Reconfiguring a Chain of Cubes*

Laurie Heyer † Anna Lubiw ‡ Debajyoti Mondal § Ulrike Stege ¶ Sue Whitesides§

Abstract

Any configuration of a chain of cubes can be transformed into any other while maintaining contact between adjacent cubes in the chain using a quadratic number of moves. We show that this result is also true in the more constrained setting of a "Kibble chain" where the cubes are threaded on an elastic string, and slits in the cubes allow limited turns at joints.

1 Introduction

This paper is about a puzzle shown in Figure 1 that consists of a sequence of n cubes threaded on an elastic string. The string forces cubes that are adjacent in the sequence to remain in contact, but slits in the cubes allow limited rotations. We show that any configuration of such a puzzle can be transformed to any other using $O(n^2)$ steps. A main tool is to show the same result in a more abstract setting without the elastic string or the slits: to reconfigure a chain of cubes while maintaining contact between adjacent cubes.

More generally, reconfiguring a set of disjoint cubes or modules is of interest both for puzzles and for modular robotics. Different constraints on the reconfiguration process arise in different scenarios. In modular robotics, one well-studied constraint is that the set of modules should remain connected. More precisely, the graph with vertices representing modules and edges representing contact between pairs of modules must remain connected. A common model of allowable motions for cube modules is that one cube may "slide" along another cube [3, 5, 6]. Dumitrescu and Pach [11] showed how to reconfigure a set of squares in this model, and Abel and Kominers [2] extended to cubes in three and higher dimensions. More general "mover" problems were surveyed by Dumitrescu [10]. Hurtado et al. [12] considered distributed algorithms. Alternative "pivoting" motions were considered in [15] and can be physically realized. Reconfiguration of rectilinear chains is also relevant for protein folding [13].

Our setting is more constrained in that an ordering $c_1, c_2, \ldots, c_n$ of the cubes is specified, and each cube must remain in contact with its neighbours in the sequence. We show in Section 3 that any configuration of a chain of n cubes can be straightened using $O(n^2)$ slide motions, and hence that any configuration can be transformed into any other with the same bound. This is an easy generalization of a result known in 2D[1]. The idea (similar to those in [7, 13]) is to pull the chain out at a cube that lies on the boundary.

There are a number of physical puzzles made of cubes strung together in a chain. See Figure 2. In a "snake puzzle" each cube has a hole drilled through it, either straight through or making a right angle turn, and the face contacts between successive cubes are determined by the placement of the holes. A snake puzzle cannot be straightened to have all the cubes in a line and deciding whether one configuration can be transformed to another is of open complexity [1].

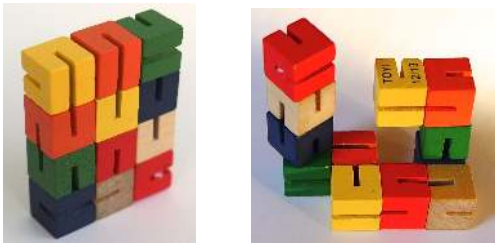

Figure 1: The Wooden Fidget puzzle.

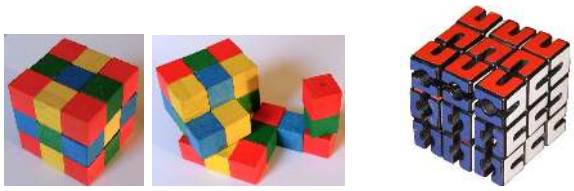

Figure 2: A snake puzzle (left) and a Kibble puzzle (right).

A "Kibble chain" is different from a snake puzzle in that there are slits in each cube that allow the face contacts to change as the elastic cord moves in the slits. It is crucial to have some stretch in the cord, otherwise face contacts are fixed and it becomes a snake puzzle. The commercial Kibble puzzle also involves face colours, but we will not be concerned with colours. A

*INRIA-UVic-McGill 2015 Geometry Workshop at Bellairs

†Davidson College, USA, laheyer@davidson.edu
‡University of Waterloo, Canada, alubiw@uwaterloo.ca
§University of Manitoba, Canada, jyoti@cs.umanitoba.ca
¶University of Victoria, Canada, {ustege,sue}@uvic.ca

[1]Aloupis, Demaine, Lubiw, private communication, from a 2009 Carleton workshop

commercial version of the puzzle we deal with is called "Shapeshifter Creativity blocks" or "Wooden Fidget Puzzle"—see Figure 1. Details of the slits are shown in Figure 3. We show in Section 4 that any configuration of a Kibble chain can be straightened in $O(n^2)$ moves, and hence that any configuration can be transformed into any other. We need rotations in addition to slides. We also show in Section 5 that in some special cases, pivot moves (to be defined) suffice to straighten a chain.

