Efficient Reconfiguration of Lattice-Based Modular Robots

Greg Aloupis[a], Nadia Benbernou[b], Mirela Damian[c], Erik D. Demaine[d], Robin Flatland[e,*], John Iacono[f], Stefanie Wuhrer[g]

[a]*Département d'Informatique, Université Libre de Bruxelles, CP212, Boulevard du Triomphe, 1050 Bruxelles, Belgium*
[b]*Mathematics Department, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, Massachusetts 02139 USA*
[c]*Computer Science Department, Villanova University, 800 Lancaster Avenue, Villanova, PA, 19085 USA*
[d]*Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, Massachusetts 02139 USA*
[e]*Computer Science Department, Siena College, 515 Loudon Road, Loudonville, New York, 12211 USA*
[f]*Computer Science and Engineering, Polytechnic Institute of New York University, Brooklyn, New York, 11201 USA*
[g]*Computer Science Department, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6 Canada*

Abstract

Modular robots consist of many identical *units* (or *atoms*) that can attach together and perform local motions. By combining such motions, one can achieve a reconfiguration of the global shape of a robot. The term *modular* comes from the idea of grouping together a fixed number of atoms into a *metamodule*, which behaves as a larger individual component. Recently, a fair amount of research has focused on algorithms for universal reconfiguration using *Crystalline* and *Telecube* metamodules, which use expanding/contracting cubical atoms.

From an algorithmic perspective, this work has achieved some of the best asymptotic reconfiguration times under a variety of different physical models. In

*Corresponding author

Email addresses: aloupis.greg@gmail.com (Greg Aloupis), nbenbern@MIT.EDU (Nadia Benbernou), mirela.damian@villanova.edu (Mirela Damian), edemaine@mit.edu (Erik D. Demaine), flatland@siena.edu (Robin Flatland), jiacono@poly.edu (John Iacono), swuhrer@scs.carleton.ca (Stefanie Wuhrer)

this paper we show that these results extend to other types of modular robots, thus establishing improved upper bounds on their reconfiguration times. We describe a generic class of modular robots, and we prove that any robot meeting the generic class requirements can simulate the operation of a Crystalline atom by forming a six-arm structure. Previous reconfiguration bounds thus transfer automatically by substituting the six-arm structures for the Crystalline atoms. We also discuss four prototyped robots that satisfy the generic class requirements: M-TRAN, SuperBot, Molecube, and RoomBot.

Keywords: self-reconfiguring modular robots, modular robot reconfiguration algorithms, Crystalline atoms, cubical units, lattice-based modular robots

---

1. Introduction

A self-reconfiguring modular robot consists of a large number of independent *units*, or *atoms*, that can arrange themselves into a structure best suited for a given environment or task. For example, a robot may reconfigure into a thin linear shape to facilitate passage through a narrow tunnel, transform into an emergency structure such as a bridge, or surround and manipulate objects. Because modular robots comprise groups of identical atoms, they are also more easily repaired, by replacing damaged atoms with functional ones. Such robots are well-suited for working in unknown and remote environments.

A variety of atom types have been designed and prototyped in the robotics community, differing in shape and in the operations they perform. We focus here on lattice-based modular robots in which atoms are arranged on a regular grid. Examples of prototyped atoms include Crystalline [4], M-TRAN [11], Molecube [25], SuperBot [19, 6], and RoomBot [20]. For a comprehensive list, see [14, 23]. Atoms are equipped with mechanisms that allow them to attach/detach to/from neighboring atoms, and motion is typically achieved through the activation of one or more revolute or prismatic joints.

One of the algorithmic challenges for these modular systems is to determine efficient sequences of atom operations that transform a robot from one configuration to another. A typical requirement is that the atoms maintain connectivity at all times. As observed in [15], difficulties can arise from blocking constraints, such as when an atom is unable to directly move into an adjacent empty position of the lattice because it is blocked by tightly packed neighboring atoms. As a consequence of such constraints, it is possible that for certain configurations of a robot, no atom can move. This was demonstrated in [15] for hexagonal atoms. Certain

atom types require a sufficient number of neighboring atoms to move non-trivially. For example, a $1 \times n$ linear configuration of Crystalline atoms is not universally reconfigurable.

To address some of these difficulties, the concept of *metamodules* was introduced in [15, 10]. A metamodule is a small collection of atoms that behave as a single unit. Rather than specifying robots at the atom level, they are specified in terms of metamodules on a lower resolution lattice. These atoms combine to produce a synergistic effect, so that a metamodule has more freedom of movement than any individual atom. It is often the case that metamodules are sparsely constructed, which enables them to pass very close to (or in some sense, through) each other, without the type of blocking constraints mentioned previously. Throughout this paper, we will refer to robots that have $\Theta(n)$ atoms and metamodules. That is, we are only interested in metamodules consisting of a constant number of atoms.

Nguyen et al. [15] proposed metamodules consisting of 36 hexagonal atoms arranged along the boundary of a larger hexagonal region with an empty interior. They provided an algorithm to transform between any two "fat" robot configurations in $O(n)$ time. Prevas et al. [16] used metamodules consisting of 8 I-Cube atoms and 16 links to achieve an $O(n^2)$ time universal reconfiguration algorithm. Recently, there has been a fair amount of algorithmic research on universal reconfiguration using metamodules of expanding/contracting cubical atoms. The two main prototypes considered have been Crystalline and Telecube atoms, which are similar enough that we will henceforth refer only to the former. Crystalline metamodules are $k \times k \times k$ arrangements of atoms, where k is a small constant that varies depending on the situation. Specifically, the size of a metamodule depends on various factors, including the assumed physical capabilities of each atom.

In the weakest and most realistic physical model, Crystalline atoms have constant strength (i.e., they can push and pull a constant number of other atoms when expanding and contracting) and the maximum speed they can reach during reconfiguration is also bounded by a constant. Early reconfiguration algorithms that used this model include the "melt-grow" algorithm of Rus and Vona [18], and that of Vassilvitskii et al. [21], both of which reconfigure in $O(n^2)$ time. This was improved in [1] to linear time using metamodules of $2\times2\times2$ atoms, with a total of $O(n^2)$ atom operations (counting operations performed in parallel). Both bounds were shown to be worst-case optimal. The linear time algorithm also requires only constant memory per atom and local communication between atoms.

More physically capable atoms naturally allow for faster reconfiguration algorithms. For instance, the total number of atom operations can be reduced to $O(n)$ in a model that assumes that atoms have linear strength [2]. This means

that atoms can push and pull up to n other atoms when contracting and expanding. When constant strength is assumed but velocities are allowed to build up over time, reconfiguration is possible in $O(\sqrt{n})$ time in 2D, using 4×4 metamodules and the third dimension as an intermediate [17].

In the most physically capable model where atoms have linear strength and velocities can be instantly linear, any 2D reconfiguration is possible in $O(\log n)$ time using $O(n \log n)$ total operations [3]; the algorithm uses metamodules of size 4×4, but it is claimed that it can be reduced to 2×2. A straightforward (yet unpublished) extension of this result achieves the same asymptotic bound in 3D, using larger but still constant-sized cube-shaped metamodules.

