Asymptotically Optimal Kinodynamic Motion Planning for Self-Reconfigurable Robots

John H. Reif[1] and Sam Slee[1]

Department of Computer Science, Duke University, Durham, NC, USA.
{reif,sgs}@cs.duke.edu

Summary. Self-reconfigurable robots are composed of many individual modules that can autonomously move to transform the shape and structure of the robot. In this paper we present a kinodynamically optimal algorithm for the following "x-axis to y-axis" reconfiguration problem: given a horizontal row of n modules, reconfigure that collection into a vertical column of n modules. The goal is to determine the sequence of movements of the modules that minimizes the movement time needed to achieve the desired reconfiguration of the modules. Prior work on self-reconfigurable (SR) robots assumed a constant velocity bound on module movement and so required time linear in n to solve this problem.

In this paper we define an abstract model that assumes unit bounds on various physical properties of modules such as shape, aspect ratio, mass, and the maximum magnitude of force that an individual module can exert. We also define concrete instances of our abstract model similar to those found in the prior literature on reconfigurable robots, including various examples where the modules are cubes that are attached and can apply forces to neighboring cubes. In one of these concrete models, the cube's sides can contract and expand with controllable force, and in another the cubes can apply rotational torque to their neighbors. Our main result is a proof of tight $\Theta(\sqrt{n})$ upper and lower bounds on the movement time for the above reconguration problem for concrete instances of our abstract model.

This paper's analysis characterizes optimal reconfiguration movements in terms of basic laws of physics relating force, mass, acceleration, distance traveled, and movement time. A key property resulting from this is that through the simultaneous application of constant-bounded forces by a system of modules, certain modules in the system can achieve velocities exceeding any constant bounds. This delays modules with the least distance to travel when reconfiguring in order to accelerate modules that have the farthest to travel. We utilize this tradeoff in our algorithm for the x-axis to y-axis problem to achieve an $O(\sqrt{n})$ movement time.
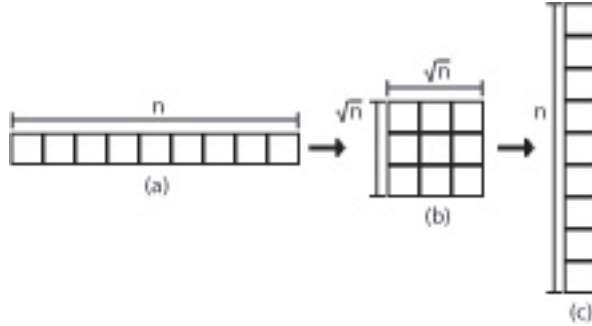
Fig. 1. The x-axis to y-axis problem: transforming a row of modules into a column. Our algorithm uses an intermediate step of forming a square.

1 Introduction

This paper develops efficient algorithms by treating reconfiguration as a kino-dynamic planning problem. Kinodynamic planning refers to motion planning problems subject to simultaneous kinematic and dynamics constraints [3]. To that end, in this paper we define an abstract model for modules in SR robots. This extends the definitions of previous models [4, 5] by setting fixed unit bounds on a module's shape, aspect ratio, mass, and the force it can apply, among other requirements.

To exhibit the significance of these bounds we consider the following "x-axis to y-axis" reconfiguration problem. Given a horizontal row of n modules, reconfigure that collection into a vertical column of n modules. This simple problem, illustrated in Figure 1, provides a worst-case example in that $\Omega(n)$ modules must move $\Omega(n)$ module lengths to reach any position in the goal configuration regardless of that goal column's horizontal placement. Any horizontal positioning of the vertical column along the initial row configuration is deemed acceptable in our treatment of the problem in this paper.

We define the *movement time* of a reconfiguration problem to be the time taken for a system of n modules to reconfigure from an initial configuration to a desired goal configuration. Modules are assumed to be interchangeable so the exact placement of a given module in the initial or goal configuration is irrelevant. An implicit assumption in various prior papers on reconfigurable robot motion planning [6, 4, 7] is that the modules are permitted only a fixed unit velocity. This assumption is not essential to the physics of these systems and has constrained prior work in the area.

For the x-axis to y-axis problem, if only one module in the SR robot is permitted to move at a given time and with only a fixed unit velocity, a lower bound of $\Omega(n^2)$-time is clear [6]. Allowing concurrent movement of modules, a movement time of $O(n)$ is possible while still keeping all modules connected to the system and keeping unit velocities. However, we observe that faster

reconfiguration is possible if we do not restrict modules to move at a fixed, uniform pace.

Another major principle that we use, and that has seen use in prior reconfiguration algorithms [8, 1], is what we refer to as the principle of *time reversal*. This principle is simply that executing reconfiguration movements in reverse is always possible, and they take precisely the same movement time as in the forward direction. This is, of course, ignoring concerns such as gravity. Otherwise an example of rolling a ball down a hill would require less time or less force than moving that ball back up the hill. We ignore gravity and use the principle of time reversal extensively in the work of this paper.

Before continuing further, it will be useful to define some notation that we will use throughout the remainder of this paper. As given above, let n denote the number of modules in the SR robot undergoing reconfiguration. In the initial row configuration of the x-axis to y-axis problem, let the modules be numbered $1, \ldots, n$ from left to right. Let $x_i(t)$ be the x-axis location of module i at time t. Similarly, let $v_i(t)$ and $a_i(t)$ be the velocity and acceleration, respectively, of module i at time t. For simplicity, the analysis in our examples will ignore aspects such as friction or gravity. This simplification does make this paper's results more theoretical in nature. Yet, providing motion planning bounds for fundamental reconfiguration problems like the one considered here are important for understanding what is possible for this class of robots.

In the following Section 2, we differentiate between different styles of self-reconfigurable robots and survey related work in the field. We begin introducing the work of this paper in Section 3 by giving our abstract model for SR robot modules. Given the bounds set by this model, Section 4 references physics equations that govern the movement of modules and define what reconfiguration performance is possible. To explain our algorithm, in Section 4 we also begin with a 1-dimensional case of n masses represented as a row of n points with a separation of 1 unit between adjacent points, for some unit of distance measure. The points will then contract so that they have only $1/2$ unit separation. After showing that this contraction takes $O(\sqrt{n})$ movement time while still satisfying the bounds of our abstract model, we then immediately extend this result to a matching example using a known physical architecture for modules.

Section 5 will then extend the result to a contraction/expansion case in 2 dimensions while maintaining the same time bound. This will then lead to a $O(\sqrt{n})$ movement time algorithm for the x-axis to y-axis reconfiguration problem. This algorithm recursively uses the 1-dimensional contraction operation and uses the reversible process of transforming the initial n module row into an intermediate stage $\sqrt{n} \times \sqrt{n}$ cube. The process is then reversed to go from the cube to the goal column configuration. Section 6 then matches this with a $\Omega(\sqrt{n})$ lower bound for the 1-dimensional example and the x-axis to y-axis problem, showing both cases to be $\Theta(\sqrt{n})$. Finally, the conclusion in Section 7 summarizes the results of this paper.