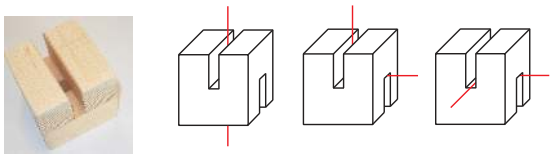

Figure 3: The slits in a single Kibble cube and the ways the string can go through it. (Model and photo: Tom Whitesides.)

Note that we assume we are given the initial and final configurations of the chain. Even in 2D the problem of deciding whether some collection of squares has a chain through it is the known NP-hard problem of finding a Hamiltonian path in a grid graph. For the snake puzzle, even the special case of deciding whether a large cube can be formed from a given snake (ignoring the actual reconfiguration process) is NP-hard [1].

Since we are dealing with a chain, our work is related to linkage reconfiguration [9, 14], and to the version where one joint is straightened at a time [4]. There has also been work on reconfiguring a chain of polygons that must remain connected at fixed points of contact between successive pairs [8]. Our version is different in that two adjacent cubes must remain in contact, but the point of contact may change.

2 Models of Motion

Slides and *pivots* are two basic motions used to reconfigure cubes. The faces of contact between two cubes A and B can be changed with either motion: the face of contact for both A and B can be changed using two slides (see Figure 4); and the face of contact for just one of them can be changed using a pivot (see Figure 5). Both motions can be carried out for some orientations of Kibble cubes.

To define the motions more exactly, we must specify what happens to the rest of the chain. A *slide* translates some cubes that lie along a line of consecutive grid positions (in one of the axis directions) by at most one grid position along the line. All other cubes remain fixed. Of course there must be some clear space for the leading cube to move into, and the constraint of maintaining contact along the chain further restricts the slides

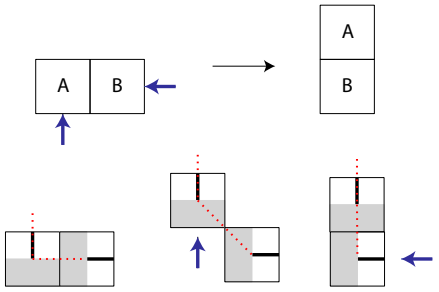

Figure 4: Changing face of contact with slides in the contact model (top)—block A slides up, then block B slides left. These slides can be performed for some orientations of Kibble cubes (bottom).

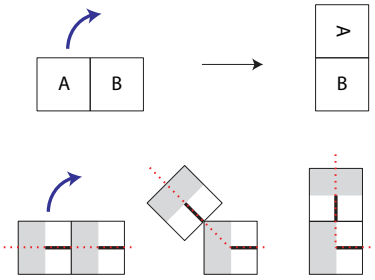

Figure 5: Changing face of contact with a pivot in the contact model (top)—block A pivots around block B. A pivot can be performed for some orientations of Kibble cubes (bottom).

that can be performed. A *rotation* rotates one cube, while all others stay fixed. Rotation may be around an axis through the center or an edge of the rotating cube. Some clear space is needed around a cube before it can be rotated. Finally, we will use the term *pivot* for the operation shown in Figure 5 that translates and rotates a cube and the subchain attached to it. In Figure 6 we show how a chain with one bend can be straightened using one pivot or using a sequence of slides.