To our knowledge, similar asymptotic bounds on universal reconfiguration for other lattice-based modular robots are not yet known. In this paper, we extend the Crystalline reconfiguration results to other lattice-based modular robots. Specifically, we describe a generic class of modular robots, and we prove that any robot meeting the generic class requirements can simulate the operation of a Crystalline atom by forming a structure called a 6-arm. We also discuss four prototyped robots that meet the generic class requirements: M-TRAN, SuperBot, Molecube, and RoomBot. By replacing Crystalline atoms with 6-arms in the $k \times k \times k$ metamodules used by existing Crystalline algorithms, the reconfiguration results immediately apply. Thus as in the previous work, reconfiguration assumes (and exploits) the existence of specifically constructed metamodules, and the reconfiguration bounds apply to robots composed of these metamodules. From an algorithmic perspective, the asymptotic universal reconfiguration times achieved here via the Crystalline algorithms using the 6-arm construction are the most efficient known for the M-TRAN, SuperBot, Molecube, and RoomBot robots.

The term "efficient" in this paper refers to the asymptotic time complexity of the reconfiguration algorithms. We point out that cost and physical limitations of prototyped atoms make the work here a theoretical contribution, rather than a practical one. The number of atoms in the 6-arm is 58, which renders the 6-arm construction and its operation impractical with existing hardware implementations of prototyped atoms. Our goal has been to establish that several prototypes have no fundamental geometric disadvantages compared to Crystaline atoms (metamodules). The limiting effects of torque, motor abilities, and gravity are not of primary concern here.

Our aim has been to establish a global simulation structure. The 6-arm metamodule is a general construction that can be applied to a variety of atom types. One would naturally expect that customizing a metamodule for each type of atom would lead to smaller constructions. For example, it has been shown that an 8

atom customized M-TRAN metamodule can simulate a 2D Crystalline atom [12].

Using one prototyped module to simulate others has gained popularity. A recent paper by Davey et al. [7] demonstrates that the SMORES robot can emulate (either exactly or approximately) other robots such as PolyBot [22], SuperBot, CONRO [5], and a system from Johns Hopkins University [13]. In other work, Dewey et al. [8] developed a universal reconfiguration algorithm for an abstract robot system in which metamodules can absorb adjacent metamodules into their interior and transfer them out again into adjacent metamodules or into empty lattice locations. The algorithm may be used with any atom type for which it is possible to form metamodules capable of these operations. They demonstrated how this can be done for three different atom types using customized constructions. Although they did not address the asymptotic worst-case time complexity of their reconfiguration algorithm, experimental results suggest that the time complexity is linear.

The remainder of this paper is organized as follows. In Section 2 we describe the functionality of the Crystalline atom. In Section 3 we describe a generic class of robots with properties sufficient to design a 6-arm. In Section 4 we discuss four prototyped robots that meet the generic class requirements. The 6-arm construction is described in Section 5, and then in Section 6 we prove that the 6-arm correctly simulates a Crystalline atom. In Section 7, we summarize the Crystalline reconfiguration results that apply (via the 6-arm structure) to all robots satisfying the generic class requirements. We conclude in Section 8 with some directions for future work.

2. Crystalline Atom Operations

Because we seek to simulate the operations of the Crystalline atom, we detail its functionality here. A Crystalline atom is a cubical device equipped with an expansion/contraction mechanism that allows it to extend each face out and retract it back. When extended, a face is twice as far from the cube's center, compared to its retracted distance. Each cube face has an attachment mechanism that allows it to attach to (or detach from) adjacent atoms. When groups of atoms perform these operations (expand, contract, attach, detach) in a coordinated way, the atoms move relative to one another, resulting in a reconfiguration of the robot (see Fig. 1).

3. Generic Model

In this section we describe a generic class of lattice-based modular robots which possess certain sufficient properties to design a 6-arm. Robots in this class

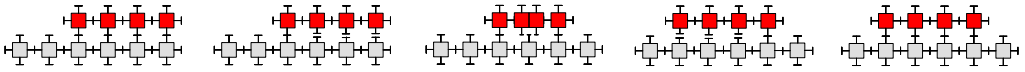

Figure 1: Crystalline robot reconfiguration (2D example). Atoms can attach to neighbors, and expand/contract their arms. Figure borrowed from [3].

must be capable of forming a generic *block* U satisfying the following three criteria:

1. **Structure.** U consists of two parts occupying two adjacent cells L, R on a 3D cubic lattice. For simplicity, we will use L to refer to both the lattice cell and the piece of U that lies in that lattice cell. This should be clear from the context. We assume that each lattice cell has dimensions $1\times1\times1$. Without loss of generality, for our descriptions we assume that R is on the right of L (see Fig. 2a).
2. **Attach/Detach Mechanisms.** Both L and R have attachment mechanisms that can be positioned so that all six y-parallel faces surrounding U contain an attachment, as shown in Fig. 2a. In this configuration, U can attach to an adjacent block along any of these faces. When the six attachment mechanisms are positioned in this way, we say the block is in a *straight* configuration. This is in contrast with the *bent* configuration, which we describe next.

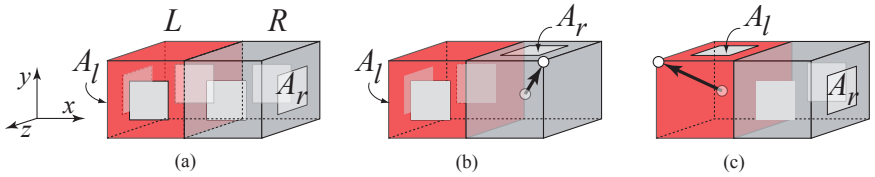

Figure 2: The generic block. (a) straight configuration, (b,c) bent configurations. Positions of attach/detach mechanisms are marked as small, lightly shaded squares.

3. **Rotational Motion.** Let A_r be the attachment mechanism on the right face of cell R, as labeled in Fig. 2a. U has a rotational degree of freedom that allows A_r to rotate to a top horizontal position, while the three attachments in cell L remain stationary, as shown in Fig. 2b. We will refer to the resulting configuration as *bent*.
   Let (a_x, a_y, a_z) be a unit vector parallel to the rotation axis about which A_r rotates. To ensure a compact rotational motion, we impose four requirements on R.

3.a) The rotation axis passes through the center of cell R.
3.b) In the coordinate system of Fig. 2, we have that $a_x, a_y, a_z \geq 0$.
3.c) The rotation of R that takes it from straight to bent position is a counterclockwise rotation of at most $180°$.
3.d) The rotation maps the center point of cell R's right face to the center point of R's top face.

This last requirement of the rotational motion implies that $a_x = a_y$. To see why, consider a coordinate system centered in cell R. In this coordinate system, observe that points $p_r = (\frac{1}{2}, 0, 0)$ and $p_t = (0, \frac{1}{2}, 0)$ (i.e., the center of the right and top faces of cell R) must lie in a common plane perpendicular to the rotation axis, since the rotation transforms one point into the other. This means that vector $(p_r - p_t)$ and the rotation axis are orthogonal, and thus their dot product is zero: $(\frac{1}{2}, -\frac{1}{2}, 0) \cdot (a_x, a_y, a_z) = 0$, and so $a_x = a_y$. This observation will be useful in proving that the generic 6-arm correctly simulates the Crystalline atom.

