demonstrates tracking and navigation performance. The velocity commands for all modules in these two five-module demonstrations are shown in Fig. 11 and all commands are within the limitation of each module. Modules moves more aggressively at the beginning when executing destination navigation task in order to approach the destination as soon as possible.

2) *SMORES Chain*: The experiment setup with four SMORES-EP modules is shown in Fig. 13a. The base module $\bar{m} = m_1$ is fixed to the world frame \mathcal{W} and frame \mathcal{F} is attached to connector \mathcal{T} of module m_4 . This system has 16 DoFs and the task is to control $p_{\mathcal{F}}$ to navigate to a specified destination shown in Fig. 13b. The control loop is running at 20 Hz and the gain $K = \text{diag}(0.5, 0.5, 0.5)$. The experiment result $p_{\mathcal{F}}(t)$ is shown in Fig. 12b. The position

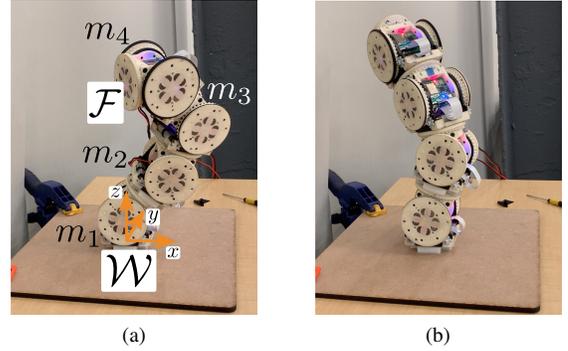


Fig. 13: Control a chain of SMORES-EP modules to navigate from its initial position (a) to a goal position (b). This chain is constructed by four modules with 16 DoFs.

sensors installed in SMORES-EP modules are customizable potentiometers using paints [23]. These low-cost sensors with a modified Kalman filter for nonlinear systems are used to provide position information of each DoF. Due to the limitations of the hardware, there are some noise in this experiment.

3) *CKBot Branch*: A configuration with nine CKBot UBar modules is shown in Fig. 14a. The base module $\bar{m} = m_1$ is fixed to the world frame \mathcal{W} . Frame \mathcal{F}_1 is attached to connector \mathcal{T} of module m_6 and frame \mathcal{F}_2 is attached to connector \mathcal{T} of module m_9 . Chain $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_1$ and $G_K : \mathcal{W} \rightsquigarrow \mathcal{F}_2$ have common parts composed by module $m_1, m_2,$ and m_3 . The task is to control $p_{\mathcal{F}_1}$ and $p_{\mathcal{F}_2}$ to follow trajectories respectively to the pose shown in Fig. 14d. The control loop is running at 20 Hz and the gain is $\text{diag}(0.1, 0.1, 0.1)$ for both motion goals. The tracking performance is shown in Fig. 15a and Fig. 15b.

VI. CONCLUSION

A new approach to simultaneous control and motion planning for arbitrary configurations of modular robots based on an autonomous kinematics modeling method is presented in

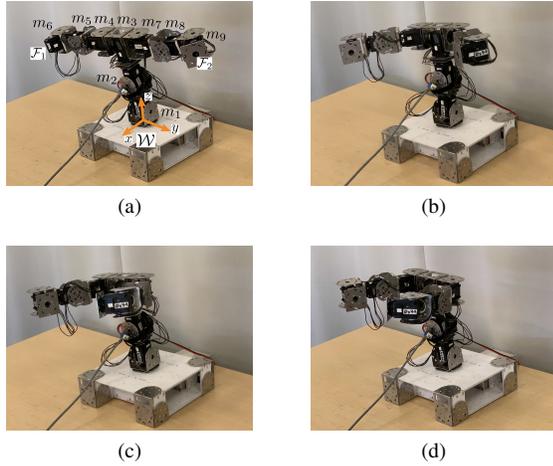


Fig. 14: Control $p_{\mathcal{F}_1}$ and $p_{\mathcal{F}_2}$ to follow two given trajectories respectively from initial pose (a) to final pose (d). Module m_1 , m_2 , and m_3 initially have to move backward (b) and then move forward (c) in order to control $p_{\mathcal{F}_1}$ and $p_{\mathcal{F}_2}$ to follow their trajectories.