2 Related Work

When developing models and algorithmic bounds for self-reconfigurable (SR) robots (also known as metamorphic robots [10, 5, 6]) it is important to note the style of SR robot we are dealing with. Two of the main types of SR robots are closed-chain style robots and lattice style robots. Closed-chain SR robots are composed of open or closed kinematic chains of modules. Their topology is described by one-dimensional combinatorial topology. [1] To reconfigure these modules are required to swing chains of other modules to new locations. One such implementation of this design by Yim et al. has had several demonstrations of locomotion [11].

For the other major style, lattice or substrate SR robots attach together only at discrete locations to form lattice-like structures. Individual modules typically move by walking along the surfaces formed by other modules. The hardware requirements for this style of module are more relaxed than those for closed-chain style systems. Here individual modules need only be strong enough to move themselves or one or two neighbors. Closed-chain style modules typically must be strong enough to swing long chains of other modules. The models and algorithms presented in this paper are meant for lattice style modules.

Previous work by several research groups has developed abstract models for lattice style SR robots. One of the most recent is the sliding cube model proposed by Rus et al. [4]. As the name implies, this model represents modules as identical cubes that can slide along the flat surfaces created by lattices of other modules. In addition, modules have the ability to make convex or concave transitions to other orthogonal surfaces. In this abstraction, a single step action for a module would be to detach from its current location and then either transition to a neighboring location on the lattice surface, or make a convex or concave transition to another orthogonal surface to which it is next. Transitions are only made in the cardinal directions (no diagonal movements) and for a module to transition to a neighboring location that location must first be unoccupied. Most architectures for lattice style SR robots satisfy the requirements of this model.

One such physical implementation is the compressible unit or expanding cube design [7, 8]. Here an individual module can expand from its original size to double its length in any given dimension, or alternatively compress to half its original length. Neighbor modules are then pushed or pulled by this action to generate movement and allow reconfiguration. This particular module design is relevant because it provides a good visual aide for the algorithms we present in this paper. For this purpose we also add the ability for expanding cubes to slide relative to each other. Otherwise a single row of these modules can only exert an expanding or contracting force along the length of that row and is unable to reconfigure in 2 dimensions. Prior work has noted that while individual expanding cube modules do not have this ability, it can be approximated when groups of modules are treated as atomic units [7, 8].

3 Abstract Model

We now begin introducing our results for this paper by defining our abstract model for SR robot modules. It generalizes many of the requirements and properties of reconfigurable robots. Our abstract model explicitly states bounds on physical properties such as the size, mass, and force exerted for each module. These properties will be utilized by the algorithms and complexity bounds that later follow. The requirements of our model are described by the following set of axioms.

- Each module is assumed to be an object in 3D with either (i) fixed shape and size or (ii) has a limited set of geometric shapes that it can acquire. In either case, the module also has a constant bound on its total volume and the aspect ratio of its shape.
- Each module can latch onto or grip adjacent modules and apply to these neighboring modules a force or torque.
- Generally, modules are all connected either directly or indirectly at any time.
- For each module there is a constant bound on its mass.
- There is a constant bound on the magnitude of the force and/or torque that a module may apply to those modules with which it is in contact.
- The motion of modules is such that they never collide with a velocity above a fixed constant magnitude.
- For modules in direct contact with each other, the magnitude of the difference in velocity between these contacting modules is always bounded by a constant.
- Each module, when attached (or latched) to other modules, can dynamically set the stiffness of the attachment. This in addition to the ability of the module to apply contraction/expansion or rotational forces at its specified attachments.

Note: The axiom on the stiffness of the attachments to neighboring modules is required to ensure that forces (external to the module) are transmitted through it to neighboring modules. We add this since contraction and/or rotational forces along a chain of modules can accumulate to amounts more than the unit maximum applied by any one module, and may need to be transfered from neighbor to neighbor. Note also that we assume idealized modules that act in synchronized movements under centralized control. This reduces analysis difficulties for cases when many sets of modules operate in parallel.

While the axioms given above state important module requirements, more concrete models are necessary. The sliding cube model and the expanding cube hardware design described in the previous section satisfy the axioms of our abstract model so long as bounds on the physical abilities of modules are implied. In particular, for these modules the "stiffness" requirement means that the transmitted forces are translational. The exact use of this will become apparent in our first 1-dimensional example with expanding cube modules.

4 1D Force Analysis

All analysis of movement planning in this paper is based on the physical limitations of individual modules stated in the last section. Given this abstract model, we can now make use of the elementary equations of Newtonian physics governing the modules' movement in space and time. For these equations we use the notation first mention in Section 1. Let $x_i(t)$ be the position of object i during movement time t. Similarly, $v_i(t)$ is its velocity after time t. Finally, given an object i with mass m_i, let F_i be the net force applied to that object and a_i be the resulting constant acceleration. Although the relevant equations are basic, we state them here so that they can be referred to again later in the paper:

$$F_i = m_i a_i \tag{1}$$

$$x_i(t) = x_i(0) + v_i(0)t + \frac{1}{2}a_i t^2 \tag{2}$$

$$v_i(t) = v_i(0) + a_i t \ . \tag{3}$$

In the first equation we get that a net force of F_i is required to move an object with mass m_i at a constant acceleration with magnitude a_i. In the second equation, an object's location $x_i(t)$ after traveling for a time t is given by it's initial position $x_i(0)$, it's initial velocity $v_i(0)$ multiplied by the time, and a function of a constant acceleration a_i and the time traveled squared. Similarly, the third equation gives the velocity after time t, $v_i(t)$, to be the initial velocity $v_i(0)$ plus the constant acceleration a_i multiplied by the time traveled.

All modules in the examples that follow are assumed to have unit mass $m = 1$ and sides with unit length 1. Our algorithms all require 2 stages of motion: one stage to begin motion and a second stage to slow it and ensure zero final velocity. So, we assume that each module is capable of producing a unit amount, 1, of force once for each stage. This still keeps within the constant-bounded force requirement of our abstract model. For an expanding cube module this unit force is capable of contracting or expanding it in unit time while pulling or pushing one neighbor module. Also, since the module has unit mass, by equation (1) we get a bound on its acceleration $a \leq 1$ given the force applied in just a single motion stage. Again, all concerns for friction or gravity (or a detailed description of physical materials to ensure the proper stiffness of modules) are ignored to simplify calculations.

Before tackling the x-axis to y-axis reconfiguration problem, we first begin by analyzing a simpler 1-dimensional case which we will refer to as the *Point Mass Contraction* problem. Here a row of n point masses will contract from a total length of n units, to a length of $n/2$. Although these point masses are not connected and do not grip each other, they otherwise satisfy the axioms of our model. After showing this reconfiguration to take $\mathrm{O}(\sqrt{n})$ movement time, we

will show that same result holds for a case with expanding-cube style modules instead of point masses. This 1-dimensional contraction case, which we refer to as the *Squeeze* problem, will be a recursive step in our algorithm for the x-axis to y-axis reconfiguration problem.

We assume the initial configuration consists of an even number n of point masses arranged in a row on the x-axis, each initially having 0 velocity. For $i = 1, \ldots, n$ the ith point initially at time $t = 0$ has x-coordinate $x_i(0) = i - (n+1)/2$. We assume each point has unit mass and can move in the x-direction with acceleration magnitude upper bounded by 1. For simplicity, we assume no friction nor any gravitational forces. Our goal configuration at the final time T is the point masses arranged in a row on the x-axis with 0 final velocity, each distance $1/2$ from the next in the x-axis direction, so that for $i = 1, \ldots, n$ the ith point at the final time T has x-coordinate $x_i(T) = \frac{x_i(0)}{2} = \frac{i}{2} - \frac{n+1}{4}$.