In Fig 2b, a corner of the rotating cell is marked to indicate the signed orientation of (a_x, a_y, a_z). Specifically, the corresponding components of (a_x, a_y, a_z) and the vector directed from the center of the rotating cell to the marked corner have the same signs.

Depending on the prototyped atoms used to construct a block, the final position of the other two attachments on R may vary. In fact, their locations in the bent configuration are not relevant to our 6-arm construction. This is why we purposely omit marking the other two attachment mechanisms on R in Fig. 2b.

The intuition here is that any structure attached to A_r rotates along with it and assumes a new position on top of R. We will not allow a bend to occur while other components are attached to other faces of R. Structures attached to L do not move.

L is a mirror reflection of R, thus capable of mirroring the rotational motion and moving A_ℓ from left vertical to top horizontal, as shown in Fig. 2c. In particular, the rotation axis for the left half is parallel to $(-a_x, a_y, a_z)$ and the rotation from straight to bent is clockwise.

4. Prototyped Blocks

4.1. SuperBot and M-TRAN blocks

The SuperBot atom [19, 6] and its predecessor M-TRAN atom [11] consist of two identical elements connected by a link. An element can be viewed as a

half-cube glued to a half cylinder, as depicted in Fig. 3a; the bounding box of an element is a unit cube. Each of the atom's six flat faces (all vertical in Fig. 3a) is equipped with an attachment mechanism. In addition, each element can rotate independently $\pm 90^\circ$ about a center axis perpendicular to the top/bottom of its half cylinder. We note that the M-TRAN atom has one additional rotational degree of freedom and the SuperBot atom has two, but these are unnecessary to meet the requirements of a generic block.

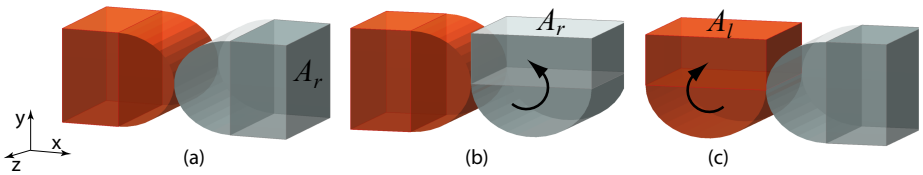

Figure 3: The M-TRAN block. (a) straight configuration, (b,c) bent configurations.

We show now that the SuperBot and M-TRAN atoms satisfy the generic block requirements of Section 3. Clearly they satisfy the criteria for structure and attach/detach mechanisms. For the rotational motion, they can go from straight to bent position by rotating their right element 90° counterclockwise, and similarly by rotating their left element 90° clockwise, as shown in Figs. 3b,c. With the rotation axis of the right element passing through its center and being parallel to $(0, 0, 1)$ (as oriented in Fig. 3) and the left atom being a mirror image of the right, the SuperBot and M-TRAN atoms satisfy the rotational requirements 3.a-3.d.

4.2. Molecube block

A Molecube atom [24, 25, 26] is a cubical structure (with rounded corners) equipped with a magnetic attachment mechanism on each of its faces. A diagonal cut extending from the top to the bottom face separates the cube into two triangular prisms (see Fig. 4a). This allows the atom to rotate its two halves about a symmetry axis orthogonal to the cut plane and passing through the center of the cube.

We define a Molecube *block* as two atoms attached face-to-face, as shown in Fig. 4. The attachment is such that the right atom's axis of rotation is parallel to $(1, 1, 1)/\sqrt{3}$, and the left atom's is parallel to $(-1, 1, 1)/\sqrt{3}$. We now show that a Molecube block satisfies the generic block requirements of Section 3. Clearly it satisfies the criteria for structure and attach/detach mechanisms. For the rotational motion, the Molecube block can go from straight to bent position by rotating the

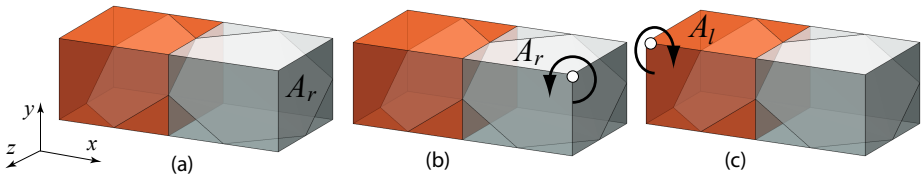

Figure 4: (a) Molecube block. (a) straight configuration, (b,c) bent configurations.

top half of the right atom by $120°$ counterclockwise, which brings the right face to the top. Similarly, the block can also go to bent position by rotating the top half of the left atom by $120°$ clockwise, which brings the left face to the top. With the rotation axis of the right atom passing through its center and being parallel to $(1, 1, 1)/\sqrt{3}$ and the left atom being a mirror image of the right, the Molecube block satisfies the rotational requirements 3.a-3.d.

4.3. RoomBot block

The RoomBot atom [20] (see Fig. 5) is designed to be suitable for reconfigurable furniture. For our purposes, its functionality is similar to that of the Molecube block in Section 4.2. The RoomBot atom consists of two spherical elements. Each spherical element is sliced into two half-spheres by a diagonal plane of the lattice cell containing the sphere. Each half-sphere can rotate independently about the diametrical axis perpendicular to the cut. When configured as in Fig 5, the rotation axis of the top left half-sphere is parallel to the rotation axis of the left atom of the Molecube block from Fig. 4, and similarly for the right counterpart. Like the Molecube block, the RoomBot atom has ten attachment mechanisms, allowing it to connect to any atom sharing an adjacent lattice face. These common characteristics of the Roombot and Molecube atoms enable us to apply the discussion from Section 4.2 directly to the Roombot, establishing that the RoomBot atom satisfies the generic block requirements of Section 3.

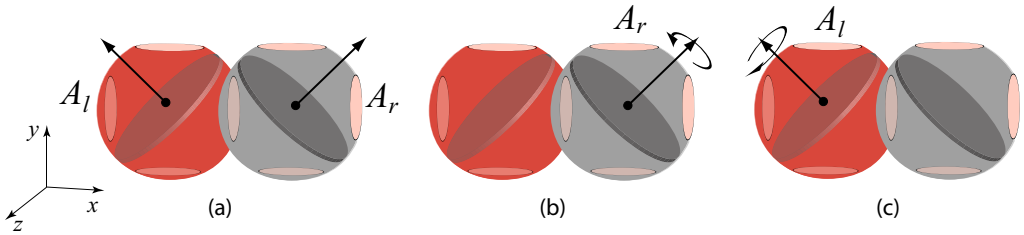

Figure 5: (a) RoomBot block. (a) straight configuration, (b,c) bent configurations.

5. Design of the Generic 6-arm

We now show how to construct a 6-arm out of generic blocks. Recall that a 6-arm is meant to simulate the four Crystalline atom operations: expand, contract, attach and detach. Our structure (see Fig. 6)[1] consists of six arms connected to a common $2\times2\times2$ *center piece*.