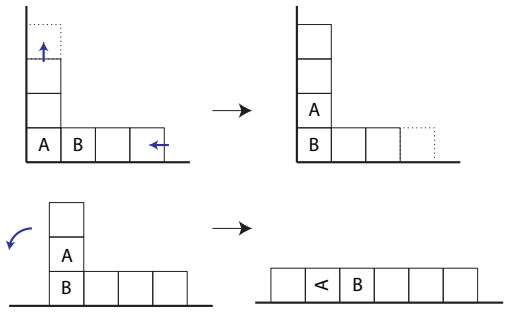

Figure 6: Straightening a chain with one bend using a sequence of slides (top) or a single pivot (bottom).

Note that the pivot requires a lot of free space, but the

sequence of slides only requires that one endpoint of the chain extends to a line in free space. Because of this, we will use pivots only to straighten chains that are initially monotone (Section 5). More generally, we use slides. Slides maintain the axis alignment of each cube. However, it is not possible to straighten every Kibble chain while maintaining axis alignment, for example see Figure 7. We need to use rotations as well. (To straighten this example see Figure 19 in the Appendix.) To rotate a cube about an axis through its center, we must clear some space around it. See Figure 8 for one example of how to do this, and observe that there are alternatives that keep any particular row/column fixed. We also need to clear some space in order to rotate a cube about an axis through an edge. See Figures 14 and 15. We perform these operations so that all required contacts are maintained.

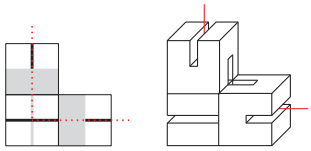

Figure 7: This chain (drawn in schematic and 3D form) cannot be straightened while keeping the cubes aligned with the axes, nor by keeping the cubes in the current plane, nor by a single pivot.

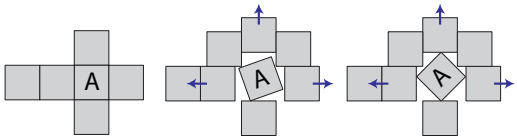

Figure 8: When cube A is connected to a cube in front of the page and to a cube below or behind, A can be rotated after we slide three rows/columns out of the way.

For a Kibble chain we note that sliding a subsequence of the chain increases the length of the string by an additive constant because the two end cubes may change their contact types—when two cubes are in face-to-face contact the length of the piece of string between their centers is 1, but when they are in edge-to-edge contact the length is $\sqrt{2}$. However, sliding a row/column (as may be needed for rotations) may increase the length by a multiplicative constant because a linear number of cube contacts may deviate from full face-to-face contact.

3 Reconfiguring in the Contact Model

In this section we deal with the "contact" model in which adjacent cubes must remain in contact, but we make no further restrictions on their motions. In the initial and final configurations, we assume complete face-to-face contact between cubes that are adjacent in the chain. During reconfiguration, most contacts will still be face-to-face contacts, but we allow edge-to-face or edge-to-edge contact.

The idea is to "pull" the chain out at a cube (or pair of cubes) on the boundary. The rest of the chain moves along the paths to these boundary cubes. We call this the "snake in a tunnel" method. See Figure 9. This idea was used by Cheung et al. [7] who showed how to reconfigure any 3D shape to any other by subdividing it into "micropixels" (spheres) joined in a Hamiltonian path with an endpoint on the boundary, and then pulling the chain out at this endpoint. The same idea was used earlier in work by Lesh et al. [13] for protein folding (with a slightly different set of elementary moves), where it was called a "reptation" method in keeping with a basic principle of polymer science.

Theorem 1 *Any configuration of a chain of n cubes with face-to-face contacts between cubes adjacent in the chain can be transformed to any other such configuration using $O(n^2)$ slides, while maintaining contact between adjacent cubes in the chain.*

Proof. As noted in the Introduction, this proof was already known for 2D. It suffices to show that slides can transform any configuration into a straight chain which then acts as an intermediate "canonical" configuration.

Consider a cube c in the topmost plane of the configuration that is connected to a cube below this plane. (If no such c exists then all the cubes lie on one plane, and we consider the rightmost column instead of the topmost plane.) Either c is an end cube of the chain, or else there is a cube c' that is adjacent to c in the chain and also lies in the topmost plane. If c is an end cube, the idea is to "pull" the chain out at c until it is straight. If c is not an end cube then we pull c and c' out until the two subchains from c and c' to the ends of the chain are straight.