To differentiate notation, we'll use parentheses to denote a function of time, such as $x_i(t)$, and square brackets to denote order of operations, such as $2 * [3 - 1] = 4$. Furthermore, we require that the velocity difference between consecutive points is at most 1, and that consecutive points never get closer than a distance of $1/2$ from each other. Given this, our goal is to find the minimal possible movement time duration from the initial configuration at time 0 to final configuration at time T.

Lemma 1. *The Point Mass Contraction problem requires at most total movement time $T = \sqrt{n - 1}$.*

Proof: Fix $T = \sqrt{n - 1}$. For each point mass $i = 1, \ldots, n$ at time t, for $0 \le t \le T$, let $x_i(t)$ be the x-coordinate of the ith point mass at time t and let $v_i(t), a_i(t)$ be its velocity and acceleration, respectively, in the x-direction (our algorithm for this Point Mass Contraction Problem will provide velocity and acceleration only in the x-direction).

For $i = 1, \ldots, n$ the ith point mass needs to move from initial x-coordinate $x_i(0) = i - (n+1)/2$ starting with initial velocity $v_i(0) = 0$ to final x-coordinate $x_i(T) = x_i(0)/2 = i/2 - (n+1)/4$ ending with final velocity $v_i(T) = 0$. To ensure the velocity at the final time is 0, during the first half of the movement time the ith point mass will be accelerated by an amount α_i in the intended direction, and then during the second half of the movement time the ith point mass will accelerate by $-\alpha_i$ (in the reverse of the intended direction). For $i = 1, \ldots, n$, set that acceleration as: $\alpha_i = \frac{n+1-2i}{n-1}$.

Note that the maximum acceleration bound is unit, since $|\alpha_i| \le 1$, satisfying the requirement of our abstract model. During the first half of the movement time t, for $0 \le t \le T/2$, by equations (3) and (2) we get that the velocity at time t is $v_i(t) = v_i(0) + \alpha_i t$ which implies $v_i(t) = \alpha_i t$ and the x-coordinate is given by:

$$x_i(t) = x_i(0) + v_i(0)t + \frac{\alpha_i}{2}t^2 = i - \frac{n+1}{2} + \frac{\alpha_i}{2}t^2 \ .$$

At the midway point $T/2$, this gives the ith point mass velocity $v_i(T/2) = \alpha_i T/2$ and x-coordinate

$$x_i(T/2) = i - \frac{n+1}{2} + \frac{\alpha_i}{2}\left[\frac{T}{2}\right]^2 = i - \frac{n+1}{2} + \frac{a_i T^2}{8} \ .$$

During the second half of the movement time t, for $T/2 < t \leq T$, we will set the ith point mass acceleration at time t to be $a_i(t) = -\alpha_i$. By equation (3) we get that the velocity at time t during the second half of the movement time is:

$$v_i(t) = v_i(T/2) - \alpha_i\left[t - \frac{T}{2}\right] = \alpha_i\left[\frac{T}{2}\right] - \alpha_i\left[t - \frac{T}{2}\right]$$

$$v_i(t) = \alpha_i[T - t] \ .$$

and the x-coordinate given by equation (2) is

$$x_i(t) = x_i(T/2) + v_i(T/2)\left[t - \frac{T}{2}\right] - \frac{\alpha_i}{2}\left[t - \frac{T}{2}\right]^2$$

$$x_i(t) = \left[i - \frac{n+1}{2} + \frac{\alpha_i T^2}{8}\right] + \frac{\alpha_i T}{2}\left[t - \frac{T}{2}\right] - \frac{\alpha_i}{2}\left[t - \frac{T}{2}\right]^2 \ .$$

This implies that at the final time T, the ith point mass acceleration has velocity $v_i(T) = \alpha_i(T - T) = 0$ and x-coordinate

$$x_i(T) = \left[i - \frac{n+1}{2} + \frac{\alpha_i T^2}{8}\right] + \frac{\alpha_i T}{2}\left[T - \frac{T}{2}\right] - \frac{\alpha_i}{2}\left[T - \frac{T}{2}\right]^2$$

$$x_i(T) = \left[i - \frac{n+1}{2} + \frac{\alpha_i T^2}{8}\right] + \frac{\alpha_i T^2}{4} - \frac{\alpha_i T^2}{8}$$

$$x_i(T) = i - \frac{n+1}{2} + \frac{\alpha_i T^2}{4} \ .$$

Recalling that we initially set $\alpha_i = \frac{n+1-2i}{n-1}$ and $T = \sqrt{n-1}$ we get:

$$x_i(T) = i - \frac{n+1}{2} + \frac{n+1-2i}{4} = \frac{i}{2} - \frac{n+1}{4} = \frac{x_i(0)}{2} \ .$$

So we get that $x_i(T) = x_i(0)/2$ as required. Moreover, the time any consecutive points are closest is the final time T, and at that time they are distance $1/2$ from each other, as required in the specification of the problem.

It is easy to verify that the velocity difference between consecutive points masses i and $i+1$ is maximized at time $T/2$, and at that time the the magnitude of the velocity difference is:

$$|v_i(T/2) - v_{i+1}(T/2)| = |\alpha_i - \alpha_{i+1}|T/2 \leq \frac{2}{n-1}\frac{\sqrt{n-1}}{2} \leq 1$$

as required in the specification of the problem. Thus, we have shown it possible to complete this problem in time $T = \sqrt{n-1}$ while still satisfying the axioms of our abstract model (excluding connectivity and gripping). $\square$

The *Squeeze* Problem With this result proved for the Point Mass Contraction problem, the same reconfiguration using expanding cube modules can be solved by directly applying Lemma 1. We refer to this reconfiguration task as the *Squeeze* problem and define it as follows. Assume an even number of n modules, numbered $i = 1, \ldots, n$ from left to right as before. Keeping the modules connected as a single row, our goal is to contract the modules from each having length 1 to each having length $1/2$ for some unit length in the x-axis direction. This reconfiguration task is shown in Figure 2. Given the bounds on the module's physical properties required by our abstract model, the goal is to perform this reconfiguration in the minimum possible movement time T.
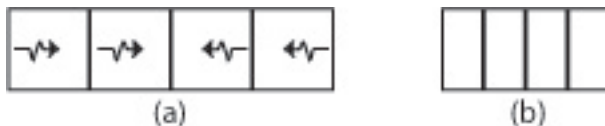
Fig. 2. The *Squeeze* problem: Expanding cube modules in (a), each of length 1, contract to each have length $1/2$ and form the configuration in (b).

Similarly, we define the *Reverse Squeeze* problem as the operation that undoes the first reconfiguration. Given a connected row of n contracted modules, each with length $1/2$ in the x-axis direction, expand those modules to each have length 1 while keeping the system connected as a single row. Since this problem is exactly the reverse of the original *Squeeze* problem, steps for an algorithm solving the *Squeeze* problem can be run in reverse order to solve the Reverse *Squeeze* problem in the same movement time with the same amount of force.