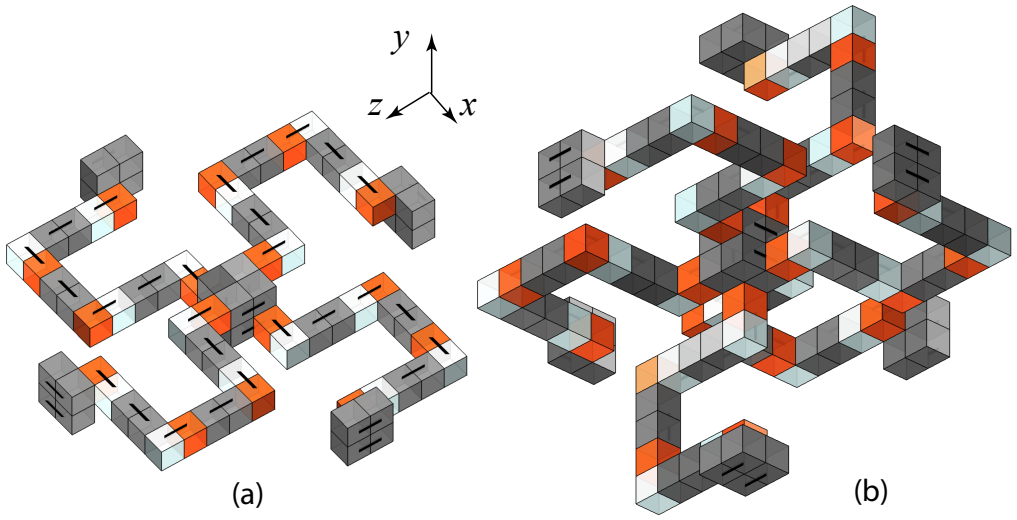

Figure 6: A 6-arm metamodule: (a) The four "horizontal" arms ; (b) entire structure from a different viewpoint.

An arm can be viewed as a chain of blocks with four revolute *joints*, henceforth referred to simply as joints. A joint is a block which we allow to straighten and bend. In Fig. 6, each block is marked with a black segment that connects its two cubes. Blocks that serve as joints are colored orange and white, in accordance with Figure 2. Blocks that remain straight throughout are gray. An arm has two functional states: expanded and contracted (like a Crystalline atom face). In an expanded state, all joints (and thus all blocks) of an arm are in a straight position (Fig. 7b). In its contracted state, an arm forms a Π-shaped bend (Fig. 7a). The tip of each arm occupies 2×2 cells and has an attachment mechanism that allows it to connect to an adjacent tip of another 6-arm. An arm constructed out of Molecube

[1]For animations of the Molecube 6-arm, the reader may visit:
`http://www.csc.villanova.edu/~mdamian/6arm`

blocks is shown in Fig. 7 and one constructed out of M-TRAN blocks is shown in Fig. 8.

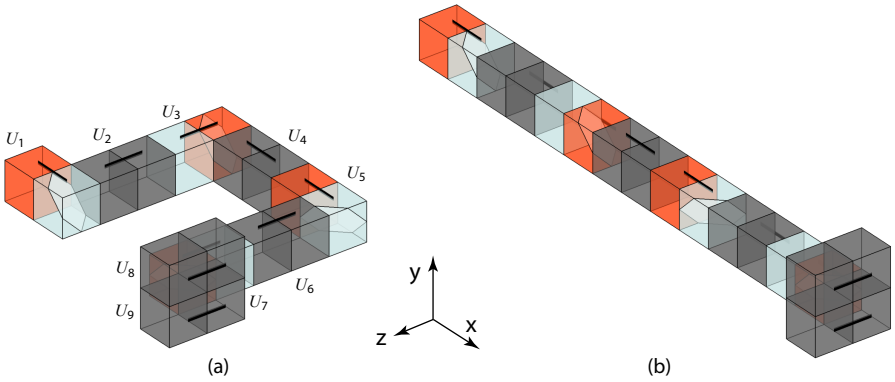

Figure 7: (a) Contracted Molecube arm. U_1 attaches to the center piece of the 6-arm; (b) Extended arm.

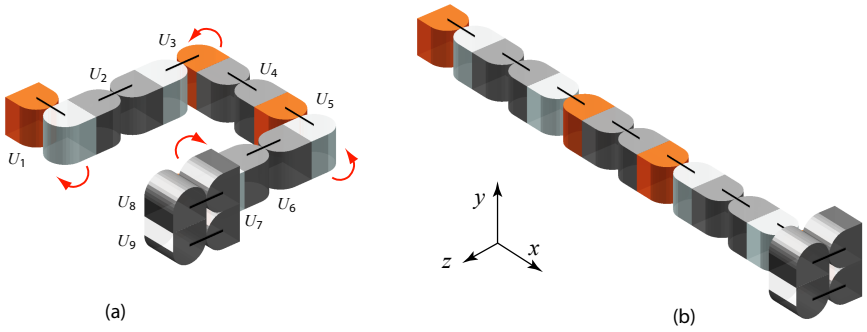

Figure 8: (a) Contracted M-TRAN arm. U_1 attaches to the center piece of the 6-arm; (b) Extended arm.

Observe that when an arm is contracted, the distance from the tip to the center is 8 lattice units (7 along the arm, and 1 in the center piece). By simultaneously having joints U_1, U_3, U_5, and U_7 straighten, the arm expands, doubling the distance from tip to center.

An arm expands by rotating (straightening) its joints in a coordinated way. By performing the four joint rotations simultaneously, we ensure that the tip attachment moves parallel to a coordinate axis. This is discussed in Section 6.

Next we discuss the details of constructing an arm. Let the terms *right, left, top, bottom, front*, and *back* refer to the $+x$, $-x$, $+y$, $-y$, $+z$, $-z$ directions, respectively. For ease of presentation, we focus on the right arm which extends out in the positive x direction, as seen in Fig. 7. It is composed of nine blocks $U_1, \ldots, U_9$, of which four behave as joints, i.e., U_1, U_3, U_5, U_7. The first eight blocks rest in a horizontal plane, whereas U_9 attaches directly above U_8; the reason for this particular attachment is to perfectly align the arm tip with the center piece. The center piece comprises four blocks arranged in a $2\times2\times2$ configuration, as illustrated in Fig. 9. The four blocks are all in bent position, with the two right blocks oriented horizontally and the two left ones oriented vertically. The six attachment points for the arms are marked in Fig. 9b. Observe that the right arm attaches to the lower layer of the center piece, and thus by adding block U_9 above the main arm level the tip aligns with the center piece.