Let $\chi(c)$ and $\chi(c')$ be the disjoint subchains from c and c', respectively, to the ends of the chain. A *phase* moves $\chi(c)$ or $\chi(c')$ along its tunnel by one grid position. If c' exists, the phases alternate between c' and c. At the end of each phase all contacts are face-to-face. Each phase is implemented as a sequence of slides, one for each bend in the subchain. Each slide moves a maximal subchain of collinear cubes (a "line" of cubes) along by one grid position, effectively transferring an empty position from the front of the line to the end of the line. We begin each phase by moving c or c' upwards, along with a maximal vertical subchain below it. This leaves a vacant grid position where the last cube of the subchain was. In general suppose that cube c_i has vacated a position at the front of line ℓ that consists of cubes

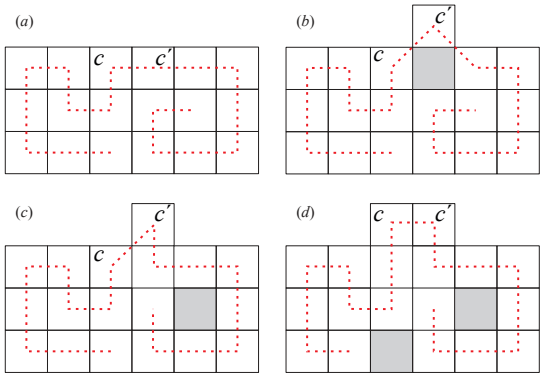

Figure 9: The snake in a tunnel method in 2D: (a) initial configuration; (b) in the first step cube c' moves up (in this example, c' is alone in its line); (c) at the end of the first phase $\chi(c')$ has moved one position further in its tunnel; (d) in the second phase c and its line move up and $\chi(c)$ moves one position further in its tunnel.

$c_{i+1} \ldots, c_j$, $j \geq i$. Cube c_{i+1} is in edge-to-edge contact with c_i. We move the cubes in line ℓ so that c_{i+1} enters the empty position, regaining face-to face-contact with c_i. Since the chain turns at c_j, the contact between c_j and c_{j+1} becomes edge-to-edge.

At the end of $O(n)$ phases $\chi(c)$ will be in one line. If $\chi(c')$ exists, it will also be in one line, and we can do a sequence of $O(n)$ slides to move the two chains into one line. Each phase takes $O(b)$ slides, where b is the number of bends in the chain, so the total number of slides is $O(nb)$ which is in $O(n^2)$. □

4 Reconfiguring a Kibble Chain

The theorem in the previous section does not apply to Kibble chains because motions are restricted by the positions of the slits. Reconfiguration is possible if we allow rotations as well as slides:

Theorem 2 *Any configuration of a Kibble chain of n cubes with face-to-face contacts between cubes adjacent in the chain can be transformed to any other such configuration using $O(n^2)$ rotations/slides, while maintaining contact between adjacent cubes in the chain.*

Proof. We follow the same plan as above, but must rotate cubes in order to re-orient the slits. Recall that a *phase* means moving c (or c') and its subchain along by one position, and each phase is composed of *steps*, where each step involves sliding a line of cubes along by one grid position.

We first claim that before each phase, we can rotate each cube in $\chi(c)$ and $\chi(c')$ (except the initial cubes c and c') so that: (1) the string enters the middle of a slit; and (2) if the string turns 90° in the cube then the string exits at the end of a slit (Figure 3, middle, with the string entering from the top). To justify this, observe that the other orientations in Figure 3 can be rotated to satisfy this.

We now show how to implement each step in the general situation. The first two phases when c and c' move up the first time need some extra care, and we discuss them later. The general situation is that we have an empty grid position at the front of a line ℓ of cubes. Suppose that X is the cube that just vacated the empty position, Y is the first cube of line ℓ, and Z is the cube after Y. Then Z is part of ℓ unless the string turns in Y. After appropriately rotating our frame of reference, the configuration of X and Y is as shown in Figure 10.

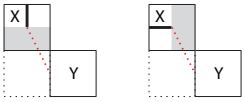

Figure 10: The two proper configurations of cubes X and Y before a step. The dotted position is empty.