We first perform the force analysis for the case of solving the *Squeeze* problem. At the initial time $t = 0$ cube i's center x-coordinate has location $x_i(0) = i - (n+1)/2$. Recall that each cube is assumed to have unit mass 1, can grip its adjoining cubes, and can exert expanding or contracting forces against those neighbors. Furthermore, a unit upper bound is assumed on the magnitude of this force, which as stated earlier also causes an acceleration bound $a_i \leq 1$ for all modules. We wish to contract the center x-coordinates of the modules from location $x_i(0)$ to $x_i(0)/2 = i/2 - [n+1]/4$ as before in the contraction example with point masses. Again, we wish to find the minimum possible movement time for contraction from the initial configuration at time 0 to the goal configuration at time T.

Lemma 2. *The Squeeze problem requires at most total movement time $T = \sqrt{n-1}$.*

Proof is given in the Appendix A.1.

Given that the above problem requires at most $T = \sqrt{n-1}$ time to be solved, we get the same result for the *Reverse Squeeze* problem.

Corollary 1. *The Reverse Squeeze problem requires at most movement time $T = \sqrt{n-1}$.*

Note that all of the operations performed in solving the *Squeeze* problem can be done in reverse. The forces and resulting accelerations used to contract modules can be performed in reverse to expand modules that were previously contracted. Thus, we can reverse the above *Squeeze* algorithm in order to solve the *Reverse Squeeze* problem in exactly the same movement time as the original *Squeeze* problem. (Note: Although the forces are reversed in the Reverse Squeeze problem, the stiffness settings remain the same. It is important to observe that without these stiff attachments between neighboring modules, the accumulated expansion forces would make the entire assembly fly apart.)

5 2D Reconfiguration

The above analysis of the 1-dimensional *Squeeze* problem has laid the groundwork for reconfiguration in 2 dimensions. This includes the x-axis to y-axis reconfiguration problem which we will provide an algorithm for at the end of this section. We build to that result by first looking at a simpler example of 2-dimensional reconfiguration that will serve as an intermediate step in our final algorithm. We continue to use the same expanding cube model here as was used in the previous section.

In that previous section a connected row of an even number of n expanding cube modules was contracted from each module having length 1 in the x-axis direction to each having length $1/2$. We now begin with that contracted row and reconfigure it into two stacked rows of $n/2$ contracted modules. Modules begin with $\frac{1}{2} \times 1$ width by height dimensions and then finish with $1 \times \frac{1}{2}$ dimensions. This allows pairs of adjacent modules to rotate positions within a bounded 1×1 unit dimension square.We denote this reconfiguration task as the *confined cubes swapping* problem.

The process that we use to achieve reconfiguration is shown in Figure 3. We begin with a row of modules, each with dimension $\frac{1}{2} \times 1$. Here motion occurs in two parts. First, fix the bottom edge of odd numbered modules so that edge does not move. Do the same for the top edge of even numbered modules. Then contract all modules in the y direction from length 1 to length $1/2$. Note that to achieve this two counterbalancing forces are required: (1) a force within each module to contract it, and (2) sliding forces between adjacent
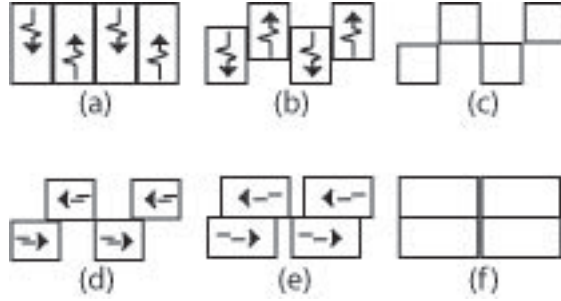
Fig. 3. Expanding cube modules contracting vertically (a - c), then expanding horizontally (d - f). Here the scrunched arrow represents contraction and the 3 piece arrow denotes expansion.

modules in the row to keep the required top/bottom edges in fixed locations as described earlier. This process creates the "checkered" configuration in part (c) of Figure 3.

We can then reverse the process, but execute it in the x direction instead, to expand the modules in the x direction and create 2 stacked rows of modules, each with dimension $1 \times \frac{1}{2}$. Note that requiring certain edges to stay in fixed locations had an important byproduct. This results in pairs of adjacent modules moving within the same 1×1 square at all times during reconfiguration. This also means that the bounding box of the entire row does not change as it reconfigures into 2 rows. This trait will be of great significance when this reconfiguration is executed in parallel on an initial configuration of several stacked rows.

Again, number modules $i = 1, \ldots, n$ from left to right and assume modules have unit mass 1, can grip each other, and can apply contraction, expansion and sliding forces. In this problem we will only use a unit force 1 total per module for both stages of motion. This means a force of $1/2$ applied in each motion stage and, by the physics equations in Section 4, an acceleration upper bounded by $1/2$ in each stage as well. Finally, concerns for gravity or friction are again ignored for the sake of simplicity. Given these bounds, our goal is to find the minimum reconfiguration time for this confined cubes swapping problem.

Lemma 3. *The confined cubes swapping problem described above requires* $T = O(1)$ *movement time for reconfiguration.*

Proof is given in the Appendix A.2.

Extending this analysis, we now consider the case of an $m \times n$ array of normal, unit-dimension modules. That is, we have m rows with n modules each. We wish to transform this into a $2m \times \frac{n}{2}$ array configuration. All of the same bounds on the physical properties of modules hold and gravity and

friction are still ignored. Once more we wish to find the minimum movement time T for this reconfiguration.
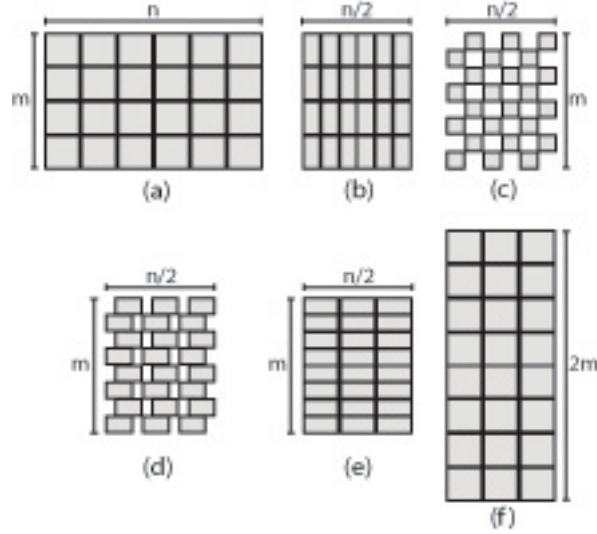
Fig. 4. Horizontal contraction from stage a to b. Vertical expansion from state e to f. Note that the dimensions of the array remain unchanged through stages b - e.

Lemma 4. *Reconfiguring from an $m \times n$ array of unit-dimension modules to an array of $2m \times \frac{n}{2}$ unit-dimension modules takes $O(\sqrt{m} + \sqrt{n})$ movement time.*

Proofs are given in the Appendix A.3 and Appendix A.4.

The problem just analyzed may now be used iteratively to solve the x-axis to y-axis reconfiguration problem. By repeatedly applying the above array reconfiguration step, we double the height and halve the width of the array each time until we progress from a $1 \times n$ to an $n \times 1$ array of modules.