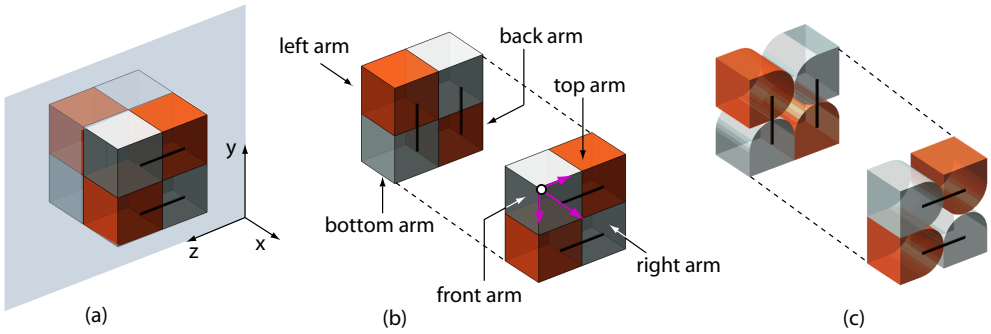

Figure 9: (a) The center piece consists of four blocks in a $2\times2\times2$ grid of cells. (b) Molecube; (c) M-TRAN; The left and right halves of the center pieces are shown separated, with dotted lines indicating how the two halves attach in 3D. Attachment points for the arms are marked.

The four arm joints are oriented so that the rotation axes of U_1 and U_3 are parallel, as are the rotation axes of U_5 and U_7. To see that these parallel alignments are possible, notice that U_1 in Fig. 7a is an instance of the block shown in Fig. 2b that has been rotated by 90° clockwise about a center x parallel axis; U_3 is an instance of Fig. 2c that has been rotated by 90° clockwise about a center x parallel axis, and then 90° clockwise about a center y parallel axis. It is straightforward to show that $R_x(-90)(a_x, a_y, a_z)^T = R_y(-90)R_x(-90)(-a_x, a_y, a_z)^T$ when $a_x = a_y$ (as specified by rotational requirement 3.d.), and so the two joints have parallel rotation vectors. U_5, U_7 are similar.

To avoid collisions in the 6-arm, we have designed two variations of the

generic arm. Fig. 10 illustrates the design of the right arm; we call this design a *type-A* arm. In a type-A arm, the rotation vector of joints J_1 and J_3 has components with signs $(+,+,-)$, and the rotation vector of J_5 and J_7 has components with signs $(+,-,+)$. During an extension of the arm, J_1, J_5 rotate clockwise and J_3, J_7 rotate counterclockwise. With this design, the arm sweeps downwards when extending and contracting. The amount of downward sweep varies depending on the rotation vector, and in the special case when the rotation vector is $(0,1,0)$ the arm stays horizontal with no downward sweep. For example, Fig. 10b shows the arm position after J_1 and J_5 rotate by -60° and J_3 and J_7 rotate by 60°, for the case when J_1, J_3 have rotation vector $(1,1,-1)/\sqrt{3}$ and J_5, J_7 have rotation vector $(1,-1,1)/\sqrt{3}$. Note that the tip of the arm has translated along the x-axis.

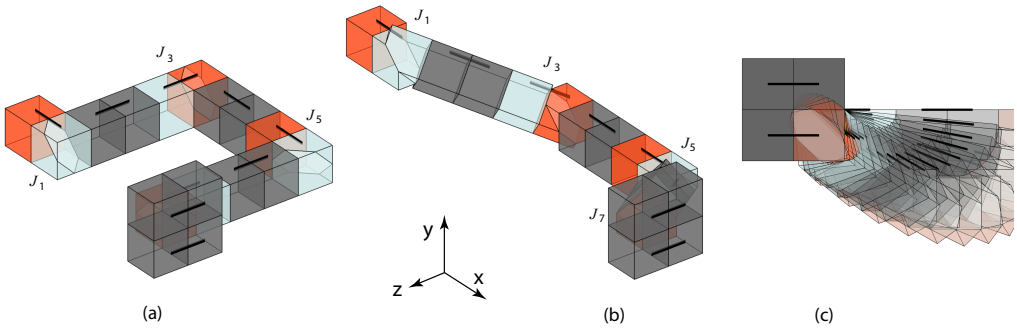

Figure 10: Type-A Molecube right arm: (a) Contracted state; (b) After J_1 and J_5 rotate by angle -60°, while J_3 and J_7 rotate by 60°; (c) Space swept by the arm through a rotation of 120° (view from $x = +\infty$).

An alternate design for the right arm is depicted in Fig. 11a; we will refer to this as a *type-B* arm. In the type-B arm, the rotation vector of joints J_1 and J_3 has components with signs $(+,-,-)$, and the rotation vector of J_5 and J_7 has components with signs $(+,+,+)$. The arm extends by rotating its joints in directions opposite to those of a type-A arm. With this design, the arm sweeps upwards when extending and contracting. For example, Fig. 11b shows the arm position after J_1 and J_5 rotate by 60° and J_3 and J_7 rotate by -60°, for the case when J_1, J_3 have rotation vector $(1,-1,-1)/\sqrt{3}$ and J_5, J_7 have rotation vector $(1,1,1)/\sqrt{3}$.

The main difference between the type-A and type-B arms is the space swept during extension/contraction (see Fig. 10c vs. 11c). We carefully design the 6-arm so that the spaces swept by the six arms are disjoint and therefore no collisions

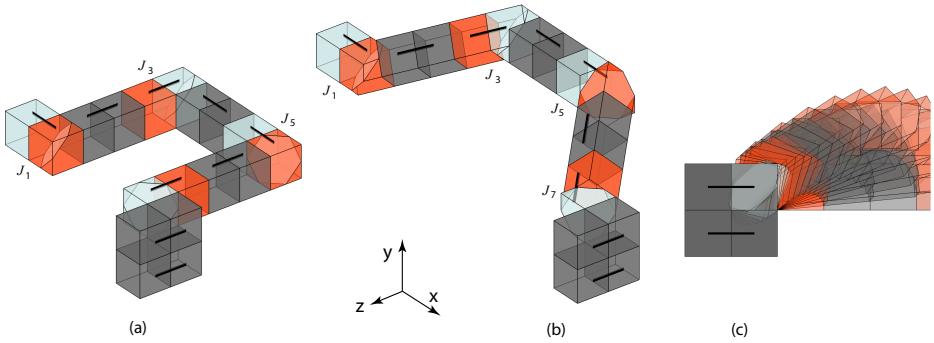

Figure 11: Type-B Molecube right arm: (a) Contracted state; (b) After J_1 and J_5 rotate by $60°$, while J_3 and J_7 rotate by $-60°$; (c) Space swept by the arm through a rotation of $120°$ (view from $x = +\infty$).

occur. This is achieved as follows: the right and back arms are type-A, and the top, bottom, front and left arms are type-B.

6. Simulation of Crystalline Atom Operations

In this section we prove that a 6-arm can simulate the operations of a Crystalline atom. Let $R_u(\theta)$ denote the matrix that rotates a point by θ degrees (counterclockwise) about an axis parallel to the unit vector $u = (u_x, u_y, u_z)$ with fixed point at the origin. (See Equation (A.1) in the Appendix.) Let $T(d_x, d_y, d_z)$ denote the translation matrix that translates a point by d_x, d_y and d_z units in the directions x, y and z, respectively. Let J_i denote the rotating half of U_i, for $i = 1, 3, 5, 7$, and let O_i denote the center of J_i. We denote the x-coordinate of O_i by $x(O_i)$, and similarly for the y and z-coordinates of O_i.