For the rest of our argument, we will assume this configuration and speak of "above", "in front", etc. We will implement steps in such a way as to ensure that X is in one of two orientations, as shown in Figure 10. We call these *proper* configurations. In particular, note that if X moved upward via a slide, then we have the first proper configuration.

Because the string enters the middle of a slit of cube Y, there are two possibilities for Y—the string enters a vertical slit or a horizontal slit—see Figure 11. We consider each of them.

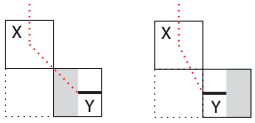

Figure 11: The slit in Y may be vertical (left) or horizontal (right).

Case 1. The string enters Y on a vertical slit. Cube Z can lie in one of three possible positions: (a) in line ℓ; (b) behind Y; (c) in front of Y. See Figure 12. In all cases, we slide the cube(s) of line ℓ along by one position, which moves Y into the empty position. Observe that the resulting configuration is proper in all cases.

Case 2. The string enters Y on a horizontal slit. The next cube, Z, can lie in one of three possible positions as shown in Figure 13. We consider these in turn.

(a) Z lies in line ℓ. Rotate cube Y around the axis of line ℓ. This yields Case 1(a).

(b) Z lies below Y. As shown in Figure 14, rotate cube Y into the empty position maintaining contact with X

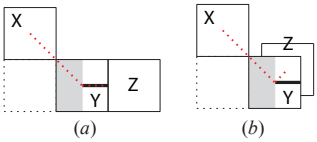

Figure 12: Positions for Z in Cases 1(a) and (b).

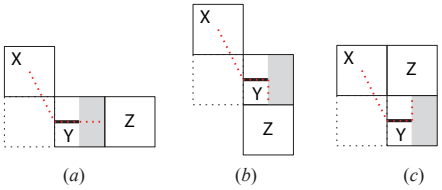

Figure 13: Positions for Z in Case 2.

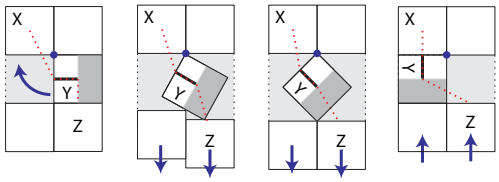

Figure 14: Handling Case 2(b) by rotating Y. Empty space is lightly shaded. Observe that instead of making room for the rotation by pushing the lower cubes downward, we could push the upper ones upward.

and Z. In order to do this, we pull two columns of cubes downward. Observe that the resulting configuration is proper.

(c) Z lies above Y. This "U-turn" is the trickiest case. We may find it necessary to do two steps at once. Note that we cannot have two U-turns in a row.

Rotate cube Y into the empty position as in the previous case (see Figure 15(a)). The resulting configuration is not proper. The bottom of cube Z may have a side-to-side slit or a front-to-back slit. In the first case (see Figure 15(b)), we slide Z (and any cubes in its line) downward, observing that the resulting configuration is proper. In the second case (see Figure 15(c)), the next cube Z' may lie above Z or to the right of Z, but it cannot lie to the left of Z because X is already there. If Z' lies above then rotate cube Z so it has a side-to-side slit, which we just saw how to handle. Finally, if Z' lies to the right of Z then rotate Z into the empty position. Observe that the resulting configuration is proper.

There are two issues outstanding. One is the orientation of c and c' in the initial phases. We rotate them into the orientations shown in Figure 16. In the first phase c' moves up and in the second phase c moves up. At the beginning of each phase we have a proper configuration.

The other issue is that during odd-numbered phases, c

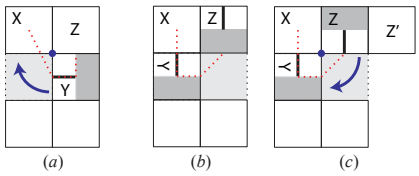

Figure 15: Handling Case 2(c) by rotating Y (a) and then either moving Z down (b) or rotating it (c).