This process will take $O(\lg_2 n)$ such steps, and so there would seem to be a danger of the reconfiguration problem requiring an extra $\lg_2 n$ factor in its movement time. This is avoided because far less time is required to reconfigure arrays in intermediate steps. Note that the $O(\sqrt{m} + \sqrt{n})$ time bound will be dominated by the larger of the two values m and n. For the first $\lfloor (\lg_2 n)/2 \rfloor$ steps the larger value will be the number of columns n, until an array of $\lfloor \sqrt{n} \rfloor \times \lceil \sqrt{n} \rceil$ dimensions is reached. In the next step a $\lceil \sqrt{n} \rceil \times \lfloor \sqrt{n} \rfloor$ array of modules is created, and from that point on we have more rows than columns and the time bound is dominated by m.

The key aspect is that the movement time for each reconfiguration step is decreased by half from the time we begin until we reach an intermediate array of dimensions about $\sqrt{n} \times \sqrt{n}$. By the principle of time reversal it should take us the same amount of movement time to go from a single row of n modules to an $\sqrt{n} \times \sqrt{n}$ cube as it does to go from that cube to a single column of n modules. This tactic is now used in our analysis to find the minimum reconfiguration time for the x-axis to y-axis problem.

Lemma 5. *The x-axis to y-axis reconfiguration problem only requires movement time $O(\sqrt{n})$.*

Proof: For simplicity, let n be even and let $n = 2^p$ for some even integer $p > 0$. Let $r(i)$ and $c(i)$ be the number of rows and columns, respectively, in the module system after i reconfiguration steps. From the previous Lemma 4 in this section we have that a single step of reconfiguring an $m \times n$ array of modules into a $2m \times \frac{n}{2}$ array requires time $O(\sqrt{m} + \sqrt{n})$. Initially, assuming a large initial row length, then $c(0) = n$, $n \gg 1$, and the reconfiguration step takes $O(\sqrt{n})$ time. For subsequent steps we still have $c(i) \gg r(i)$, but $c(1) = n/2, c(2) = n/4$, etc. while $r(1) = 2, r(2) = 4$, etc. In general $c(i) = n/2^i$ and $r(i) = 2^i$. Eventually, we get $c(i') = r(i') = \sqrt{n}$ at $i' = (\lg_2 n)/2$. Up until that point the time for each reconfiguration stage $i + 1$ is $O(\sqrt{c(i)})$. So, the total reconfiguration time to that point is given by the following summation:

$$\sum_{i=0}^{(\lg_2 n)/2} \sqrt{c(i)} = \sum_{i=0}^{(\lg_2 n)/2} \sqrt{\frac{n}{2^i}} \leq \sqrt{n} \sum_{i=0}^{\infty} \left(\frac{1}{\sqrt{2}} \right)^i = \frac{\sqrt{n}}{1 - (1/\sqrt{2})} = O(\sqrt{n}) \ .$$

Thus we have that reconfiguration from the initial row of n modules to the intermediate $\sqrt{n} \times \sqrt{n}$ square configuration takes $O(\sqrt{n})$ movement time. Reconfiguring from this cube to the goal configuration is just the reverse operation: we are simply creating a "vertical row" now instead of a horizontal one. This reverse operation will then take the exact same movement time using the same amounts of force as the original operation, so it too takes $O(\sqrt{n})$ movement time. Thus, we have that while satisfying the requirements of our abstract model the x-axis to y-axis problem takes $O(\sqrt{n})$ movement time. $\square$

6 Lower Bounds

In Section 4 we showed that the 1-dimensional Point Mass Contraction Problem, reconfiguring a row of n point masses with unit separation to have $1/2$ distance separation, could be solved in time $T = \sqrt{n-1}$. We now show a matching lower bound for this problem. Again, the same assumptions about the physical properties of the point masses are held and concerns for friction or gravity are ignored.

Lemma 6. *The Point Mass Contraction Problem requires at least total movement time $T = \sqrt{n-1}$.*

Proof: Consider the movement of the 1st point mass which needs to move from initial x-coordinate $x_1(0) = 1 - [n+1]/2 = 1/2 - n/2$ starting with initial velocity $v_1(0) = 0$ to final x-coordinate $x_1(T) = x_1(0)/2 = 1/4 - n/4$ and ending with final velocity $v_1(T) = 0$. The total distance this point mass needs to travel is $x_1(T) - x_1(0) = [n-1]/4$.

Taking into consideration the constraint that the final velocity is to be 0, it is easy to verify that the time-optimal trajectory for point mass $i = 1$ is an acceleration of 1 in the positive x-direction from time 0 to time $T/2$, followed by a reverse acceleration of the same magnitude in the negative x-direction. The total distance traversed in each of the two stages is at most $[a_1/2]t^2 = 1 * [T/2]^2/2$. So, the total distance traversed by the 1st point mass is at most $[T/2]^2 = T^2/4$ which needs to be $[n-1]/4$. Hence, $T^2/4 \geq [n-1]/4$ and so $T \geq \sqrt{n-1}$. $\square$

Hence, we have shown:

Theorem 1. *The lower and upper bound for the total movement time for the Point Mass Contraction Problem is exactly $T = \sqrt{n-1}$.*

As in Section 4, we can use an extension of this lower bound argument to prove the following lemma.

Lemma 7. *The Squeeze problem requires total movement time $\Omega(\sqrt{n})$.*

Proof is given in the Appendix A.5.

Hence, we have also shown:

Theorem 2. *The total movement time for the Squeeze problem is both upper and lower bounded by $\Theta(\sqrt{n})$.*

By the same argument, we can also get a bound on the x-axis to y-axis problem.

Corollary 2. *The x-axis to y-axis reconfiguration problem requires total movement time $\Omega(\sqrt{n})$.*

Proof: Given the initial row configuration and the goal column configuration, pick the end of the row farthest away from the goal column configuration's horizontal placement. Select the n/c cubes at this end of the row, for some real number greater than 2. Among these n/c cubes we can again find some ith cube that must have an acceleration at most $a_i \approx c/2$ and that it must travel a distance $\geq n[1/4 - 1/[2c]] + 1/4$. This again leads to the movement time being bounded as $T = \Omega(\sqrt{n})$. $\square$

7 Conclusion

In this paper we have presented a novel abstract model for self-recongurable (SR) robots that provides a basis for kinodynamic motion planning for these robots. Our model explicitly requires that SR robot modules have unit bounds on their size, mass, magnitude of force or torque they can apply, and the relative velocity between directly connected modules. The model allows for feasible physical implementations and permits the use of basic laws of physics to derive improved reconguration algorithms and lower bounds.

In this paper we have focused on a simple and basic reconfiguration problem. Our main results were tight upper and lower bounds for the movement time for this problem. Our recursive *Squeeze* algorithm recongures a horizontal row of n modules into a vertical column in $\mathrm{O}(\sqrt{n})$-time. This result significantly improves on the running time of previous reconfiguration algorithms. Our algorithm satisfies the restrictions imposed by our abstract model and we also show that it is kinodynamically optimal given the assumptions of that model.

While carefully using the forces produced by modules, our analysis ignored forces caused by gravity and friction. Addressing these concerns is a topic for future work as the algorithm is brought closer to physical implementation. Also, the algorithm given was a centralized planner and only solved a simple example to demonstrate how faster reconfiguration algorithms were possible.