Lemma 1. *Throughout the expansion/contraction of the right arm, the component connecting J_3 and J_5 remains parallel to the x-axis and does not rotate about O_3O_5.*

Proof. Fix a coordinate system with origin at O_1. The matrix that determines the position and orientation of J_3 is given by

$$M_3 = R_a(-\theta)T(0, 0, -4)R_a(\theta) \tag{1}$$

where a is the rotation vector for J_1 and J_3 and θ is the joint rotation. For any axis a, the two rotations in the matrix product cancel each other out, and the net result

is that J_3 undergoes only translational motion. This suffices for our proof because the component between J_3 and J_5 moves rigidly with J_3. □

The next intuitive lemma shows that the joint centers maintain their ordering in the x direction.

Lemma 2. *Throughout the expansion/contraction of the right arm,* $x(O_1) \leq x(O_3) < x(O_5) \leq x(O_7)$.

Proof. Fix a coordinate system with origin at O_1. Since J_1 undergoes no translation, $x(O_1) = 0$ throughout the expansion motion. The value of $x(O_3)$ is given by the x-translation component of the matrix M_3 from Equation 1. With the rotational requirements (3.b) and (3.d) of Section 3 adjusted for the orientation of units U_1, U_3 of the right arm, we have $a_y \geq 0$ and $a_x = -a_z$. With the rotational requirement (3.c), we have $0 \leq \theta \leq 180$, and so $\sin(\theta) \geq 0$. Thus

$$\begin{aligned} x(O_3) &= -4(1-\cos(\theta))a_z a_x + 4\sin(\theta)a_y \\ &\geq -4(1-\cos(\theta))a_z a_x \text{ (as } a_y \geq 0 \text{ and } \sin(\theta) \geq 0) \\ &\geq 0 \text{ (as } a_z a_x \leq 0 \text{ and } 1-\cos(\theta) \geq 0) \end{aligned}$$

Thus we obtain the relation $x(O_1) \leq x(O_3)$. The value of $x(O_5)$ is given by the x-translation component of the matrix $M_3 T(4,0,0) R_b(-\theta)$, where b is the rotation vector for J_5 and J_7. This shows that $x(O_5) = x(O_3) + 4$. Finally, $x(O_7)$ is given by the x translation component of $M_3 T(4,0,0) R_b(-\theta) T(0,0,4) R_b(\theta)$. Simple calculations show that

$$\begin{aligned} x(O_7) &= -8a_z a_x + 8\cos(\theta)a_z a_x + 8\sin(\theta)a_y + 4 \\ &= 2 * x(O_3) + 4 \\ &= x(O_5) + x(O_3) \\ &\geq x(O_5) \quad (\text{as } x(O_3) \geq 0) \end{aligned}$$

□

Lemma 3. *During the expansion/contraction of the right arm, the attachment at its tip moves parallel to the* x*-axis.*

Proof. By Lemma 1, the midpoint m of the segment O_3O_5 undergoes a translation. Furthermore, by Lemma 2, this motion is x-monotone.

We can divide the arm into two halves, which are mirror images of each other, through a plane parallel to x=0, containing m. The second half can follow the motions of the first half symmetrically, in order to complete the desired motion.

As noted previously, the matrix that determines the orientation and position of J_7 relative to O_1 is

$$M_3T(4,0,0)R_b(-\theta)T(0,0,4)R_b(\theta) \tag{2}$$

The 6-arm construction and the requirements of the generic block ensure that the rotation vectors a and b of the right arm satisfy $(a_x, a_y, a_z) = (b_x, -b_y, -b_z)$. Then the product in Equation (2) is a translation matrix having zero y- and z-components and a positive x-component. □

Lemma 4. *An arm does not self-intersect during expansion/contraction.*

Proof. We prove the claim for the right arm. By Lemma 1, the segment O_3O_5 remains parallel to the x-axis throughout the arm motion, meaning that $x(O_3) + 4 = x(O_5)$. Along with Lemma 2, this implies that

$$x(O_1) + 4 \leq x(O_3) + 4 = x(O_5) \leq x(O_7)$$

Thus any pair of points in the set defined by the Cartesian product $\{O_1, O_3\} \times \{O_5, O_7\}$ are separated by at least 4 units in the x-dimension. This guarantees non self-intersection for the right arm. By symmetry, the arguments hold for the other arms as well. □

Lemma 5. *A 6-arm always remains inside the axis-aligned bounding box determined by the tips of its arms.*

Proof. Our claim is equivalent to saying that the tip of the right arm is always the point with strictly maximum $+x$ coordinate. For a coordinate system with origin centered in the center piece, the range of the right tip is $8 \leq x \leq 16$. It suffices to show that no other arm ever enters the $x \geq 8$ halfplane.

The length of any arm is 15. One end of the arm is anchored to the center piece and the tip is constrained to a coordinate axis, by Lemma 3. Furthermore, by Lemma 1, the segment O_3O_5 is constrained to be parallel to this axis.

Therefore the arm is confined within a cylindrical region aligned with the coordinate axis, with radius strictly less than 6. Thus the cylindrical region avoids the $x \geq 8$ halfplane. □

Define the *octant* of the right arm as the intersection of three halfplanes: ($x \geq 1$, $y \leq 1$, $z \leq 1$). This contains the right arm in its contracted position. Notice that two of the octant boundary halfplanes are tangent to the arm tip, and all three are tangent to the center piece. The origin of the octant is on a corner of the center piece, on the same face but diagonally across the connection of the right arm. The octant of each other arm can be defined symmetrically.

Lemma 6. *An arm remains within its own octant during expansion/contraction.*

Proof. We only focus on the expanding motion of the right arm in this proof. Other arms are handled symmetrically. First note that the right arm is type-A. Recall that, because it is horizontal, this means that some of its components will temporarily move downward when it expands and contracts.

Fix a coordinate system with origin centered in the center piece. By Lemma 2 and the fact that $x(O_1) = 2.5$, we know that the arm lies within the $x \geq 1$ halfplane.

Next we show that the right arm stays in the $z \leq 1$ halfplane. Consider an arm component between the rotating halves of two consecutive joints. Among all points on such a component, the point with the highest z-coordinate must lie on a joint (i.e, on one of the component endpoints). Thus, it suffices to focus only on the z-coordinates of joints (J_1, J_3, J_5, J_7). Note that for any point $p \in J_1$, $z(p) \leq 1$. Also observe that, for any point $p \in J_3$, the value of $z(p)$ is less than $z(O_3) + 1$, where $z(O_3)$ is the z-translation component of matrix M_3.

A straightforward argument detailed in Lemma 7 of the Appendix shows that the z-translation component of M_3 is never greater than 0 for our generic model. This implies that $z(p) < 1$ for any $p \in J_3$. By Lemma 1, the same holds true for $p \in J_5$. Finally, by Lemma 3 and the fact that J_7 is glued to the arm tip, we conclude that $z(p) < 1$, for any $p \in U_7$.