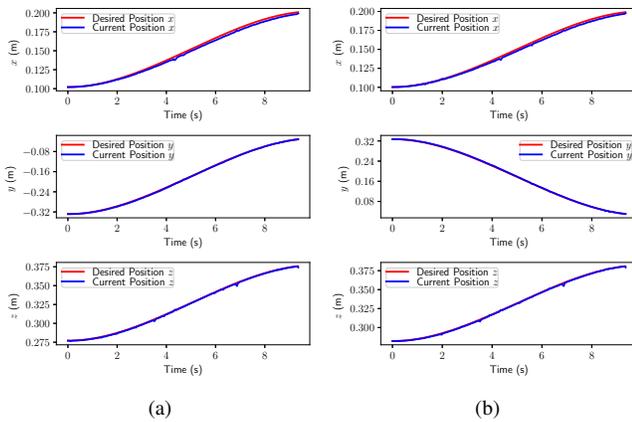


Fig. 15: (a) Tracking result for $p_{\mathcal{F}_1}$ and (b) tracking result for $p_{\mathcal{F}_2}$.

this paper. The control requirement and the motion planning constraints can be incorporated into a linearly constrained quadratic programming problem that can be solved efficiently. Multiple strongly coupled motion goals can be handled easily and some hardware demonstration and experiment results are provided. Future work includes objective functions exploration for different tasks and dynamics constraints for the system.

REFERENCES

- [1] M. Yim, W. M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems: Grand challenges of robotics," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 43–52, 2007.
- [2] M. Yim, D. G. Duff, and K. D. Roufas, "Polybot: a modular reconfigurable robot," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, April 2000, pp. 514–520 vol.1.

- [3] M. Yim, P. White, M. Park, and J. Sastra, "Modular self-reconfigurable robots," *Encyclopedia of Complexity and Systems Science*, pp. 5618–5631, 2009.
- [4] C. Liu, M. Whitzer, and M. Yim, "A distributed reconfiguration planning algorithm for modular robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4231–4238, Oct 2019.
- [5] M. Yim, "A reconfigurable modular robot with many modes of locomotion," in *Proceedings. JSME International Conference on Advanced Mechatronics*, Tokyo, Japan, 1993, pp. 283–288.
- [6] R. Fitch and Z. Butler, "Million module march: Scalable locomotion for large self-reconfiguring robots," *The International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 331–343, 2008.
- [7] C. Liu, S. Yu, and M. Yim, "Motion Planning for Variable Topology Truss Modular Robot," in *Proceedings of Robotics: Science and Systems*, Corvallis, Oregon, USA, July 2020.
- [8] S. K. Agrawal, L. Kissner, and M. Yim, "Joint solutions of many degrees-of-freedom systems using dextrous workspaces," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 3, May 2001, pp. 2480–2485 vol.3.
- [9] M. Fromherz, T. Hogg, Y. Shang, and W. Jackson, "Modular robot control and continuous constraint satisfaction," in *Proceedings of IJCAI Workshop on Modelling and Solving Problems with Constraints*, Seattle, WA, 2001, pp. 47–56.
- [10] Yunong Zhang, S. S. Ge, and Tong Heng Lee, "A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 5, pp. 2126–2132, Oct 2004.
- [11] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.
- [12] K. Shankar, J. W. Burdick, and N. H. Hudson, "A quadratic programming approach to quasi-static whole-body manipulation," *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*, pp. 553–570, 2015.
- [13] W. W. Melek and A. A. Goldenberg, "Neurofuzzy control of modular and reconfigurable robots," *IEEE/ASME Transactions on Mechatronics*, vol. 8, no. 3, pp. 381–389, Sep. 2003.
- [14] W. Zhu, T. Lamarche, E. Dupuis, D. Jameux, P. Barnard, and G. Liu, "Precision control of modular robot manipulators: The vdc approach with embedded fpga," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1162–1179, Oct 2013.
- [15] G. Liu, S. Abdul, and A. A. Goldenberg, "Distributed control of modular and reconfigurable robot with torque sensing," *Robotica*, vol. 26, no. 1, p. 75–84, 2008.
- [16] A. Giusti and M. Althoff, "Automatic centralized controller design for modular and reconfigurable robot manipulators," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 3268–3275.
- [17] N. M. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, April 1996, pp. 113–120 vol.1.
- [18] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Computer Science Dept., Iowa State Univ, Tech. Rep., 1998.
- [19] C. Liu and M. Yim, "Configuration recognition with distributed information for modular robots," in *IFRR International Symposium on Robotics Research*, 2017.
- [20] R. M. Murry, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.
- [21] G. Bradshaw and C. O'Sullivan, "Adaptive medial-axis approximation for sphere-tree construction," *ACM Trans. Graph.*, vol. 23, no. 1, pp. 1–26, Jan. 2004.
- [22] Y. Zhao, H. Lin, and M. Tomizuka, "Efficient trajectory optimization for robot motion planning," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov 2018, pp. 260–265.
- [23] T. Tosun, D. Edgar, C. Liu, T. Tsabedze, and M. Yim, "Paintpots: Low cost, accurate, highly customizable potentiometers for position sensing," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1212–1218.