For reconfiguration in more general cases, the simplest extension of this paper's results would be movement between configurations with contiguous rows and columns. By excluding holes from configurations in this way, we can reconfigure the robot into a single horizontal row in the same manner that we reconfigured rectangles in intermediate steps of our x-axis to y-axis algorithm. Then by the principle of time reversal we can reconfigure from that row to any other state with contiguous rows and columns. Finally, the multipart nature of SR robots makes distributed algorithms a necessity. Extending our lower-bound analysis to more complex cases, and developing distributed algorithms to match those bounds, is a topic of future work.

References

1. A. Casal and M. Yim. Self-reconfiguration planning for a class of modular robots. In *Proceedings of SPIE, Sensor Fusion and Decentralized Control in Robotic Systems II, vol. 3839*, pages 246–255, 1999.
2. G. Chirikjian, A. Pamecha, and I. Ebert-Uphoff. Evaluating efficiency of self-reconfiguration in a class of modular robots. *Journal of Robotic Systems*, 13(5), May 1996.
3. Bruce Randall Donald, Patrick G. Xavier, John F. Canny, and John H. Reif. Kinodynamic motion planning. *Journal of the ACM*, 40(5):1048–1066, 1993.
4. K. Kotay and D. Rus. Generic distributed assembly and repair algorithms for self-reconfiguring robots. In *Proc. of IEEE Intl. Conf. on Intelligent Robots and Systems*, 2004.

5. A. Pamecha, C. Chiang, D. Stein, and G. Chirikjian. Design and implementation of metamorphic robots. In *Proceedings of the 1996 ASME Design Engineering Technical Conference and Computers in Engineering Conference*, 1996.
6. A. Pamecha, I. Ebert-Uphoff, and G. Chirikjian. Useful metrics for modular robot motion planning. In *IEEE Trans. Robot. Automat.*, pages 531–545, 1997.
7. Serguei Vassilvitskii, Jeremy Kubica, Eleanor Rieffel, John Suh, and Mark Yim. On the general reconfiguration problem for expanding cube style modular robots. In *Proceedings of the 2002 IEEE Int. Conference on Robotics and Automation*, pages 801–808, 11-15 May 2002.
8. M. Vona and D. Rus. Self-reconfiguration planning with compressible unit modules. In *1999 IEEE International Conference on Robotics and Automation*, 1999.
9. J. Walter, E. Tsai, and N. Amato. Algorithms for fast concurrent reconfiguration of hexagonal metamorphic robots. *IEEE Transactions on Robotics*, 21(4):621–631, August 2005.
10. Jennifer E. Walter, Jennifer L. Welch, and Nancy M. Amato. Distributed reconfiguration of metamorphic robot chains. In *PODC '00*, pages 171–180, 2000.
11. Mark Yim, David Duff, and Kimon Roufas. Polybot: A modular reconfigurable robot. In *ICRA*, pages 514–520, 2000.

APPENDIX

A Force Analysis Proofs

A.1 The *Squeeze* Problem

Proof of Lemma 2.

Proof: For each cube $i = 1, \ldots, n$ at time t for $0 \leq t \leq T$ let $x_i(t)$ be the x-coordinate of the center point of the ith cube and let $v_i(t)$ be its velocity in the x-direction. (Our algorithm for this *Squeeze* problem will provide velocity only in the x-direction.)

In the *Squeeze* problem, there will be an expansion/contraction force applied by each ith cube to its neighbors that are located just before and just after it on the x-axis. These consecutive neighbors are assumed to be gripped together. Let $\alpha_i = [n + 1 - 2i]/[n - 1]$ be the acceleration of each module i just as defined in the Point Mass Contraction problem.

During the first half of the movement time t, for $0 \leq t \leq T/2$, the ith cube will be subjected to external force $F_{i,L}(t) = \sum_{j=1}^{j=i-1} a_j(t)$ from its left neighbor (the $i - 1$th cube) and will be subjected to external force $F_{i,R}(t) = \sum_{j=i+1}^{j=n} a_j(t)$ from its right neighbor (the $i + 1$th cube). During this time, we let the ith cube apply the additional contraction force $a_i(t) = \alpha_i$. This will result in the constant acceleration of the center of the ith cube by α_i.

During the second half of the movement time t, for $T/2 < t \leq T$, the ith cube will be subjected to external force $F_{i,L}(t) = \sum_{j=1}^{j=i-1} -a_j(t)$ from its left neighbor and will be subjected to external force $F_{i,R}(t) = \sum_{j=i+1}^{j=n} -a_j(t)$ from its right neighbor. During this time we will let the ith cube apply the additional force $a_i(t) = -\alpha_i$. This will result in the constant deceleration of the center of cube i by $-\alpha_i$.

The above sums describing forces $F_{i,L}(t)$ and $F_{i,R}(t)$ are due to the accumulation of the forces (the individual mass of each cube is unit, so each contributed force is just the jth cube's acceleration) along the chain of cubes (to the left or right, respectively) of the ith cube. Note that the model's axiom on allowed specification of stiffness of the attachments to neighbors of a module is required to ensure forces (external to the module) are transmitted between neighbors of the module. (For example, during the middle time phases of the *Squeeze* algorithm, the middle two modules indexed $n/2$ and $n/2 + 1$ can each only apply a unit maximum force between each other, but there is an accumulated force transmitted between them of a constant times n.) For our *Squeeze* algorithm, this is immediate since the accumulated forces $F_{i,L}(t)$ and $F_{i,R}(t)$ are precisely known for each ith module.

It is useful to observe that $\sum_{j=1}^{j=n} a_j(t) = 0$ due to the definition of the $a_j(t)$ (that is, the center of the overall system does not change x-coordinates). This implies that at all times the total sum of forces involving the the ith cube —

including both external forces as well as forces providing acceleration of the mass of the ith cube — is balanced: $F_{i,L}(t) + F_{i,R}(t) + a_i(t) = \sum_{j=1}^{j=n} a_j(t) = 0$.

In summary, the resulting acceleration of the center point of the ith cube is $a_i(t) = \alpha_i$ for $0 \le t \le T/2$ and is $a_i(t) = -\alpha_i$ for $T/2 < t < T$. Hence, we have made the center of each cube translate in the x-axis on a trajectory exactly the same as the ith point mass of our solution to the Point Mass Contraction Problem. That way, we satisfy the required movement restrictions of the *Squeeze* Problem. That is, for $i = 1, \ldots, n$ the center of the ith cube will move from the initial x-coordinate $x_i(0) = i - [n+1]/2$ starting with initial velocity $v_i(0) = 0$ to final x-coordinate $x_i(T) = x_i(0)/2 = i/2 - [n+1]/4$ and ending with final velocity $v_i(T) = 0$. $\square$

A.2 The Confined Cubes Swapping Problem

Proof of Lemma 3.

Proof: First consider the step of transforming the single row of modules into the intermediate "checkered" configuration. In this step all forces and movement occur in the y-axis direction. Let the number of modules n be even. From the initial row configuration let the even numbered cubes $i = 2, \ldots, n$ be the modules that contract upwards while the odd numbered cubes $i = 1, \ldots, n-1$ are contracted downwards.