The calculations regarding the $y \leq 1$ constraint are similar. Lemma 7 in the Appendix shows that the y-translation component of M_3 is never positive. □

Theorem 1. *A 6-arm cannot self-intersect.*

Proof. By Lemma 4, no arm self-intersects. By definition of the six octants, it can be verified that no octants intersect. By Lemma 6, this means that no two arms intersect. □

Because a 6-arm always avoids self-intersection (Theorem 1), stays within the bounding box determined by its tips (Lemma 5), and each tip moves only along the normal of its axis-aligned face (Lemma 3), we conclude that the generic 6-arm correctly simulates a Crystalline atom.

7. Efficiency of 6-arm reconfiguration

We have established that a generic 6-arm simulates a Crystalline atom. Moreover, the number of atoms used to construct a 6-arm is constant for all prototyped robots that we have considered. Thus any motion carried out by a 6-arm can be considered to use constant force and achieve constant velocities. In other words, our 6-arm construction does not affect any of the models considered in the literature. Let a 6-arm k-metamodule be a $k\times k\times k$ collection of 6-arm metamodules. By substituting 6-arm k-metamodules for Crystalline metamodules, in prior work in the literature, we obtain identical *upper* bounds, while worst-case optimality is obtained in an almost identical way:

Theorem 2. [2] *We can universally reconfigure n 6-arm 2-metamodules in $O(n)$ time using $O(n)$ operations.*

We note here that the number of operations in Theorem 2 is asymptotically optimal in the worst case. In [2] this was demonstrated by a simple example of reconfiguring from a horizontal line to a vertical line. However, for M-TRAN and Molecube, this can be done in constant time. Instead, we can use the simple reconfiguration from all blocks straight, to an alternating straight-bent pattern. Then every other block must reconfigure.

Theorem 3. [3] *We can universally reconfigure n 6-arm constant-size metamodules in $O(\log n)$ parallel steps and $O(n \log n)$ operations.*[2] *The number of parallel steps is optimal for labeled metamodules.*

[2]The result in [3] is restricted to 2D and the constant is 32, but a straightforward extension applies to 3D. The best constant remains to be rigorously verified.

Theorem 4. [1] *If only constant forces and velocities are allowed, we can universally reconfigure n 6-arm 2-metamodules in $O(n)$ parallel steps and $O(n^2)$ operations, and these bounds are optimal in the worst case. Furthermore, this can be done using only constant memory per atom, and with only local communication.*

We note that the worst-case optimality cannot be deduced directly from [1], because we have only proved a one-way reduction. However, the same reasoning and example suffice: reconfiguring from a horizontal straight configuration to a vertical straight configuration. First, a constant fraction of the metamodules must change their vertical coordinate by an additive $\Omega(n)$, for a total change of $\Omega(n^2)$. Second, each constant-force operation changes the vertical coordinates of a constant number of metamodules by an additive constant, for a total change of $O(1)$. Therefore, the total number of operations must be at least $\Omega(n^2)$. Each parallel step can perform at most $O(n)$ operations (one per unit), so the total number of parallel steps must be $\Omega(n^2/n) = \Omega(n)$.

Theorem 5. [17] *If constant forces are required (but velocity is unrestricted), we can universally reconfigure between any two 2D configurations of n 6-arm constant-size metamodules in $O(\sqrt{n})$ time, using the third dimension as an intermediate.* [3]

8. Comments and Future Work

Our results show that all modular robots meeting the requirements of our generic block can be reconfigured within the same asymptotic time bounds as Crystalline robots, provided an appropriate metamodule structure is used. We discussed four prototypes that fit our generic model. These prototypes are representatives of two main classes of hinge models: the M-TRAN and SuperBot robots are edge-hinged, and the Molecube and RoomBot are central-point-hinged. We believe that the techniques developed here can be applied to other modular robots such as PolyBot G3 [22] and ATRON [9], so this is one direction for future work. Another direction is to customize our 6-arm metamodule for each type of robot to minimize its size while preserving its functionality. This was done in 2D for

[3] The constant is not specified in [17], but metamodules of size at least $k = 4$ are necessary to carry out the operations upon which their algorithm is based.

the M-TRAN robot [12]. It would also be interesting to use space-filling meta-modules (densely filling a $k \times k \times k$ cube) or prove such a reduction impossible. Another interesting question is whether any modular robot is fundamentally more powerful than the Crystalline robot in the sense that it can reconfigure itself in $o(\log n)$ time, because other bounds seem to be tight for any model.

References

[1] G. Aloupis, S. Collette, M. Damian, E. D. Demaine, D. El-Khechen, R. Flatland, S. Langerman, J. O'Rourke, V. Pinciu, S. Ramaswami, V. Sacristán, and S. Wuhrer. Realistic reconfiguration of Crystalline and Telecube robots. In *Proceedings of the 8th International Workshop on Algorithmic Foundations of Robotics*, pages 433–447, 2008.

[2] G. Aloupis, S. Collette, M. Damian, E. D. Demaine, R. Flatland, S. Langerman, J. O'Rourke, S. Ramaswami, V. Sacristán, and S. Wuhrer. Linear reconfiguration of cube-style modular robots. *Computational Geometry: Theory and Applications*, 42(6,7):652–663, 2009.

[3] G. Aloupis, S. Collette, E. D. Demaine, S. Langerman, V. Sacristán, and S. Wuhrer. Reconfiguration of cube-style modular robots using $O(\log n)$ parallel moves. In *Proceedings of the 19th International Symposium on Algorithms and Computation*, volume 5369 of *LNCS*, 2008.

[4] Z. Butler, R. Fitch, and D. Rus. Distributed control for unit-compressible robots: Goal-recognition, locomotion and splitting. *IEEE/ASME Transactions on Mechatronics*, 7(4):418–430, 2002.

[5] A. Castano, A. Behar, and P. Will. The CONRO modules for reconfigurable robots. *IEEE/ASME Transactions on Mechatronics*, 7(4):403–409, 2002.

[6] H. Chiu, M. Rubenstein, and W. Shen. Multifunctional Superbot with rolling track configuration. In *Workshop on Self-Reconfigurable Robots & Systems and Applications*, pages 50–53, November 2007.

[7] J. Davey, N. Kwok, and M. Yim. Emulating self-reconfigurable robots - design of the SMORES system. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 4464–4469, 2012.

[8] D. J. Dewey, M. P. Ashley-Rollman, M. D. Rosa, S. C. Goldstein, and T. C. Mowry. Generalizing metamodules to simplify planning in modular robotic systems. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 1338–1345, 2008.

[9] M. W. Jørgensen, E. H. Østergaard, and H. H. Lund. Modular ATRON: Modules for a self-reconfigurable robot. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 2068–2073, 2004.

[10] K. Kotay and D. Rus. Algorithms for self-reconfiguring molecule motion planning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2184–2193, 2000.

[11] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, and S. Murata. Self-reconfigurable modular robot M-TRAN: distributed control and communication. In *Proceedings of the 1st International Conference on Robot Communication and Coordination*, pages 1–7. IEEE Press, 2007.

[12] H. Kurokawa, E. Yoshida, K. Tomita, A. Kamimura, S. Murata, and S. Kokaji. Self-reconfigurable M-TRAN structures and walker generation. *Robotics and Autonomous Systems*, 54:142–149, 2006.