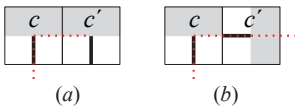

Figure 16: The initial orientations of c and c': (a) if the string turns in c'; (b) if the string goes straight through c'.

and c' will only be in edge-to-edge contact, which poses some difficulties for sliding an axis-parallel "column" of cubes to accommodate rotations. During phase 1 if we want to slide a non-vertical column that contains c, we simply slide c' as well, thus maintaining the edge-to-edge contact. The other difficulty, which may arise in any odd-numbered phase, is sliding the vertical column that contains c' upward. We note that in all cases of rotations we have the freedom to avoid sliding one particular column.

Our method uses $O(nb)$ steps in total, and requires the string to stretch by a small constant factor. □

5 Straightening a Chain with Minimum Moves

The methods in the previous sections used a quadratic number of moves to straighten a chain of cubes. One would hope that a linear number of moves would suffice, although we have not been able to prove this, nor to find a lower bound larger than b, the number of bends in the chain. In this section we consider configurations where b moves or $O(b)$ moves suffice.

The only way to straighten a chain with b moves is to use one pivot per bend. The pivot can happen on either face of the cube where the bend occurs—for example, in Figure 6(bottom) the pivot could instead be done between cube B and its right neighbour. Except for this choice, the straight sections of the chain act as rigid fixed-length segments. There is relevant work by Arkin et al. [4] on straightening a chain of line segments by straightening one joint at a time. They showed that the decision problem is weakly NP-hard even for a 2D rectilinear chain. This does not carry over to our situation because weak NP-hardness is incompatible with representing segments by unit cubes. Is it NP-hard to decide if a chain of cubes can be straightened in b moves?

Arkin et al. also considered the special case where

joints must be opened in order along the chain. It is easy to test if this works because the motion of the chain is completely determined. Similarly, for a chain of cubes we can test in polynomial time if the bends can be straightened in order along the chain via pivots.

We can make the same test for Kibble chains. In this case not every bend can be straightened with one pivot. Consider a bend in a Kibble chain, and change the frame of reference so that the bend is an L-shape as in the figure below. There are six possible orientations of the slits in the central cube: two of them cannot occur in this L-shaped bend; the two *full-slit* orientations shown in Figure 17 are impossible to straighten with one pivot; and the remaining two orientations, and the pivots that straighten them, are shown in Figure 18.

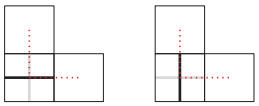

Figure 17: These *full-slit* orientations cannot be straightened with one pivot.

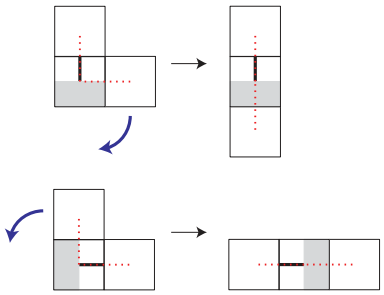

Figure 18: These two orientations of a bend in a Kibble chain can be straightened with one pivot each.

By straightening bends in order along the chain, we can prove that some chains with monotonicity properties can be straightened (details in Appendix):

Lemma 3 *Any 2D or 3D Kibble chain that is monotone in all but one of the axis directions can be straightened with $b + f$ pivots, where b is the number of bends and f is the number of full-slit bends. In the case of a 2D chain with full-slit bends, the height must expand from 1 to $\sqrt{2}$ in the third dimension.*

References

[1] Z. Abel, E. D. Demaine, M. L. Demaine, S. Eisenstat, J. Lynch, and T. B. Schardl. Finding a Hamiltonian path in a cube with specified turns is hard. *Journal of Information Processing* 21(3):368–377, 2013, doi:10.2197/ipsjjip.21.368.

[2] Z. Abel and S. D. Kominers. Universal reconfiguration of (hyper-) cubic robots, http://arxiv.org/abs/0802.3414. arxiv preprint, 2008.

[3] G. Aloupis, N. Benbernou, M. Damian, E. D. Demaine, R. Flatland, J. Iacono, and S. Wuhrer. Efficient reconfiguration of lattice-based modular robots. *Computational Geometry* 46(8):917–928, 2013, doi:10.1016/j.comgeo.2013.03.004.

[4] E. M. Arkin, S. P. Fekete, and J. S. Mitchell. An algorithmic study of manufacturing paperclips and other folded structures. *Computational Geometry* 25(1):117–138, 2003, doi:10.1016/S0925-7721(02)00133-5.