As stated above, we wish to begin reconfiguration by keeping fixed the location of top edges of even numbered (upward) modules. We also want to fix the location of bottom edges of odd (downward) modules. This is done by having sliding forces between adjacent modules to balance out contraction forces. By definition an expanding cube module can exert enough force to contract from length 1 to length $1/2$ in O(1) movement time using a unit amount of force while beginning and ending with zero velocity. To also allow for sliding forces, let $1/4$th of each module's force go toward contracting it while the other $3/4$ of its force capability is left available to generate sliding force.

A constant fraction of the original force still permits contraction to occur in O(1) time. Specifically, if a force of 1 caused contraction in time $T = 1$ then from our equations in Section 4 we conclude that a force of $1/4$ permits contraction in time $T = \sqrt{4} = 2$. Thus, set $T = 2$ as the movement time for reconfiguration.

Since the contraction motions were suitably created by applying the constant fraction $1/4$ of the original force, we focus now on the sliding motions by looking at the even numbered (upward) modules. In the initial configuration, even and odd modules have their centers at the same y coordinate. In the checkered configuration, even modules' centers have a y coordinate a distance of $1/2$ above the odd modules' center coordinates. This is created by even modules' centers moving up $1/4$ unit distance and odd modules' centers

moving down 1/4 unit. Alternatively, we can think of it as the even modules' centers moving 1/2 unit distance relative to the odd modules' centers. We focus on this relative separation between the center points of upward and downward moving modules to simplify our analysis. To create this motion let each module $i = 1, \ldots, n - 1$ apply a force of 1/4 to each of its connection boundaries in the proper direction to create sliding motion.

Given these force applications, the resulting force applied to each upward moving (even) module $i = 2, 4, \ldots, n - 2$ comes from 2 such boundaries and a total force of $(1/4 + 1/4) + (1/4 + 1/4) = 1$. The exception is module n that has only 1 such boundary and so only a force of 1/4 contributed from its lone neighbor. This matter is resolved by module n applying all of its available force of 3/4. Now all upward moving modules have a force of 1 applied to them.

With unit masses for each of the modules, from equation (1) in Section 4 we get that a force of 1 will generate an acceleration of 1. Yet, as in previous examples we first need an acceleration in the direction of movement, then a reverse acceleration in the opposite direction to ensure a final velocity of 0. Splitting the force evenly between 2 stages, we will have accelerations with magnitude 1/2. Let $y(t)$ denote an upward moving module's position relative to its neighboring downward moving modules, while $v(t)$ and $a(t)$ denote its velocity and acceleration, respectively. In the first stage, during $0 \leq t \leq T/2$, a(t) = 1/2, and during $T/2 < t \leq T$, a(t) = -1/2. We require only 3 such values $y(t), v(t)$, and $a(t)$ because in this idealized model all modules will be moving in unison.

The equations describing this motion are the same as those found during the work of Section 4. At the final time T the final velocity is given by $v(T) = 1/2(T - T) = 0$ as desired. The final position is given by $y(T) = a(0)T^2/4 = (1/2) * (4/4) = 1/2$ exactly as desired. Finally, we know that the velocity difference between adjacent modules never exceeded a constant bound because velocities themselves never exceeded a constant bound in this movement. Also, note that forces in this system are balanced with the contracting modules absorbing the sliding motion just described to result in the entire system keeping the same bounding box throughout reconfiguration.

For the second reconfiguration step, we perform the reverse of the contraction operation just described. The only difference is instead of the modules expanding in the y-axis direction, they do so in the x-axis direction. By the principle of time reversal described in the Section 1 this operation can be done in exactly the same amount of time and using the same amount of force as the prior operation (and forces are balanced in the same way). Thus we also get that this step takes O(1) time, thereby completing the confined cubes swapping problem in O(1) total time. □

A.3 2D Array Reconfiguration Problem: Proof 1

Proof of Lemma 4.

Proof: (Note that an alternate proof giving more extensive mathematical analysis is given in Appendix A.4.) For simplicity, let n be even. This reconfiguration problem will be completed in three stages. First the initial m rows will be simultaneously contracted from total length n to total length $n/2$. This is done in $O(\sqrt{n})$ time by contracting each module from length 1 to length $1/2$ as shown in Section 4. Note that forces for this contraction problem occur only between modules in the same row and only in the x-axis direction. Therefore the time bound result holds because the time required to contract a given row is independent of the number of parallel rows also contracting. Note that for simplicity, we are dealing with ideal modules and ignore the possibility of "shearing" where one row might expand faster than a neighboring row.

For the second stage we perform the confined cubes swapping problem to turn each of the m rows of n contracted modules into $2m$ vertically stacked rows each having $n/2$ contracted modules. The initial m rows are contracted in the x-axis direction while the final $2m$ rows are contracted in the y-axis direction. In the analysis of this problem given earlier in this section, the motion of a given initial row of modules was shown to be self-contained. The forces of the system were balanced and a given "row" maintains its overall position and dimensions of width and height — even while transforming from 1 row to 2 stacked rows — through careful use of the modules' contraction abilities. Thus, we find that this operation also maintains its previously constructed running time regardless of the number of rows acting in parallel. Hence, this second reconfiguration stage takes $O(1)$ movement time.

Finally, for the third stage we expand the $2m$ rows from having total height m to total height $2m$. This is done by having each module expand from length $1/2$ in the y-axis direction to length 1. Note that forces for this reconfiguration stage occur only in the y-axis direction. So, the expansion of 1 column is independent of the expansion of other columns if we ignore the possibility of shearing. This problem is then the reverse operation of contracting a column or row of $2m$ modules. By the principle of time reversal this expansion operation will take exactly the same amount of time and the same amount of force for reconfiguration as the contraction example. Therefore we can use the result of Section 4 that this operation takes $O(\sqrt{m})$ movement time while still satisfying the bounds for the physical properties of modules.

Combined, we have 3 distinct reconfiguration operations that reconfigure the entire array of modules from the initial configuration to the goal configuration as desired. All of this has been shown to happen while still satisfying the bounds imposed by our abstract model for the modules. Since the 3 stages took $O(\sqrt{n})$, $O(1)$, and $O(\sqrt{m})$ time, added together we get that reconfigur-

ing from an $m \times n$ array of non-contracted modules to an array of $2m \times \frac{n}{2}$ non-contracted modules takes $O(\sqrt{m} + \sqrt{n})$ movement time. $\square$

A.4 2D Array Reconfiguration Problem: Proof 2

Alternate proof of Lemma 4.

Proof: Assign to the modules array coordinates (i, j) for being in row i and column j, counting from the bottom and from the left of the array, respectively. Let $x_{(i,j)}(t)$ and $y_{(i,j)}(t)$ denote the x and y coordinates of module (i, j)'s center at time t. Furthermore, let $v^x_{(i,j)}(t)$ and $a^x_{(i,j)}(t)$ give the velocity and acceleration, respectively, of module (i, j) at time t in the x direction. Let $v^x_{(i,j)}(t)$ and $a^x_{(i,j)}(t)$ do the same in the y direction. This reconfiguration problem will be completed in three stages, all of which have already been analyzed: the *Squeeze* problem, the *confined cubes swapping* problem, and the *Reverse Squeeze* problem.