[13] M. Kutzer, M. Moses, C. Brown, D. Scheidt, G. Chirikjian, and M. Armand. Design of a new independently-mobile reconfigurable modular robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2758–2764, 2010.

[14] S. Murata and H. Kurokawa. Self-reconfigurable robots: Shape-changing cellular robots can exceed conventional robot flexibility. *IEEE Robotics & Automation Magazine*, 14(1):43–52, 2007.

[15] A. T. Nguyen, L. Guibas, and M. Yim. Controlled module density helps reconfiguration planning. In *Proceedings of the 4th International Workshop on Algorithmic Foundations of Robotics*, pages 15–27, 2000.

[16] K. C. Prevas, C. Ünsal, M. Önder Efe, and P. K. Khosla. A hierarchical motion planning strategy for a uniform self-reconfigurable modular robotic system. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 787–792, 2002.

[17] J. H. Reif and S. Slee. Optimal kinodynamic motion planning for self-reconfigurable robots between arbitrary 2D configurations. In *Robotics: Science and Systems Conference*, 2007.

[18] D. Rus and M. Vona. Crystalline robots: Self-reconfiguration with compressible unit modules. *Autonomous Robots*, 10(1):107–124, 2001.

[19] W. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, and J. Venkatesh. Multimode locomotion via Superbot reconfigurable robots. *Autonomous Robots*, 20(2):165–177, 2006.

[20] A. Sproewitz, A. Billard, P. Dillenbourg, and A. Ijspeert. Roombots - mechanical design of self-reconfiguring modular robots for adaptive furniture. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4259–4264, 2009.

[21] S. Vassilvitskii, J. Kubica, E. Rieffel, J. S., and M. Yim. On the general reconfiguration problem for expanding cube style modular robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 801–808, 2002.

[22] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. Homans. Modular reconfigurable robots in space applications. *Autonomous Robots*, 14(2-3):225–237, 2003.

[23] M. Yim, W. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian. Modular self-reconfigurable robots systems: Challenges and opportunities for the future. *IEEE Robotics & Automation Magazine*, 14(1):43–52, 2007.

[24] V. Zykov, A. Chan, and H. Lipson. Molecubes: An open-source modular robotics kit. In *IROS-2007 Self-Reconfigurable Robotics Workshop*, 2007.

[25] V. Zykov, E. Mytilinaios, B. Adams, and H. Lipson. Self-reproducing machines. *Nature*, 435(7038):163–164, 2005.

[26] V. Zykov, P. Williams, N. Lassabe, and H. Lipson. Molecubes extended: Diversifying capabilities of open-source modular robotics. In *IROS-2008 Self-Reconfigurable Robotics Workshop*, 2008.

Appendix A. Lemma 7 Proof

In what follows we will refer to the 4×4 rotation matrix $R_u(\theta)$ below that rotates a (homogeneous) point by θ degrees (counterclockwise) about an axis parallel to the unit vector $u = (u_x, u_y, u_z)$ with fixed point at the origin:

$$R_u(\theta) = \begin{pmatrix} c+(1-c)u_x^2 & (1-c)u_yu_x - su_z & (1-c)u_zu_x + su_y & 0 \\ (1-c)u_xu_y + su_z & c+(1-c)u_y^2 & (1-c)u_zu_y - su_x & 0 \\ (1-c)u_xu_z - su_y & (1-c)u_yu_z + su_x & c+(1-c)u_z^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.1)}$$

where $c = \cos(\theta)$ and $s = \sin(\theta)$.

Lemma 7. *The z and y translation components of matrix M_3 of Equation 1 are less than or equal to zero throughout the expansion and contraction of the right arm.*

Proof. We show that the z-translation component, Z_3, of the matrix M_3 is always less than or equal to 0. Matrix multiplication shows that M_3 is a translation matrix, and its z component is:

$$Z_3 = -4(\cos(\theta) + (1-\cos(\theta))a_z^2)$$

Observe that since $a_x = -a_z$, this can be rewritten as $Z_3 = -4(\cos(\theta) + (1-\cos(\theta))a_x^2)$. Now note that the second factor in this expression is precisely the x component of the (homogeneous) vector $R_a(\theta)(1,0,0,0)^T$. (See Equation (A.1) for $R_a(\theta)$.) Let $0 \le \theta \le \theta_{max}$ be the range of θ that takes the unit between straight and bent position. Under the rotational requirement (3.d) of our generic model, we have that $R_a(\theta_{max})(1,0,0,0)^T = (0,0,-1,0)^T$. We give a geometric argument that vector $R_a(\theta)(1,0,0,0)^T$ stays in the positive x halfspace, for $0 \le \theta \le \theta_{max}$. Recall that $a_x, a_y \ge 0$ and $a_x = -a_z$ (by rotational requirements (3.b) and (3.d), adjusted for the orientation of U_3). Thus when vector a is rooted at the origin, its tip may lie anywhere on the quarter circle in the $a_x = -a_z$ plane that is delimited by vectors $(0,1,0)$ and $(1/\sqrt{2}, 0, -1/\sqrt{2})$. For each of these two delimiting vectors and a third intermediate vector in this range, Figure A.12 illustrates the corresponding circular motion path (in green) that is traced by the tip of vector $R_a(\theta)(1,0,0,0)^T$, as it rotates between the $+x$ and $-z$ axis. The

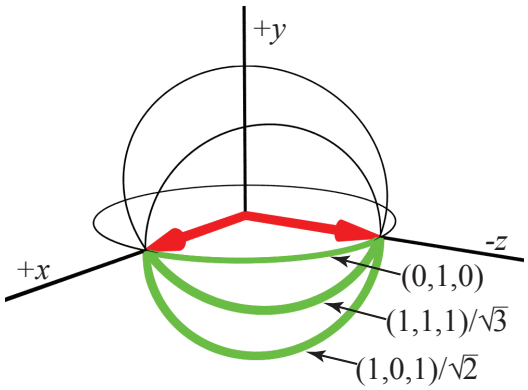

Figure A.12: Paths traced by tip of vector as it rotates between the positive x axis and the $-z$ axis, for three rotation axes.

full rotation circles are shown (in thin black) to give better perspective. Note that each rotation circle is labeled with its corresponding rotation axis. The rotation axes are not illustrated, but they are perpendicular to the plane containing the corresponding rotation circle. Observe that in each case the (green) path swept out is confined to the $(x \geq 0, y \leq 0, z \leq 0)$ octant. Since the x component is positive throughout the sweep, we have $(\cos(\theta) + (1 - \cos(\theta))a_x^2) \geq 0$ and thus $Z_3 \leq 0$.

Next we show that the y-translation component, Y_3, of matrix M_3 is always less than or equal to 0. The value of Y_3 is:

$$Y_3 = -4((1 - \cos(\theta))a_z a_y + \sin(\theta))a_x)$$

Since $a_x = -a_z$, the right term can be rewritten as $4((1-\cos(\theta))a_x a_y + \sin(\theta)a_z)$. Note that the second factor in this expression is the y component of the vector $R_a(\theta)(1, 0, 0, 0)^T$. As discussed above, the y component of this vector is at most zero as it rotates between the $+x$ and $-z$ axis, and so we have that $Y_3 \leq 0$. $\square$