[5] N. M. Benbernou. *Geometric algorithms for reconfigurable structures*. Ph.D. thesis, MIT, 2011.

[6] Z. Butler, K. Kotay, D. Rus, and K. Tomita. Generic decentralized control for lattice-based self-reconfigurable robots. *The International Journal of Robotics Research* 23(9):919–937, 2004, doi:10.1177/0278364904044409.

[7] K. C. Cheung, E. D. Demaine, J. R. Bachrach, and S. Griffith. Programmable assembly with universally foldable strings (moteins). *IEEE Transactions on Robotics* 27(4):718–729, 2011, doi:10.1109/TRO.2011.2132951.

[8] R. Connelly, E. D. Demaine, M. L. Demaine, S. P. Fekete, S. Langerman, J. S. Mitchell, A. Ribó, and G. Rote. Locked and unlocked chains of planar shapes. *Discrete & Computational Geometry* 44(2):439–462, 2010, doi:10.1007/s00454-010-9262-3.

[9] R. Connelly, E. D. Demaine, and G. Rote. Straightening polygonal arcs and convexifying polygonal cycles. *Discrete & Computational Geometry* 30:205–239, 2003, doi:10.1007/s00454-003-0006-7.

[10] A. Dumitrescu. Mover problems. *Thirty Essays on Geometric Graph Theory*, pp. 185–211. Springer, 2013, doi:10.1007/978-1-4614-0110-0_11.

[11] A. Dumitrescu and J. Pach. Pushing squares around. *Graphs and Combinatorics* 22(1):37–50, 2006, doi:10.1007/s00373-005-0640-1.

[12] F. Hurtado, E. Molina, S. Ramaswami, and V. Sacristán. Distributed reconfiguration of 2D lattice-based modular robotic systems. *Autonomous Robots* pp. 1–31, 2015, doi:10.1007/s10514-015-9421-8.

[13] N. Lesh, M. Mitzenmacher, and S. Whitesides. A complete and effective move set for simplified protein folding. *Proceedings International Conference on Research in Computational Molecular Biology*, pp. 188–195, 2003, doi:10.1145/640075.640099.

[14] I. Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. *Symposium on Foundations of Computer Science (FOCS)*, pp. 443–453, 2000, doi:10.1109/SFCS.2000.892132.

[15] C. Sung, J. Bern, J. Romanishin, and D. Rus. Reconfiguration planning for pivoting cube modular robots. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015. to appear.

Appendix

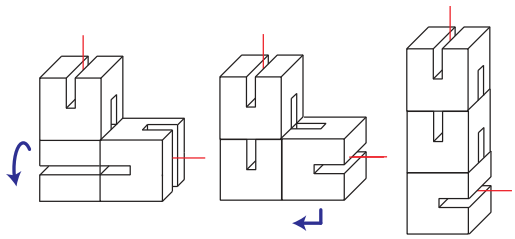

Figure 19: To straighten the chain in Figure 7, rotate the bottom two cubes around the horizontal (side-to-side) axis, and then slide (or pivot) the right-hand cube to the bottom.

Proof. [of Lemma 3] Suppose that the chain is monotone in the x-direction and, in the case of a 3D chain, also monotone in the y-direction. Direct the chain so that a sequence of cubes in a line increases in the x (and y) directions. We will straighten bends one at a time in order along the chain. In the general step, let ℓ be the initial straight portion of the chain (up to the first bend at block b) and let R be the rest of the chain. Ignoring slits for the moment, observe that if we straighten bends one at a time by keeping R fixed and moving ℓ, then R always lies in certain quadrants/octants relative to placing the origin at block b. In 2D, R will lie in the two quadrants where x is positive. In 3D, R will lie in the two octants where x and y are positive.

Bends such as those in Figure 18 can be straightened with one pivot. To deal with a full-slit bend we first rotate the cubes of ℓ around the axis of ℓ. See Figure 19 for an example (where ℓ is horizontal). Note that this motion is actually a pivot (by changing our frame of reference). Observe that this can be done without entering the quadrants/octants that R lies in. This converts the full-slit bend to a standard one, which requires one pivot. □