In the first motion stage (state (a) to state (b) if Figure 5) all m rows contract from length n to length $n/2$ in time T_1. All motion and forces occur in the x direction and the location of module (i, j) is described by $x_{(i,j)}(0) = j - [n+1]/2$ and $x_{(i,j)}(T_1) = x_{(i,j)}(0)/2 = j/2 - [n+1]/4$. Since forces and motion only occur along individual rows (ignoring shearing) we treat each row individually and find the same reconfiguration problem as was solved in the the *Squeeze* problem. Solving in the same way, we first accelerate for $0 \le t \le T_1/2$ and decelerate for $T_1/2 \le t \le T_1$ we set $\alpha^x_{(i,j)} = \frac{n+1-2j}{n-1}$ as the magnitude of that acceleration. As previously found, this leads to a final velocity of $v^x_{(i,j)}(T_1) = \alpha^x_{(i,j)}(T_1 - T_1) = 0$ and a final x coordinate of $x_{(i,j)}(T_1) = j - \frac{n+1}{2} + [\alpha^x_{(i,j)}[T_1]^2]/4$. With our stated values for $\alpha^x_{(i,j)}$, setting $T_1 = \sqrt{n-1}$ gives $x_{(i,j)}(T_1) = j/2 - [n+1]/4 = x_{(i,j)}(0)/2$ as desired. All forces are balanced and unit bound requirements are met just as during our analysis of the *Squeeze* problem.

For the second stage of motion, each row is now independently solving the confined cubes swapping problem. Let the contraction motion take place from $T_1 < t \le T_2$ and the expansion motion take place during $T_2 < t \le T_3$. Let modules in even numbered columns move upward while modules in odd numbered columns move downward. Initially, $y_{(i,j)}(T_1) = i - [m+1]/2$ for all modules. For even numbered modules, if $i \le m/2$ then $y_{(i,j)}(T_2) = i - [m+1]/2 - 1/4$ and if $i > m/2$ then $y_{(i,j)}(T_2) = i - [m+1]/2 + 1/4$. This is because all of these modules move upward, but our location for $y = 0$ is through the middle of the module array. For odd numbered modules if $i \le m/2$ then $y_{(i,j)}(T_2) = i - [m+1]/2 + 1/4$ and if $i > m/2$ then $y_{(i,j)}(T_2) = i - [m+1]/2 - 1/4$.

The shift in horizontal locations during the expansion stage is similar. For modules formerly in even numbered columns with $j \le n/2$ we have $x_{(i,j)}(T_2) = j/2 - [n+1]/4$ and $x_{(i,j)}(T_3) = j/2 - [n+1]/4 + 1/4$ while for columns

with $j > n/2$ it is $x_{(i,j)}(T_3) = j/2 - [n+1]/4 - 1/4$. For modules formerly in odd numbered columns it is the opposite: if $j \leq n/2$ then $x_{(i,j)}(T_3) = j/2 - [n+1]/4 - 1/4$ and if $j > n/2$ then $x_{(i,j)}(T_3) = j/2 - [n+1]/4 + 1/4$. This is because even numbered modules expand to the left while the odd numbered modules under them expand to the right. Note that by the analysis of the confined cubes swapping problem this stage takes time $T_3 - T_1 = O(1)$. Alternatively, we can just note that every modules moves only $O(1)$ distance and so $O(1)$ movement time is to be expected.

Now, from time T_3 to T_4 we need to perform the *Reverse Squeeze* problem and expand the modules in the vertical direction (state (e) to state (f) in Figure 5). The difficulty is that the number of rows has now doubled as modules from even numbered columns moved on top of modules from odd numbered columns. So, for each module (i, j) from an even numbered column, assign it a new row number $k = 2i$. For each module (i, j) from an odd numbered column, assign it $k = 2i - 1$. Now, let $y_k(t)$ be the vertical location of a module in row k. At time $t = T_3$ we have $y_k(T_3) = k/2 - [2m+1]/4$. After expansion, we wish to have $y_k(T_4) = k - [2m+1]/2$. This is exactly the *Reverse Squeeze* problem. So, we can again complete the expansion in to movement stages, one to accelerate and one to decelerate. It can be verified that this can be completed in time $T_4 - T_3 = \sqrt{2m-1}$ (and by the principle of time reversal, this should be expected).

Thus we have completed the reconfiguration task in 3 stages taking $\sqrt{n-1}$, $O(1)$, and $\sqrt{2m-1}$ time, respectively. Thus, we have that the total time for reconfiguration is $O(\sqrt{m} + \sqrt{n})$. $\square$

A.5 A Lower Bound for the *Squeeze* Problem

Proof of Lemma 7.

Proof: Consider the movement of the first n/c cubes $i = 1, 2, \ldots, n/c$. Let c be a real number strictly larger than 2. For $i = 1, \ldots, n$ the center of cube i needs to move distance $|x_i(0) - x_i(T)| = |[n+1]/4 - i/2|$ from the initial x-coordinate $x_i(0) = i - [n+1]/2$ starting with initial velocity $v_i(0) = 0$ to final x-coordinate $x_i(T) = x_i(0)/2 = i/2 - [n+1]/4$ and ending with final velocity $v_i(T) = 0$. Note that this distance is at least

$$\left| \frac{n+1}{4} - \frac{i}{2} \right| \geq n \left[\frac{1}{4} - \frac{1}{2c} \right] + \frac{1}{4}$$

for the first n/c cubes (those with indices from 1 to $\lfloor n/c \rfloor$) and the last n/c cubes (those with indices from $n - \lceil n/c \rceil + 1$ to n). This is a total of at least $n_1 = 2[n/c - 1]$ cubes. The total force applied by all n cubes is at most n, since by assumption each has a force magnitude upper bound 1.

Hence, since they have unit masses, the average acceleration (at their center points) applied to each of these $n_1 = 2[n/c - 1]$ cubes can be at most

$n/n_1 = c/[2[1 - c/n]] \approx c/2$. Thus, at least one of these cubes (say the ith cube) has average acceleration at most $a_i \approx c/2$. But the ith cube needs to traverse a distance of at least $\mid [n + 1]/4 - i/2 \mid \geq n[1/4 - 1/[2c]] + 1/4$.

Moreover, the ith cube must start and end with 0 velocity, so it is easy to verify that the time-optimal trajectory for its center point has an acceleration of at most $a_i \approx c/2$ in the positive x-axis direction from time 0 to $T/2$, followed by a reverse direction acceleration of the same magnitude but in the negative x-axis direction.

The total distance traversed in each of the two stages is at most $a_i t^2/2 = [c/2][T/2]^2/2$, so the total distance traversed by the center of the ith cube is at most $[c/2][T/2]^2 = cT^2/8$ which needs to be $\geq n[1/4 - 1/[2c]] + 1/4$. Thus,

$$\frac{cT^2}{8} \geq n \left[\frac{1}{4} - \frac{1}{2c}\right] + \frac{1}{4}$$

$$T \geq \sqrt{\frac{8n}{c} \left[\frac{1}{4} - \frac{1}{2c}\right] + \frac{2}{c}}$$

$$T \geq \sqrt{\frac{2n}{c} \left[1 - \frac{2}{c}\right] + \frac{2}{c}}$$

$$T = \Omega(\sqrt{n}) \ . \ \square$$