Optimal In-Place Compaction of Sliding Cubes

Irina Kostitsyna[1,2], Tim Ophelders[*2,3], Irene Parada[4], Tom Peters[2], Willem Sonke[2], and Bettina Speckmann[2]

1 **KBR at NASA Ames Research Center, USA**
irina.kostitsyna@nasa.gov
2 **TU Eindhoven, The Netherlands**
[t.peters1|w.m.sonke|b.speckmann]@tue.nl
3 **Utrecht University, The Netherlands**
t.a.e.ophelders@uu.nl
4 **Universitat Politècnica de Catalunya, Spain**
irene.parada@upc.edu

Abstract

The sliding cubes model is a well-established theoretical framework that supports the analysis of reconfiguration algorithms for modular robots consisting of face-connected cubes. As is common in the literature, we focus on reconfiguration via an intermediate canonical shape. Specifically, we present an in-place algorithm that reconfigures any n-cube configuration into a compact canonical shape using a number of moves proportional to the sum of coordinates of the input cubes. This result is asymptotically optimal and strictly improves on all prior work. Furthermore, our algorithm directly extends to two dimensions and any dimension higher than three.

Related Version A full version of the paper is available at arxiv.org/abs/2312.15096.

1 Introduction

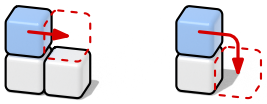
Modular robots consist of a large number of comparatively simple robotic units. These units can attach and detach to and from each other, move relative to each other, and in this way form different shapes or configurations. This shape-shifting ability allows modular robots to robustly adapt to previously unknown environments and tasks. In this paper, we study the *sliding cube model*, a well-established theoretical framework that supports the analysis of reconfiguration algorithms for modular robots consisting of face-connected cubes.

Almost 20 years ago, Dumitrescu and Pach [5] showed that the sliding cube model in 2D (or *sliding square model*) is universally reconfigurable. More precisely, they presented an algorithm that transforms any two given configurations with n squares into each other in $O(n^2)$ moves. This algorithm transforms any given configuration into a canonical shape (a horizontal line) and then reverts the procedure to reach the final configuration. Recently, Akitaya et al. [3] presented Gather&Compact: an input-sensitive in-place algorithm which uses $O(Pn)$ moves, where P is the maximum among the perimeters of the bounding boxes of the initial and final configurations. The authors also show that minimizing the number of moves required to reconfigure is NP-hard.

Until recently, the most efficient algorithm for the reconfiguration problem in 3D was the algorithm by Abel and Kominers [1], which uses $O(n^3)$ moves to transform any n-cube configuration into any other n-cube configuration. As is common in the literature, this algorithm reconfigures the input into an intermediate canonical shape. Stock et al. [7]

* T. Ophelders is partially supported by the Dutch Research Council (NWO) under project no. VI.Veni.212.260.

40th European Workshop on Computational Geometry, Ioannina, Greece, March 13–15, 2024.
This is an extended abstract of a presentation given at EuroCG'24. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.



(a) (b)

Figure 1 Moves in the sliding cube model: **(a)** slide and **(b)** convex transition.

recently announced a worst-case bound of $O(n^2)$ moves for the Abel and Kominers algorithm. Furthermore, their paper presents an in-place reconfiguration algorithm, which runs in time proportional to a measure of the size of the bounding box times the number of cubes. Specifically, their algorithm requires $O(n(wd+h))$ moves in the worst-case, where w, d, and h are the width, depth, and height of the bounding box, respectively.

In this paper we present an in-place algorithm that reconfigures any n-cube configuration into a compact canonical shape using a number of moves proportional to the sum of coordinates of the input cubes. This result is asymptotically optimal. Furthermore, our algorithm directly extends to squares in two dimensions and to hypercube reconfiguration in dimensions higher than three.

2 Preliminaries

A *configuration* $\mathcal{C}$ is a subset of coordinates in the three-dimensional grid. The elements of $\mathcal{C}$ are called *cubes*. We call two cubes *adjacent* if they lie at unit distance. For a configuration $\mathcal{C}$, denote by $G_{\mathcal{C}}$ the graph with vertex set $\mathcal{C}$, whose edges connect all adjacent cubes. We say a *cell* is a vertex of $G_{\mathbb{Z}^3}$ which is not occupied by a cube in $\mathcal{C}$. We always require a configuration to remain *connected*, that is, $G_{\mathcal{C}}$ must be connected. For ease of exposition we assume $\mathcal{C}$ consists of at least two cubes. We call a configuration $\mathcal{C}$ *nonnegative* if $\mathcal{C} \subseteq \mathbb{N}^3$. Cubes can perform *moves*. A move is an operation that replaces a single cube $c \in \mathcal{C}$ by another cube $c' \notin \mathcal{C}$. Moves come in two types: *slides* and *convex transitions* (see Figure 1). In both cases, we consider a 4-cycle γ in $G_{\mathbb{Z}^3}$. For slides, exactly three cubes of γ are in $\mathcal{C}$; c' is the cell of γ not in $\mathcal{C}$, and c is adjacent to c'. For convex transitions, γ has exactly two adjacent cubes in $\mathcal{C}$; c is one of these two cubes, and c' is the vertex of γ not adjacent to c. The slide or convex transition is a *move* if and only if $\mathcal{C} \setminus \{c\}$ is connected.

Let $\mathcal{C}$ be a nonnegative configuration. Call a cube $c = (x, y, z)$ *finished* if the cuboid spanned by the origin and c is completely in $\mathcal{C}$, that is, if $\{0, \ldots, x\} \times \{0, \ldots, y\} \times \{0, \ldots, z\} \subseteq \mathcal{C}$. We call $\mathcal{C}$ *finished* if all cubes in $\mathcal{C}$ are finished. The *compaction problem* starts with an arbitrary connected configuration $\mathcal{C}$ and is solved when all cubes are finished.

Most of the algorithm works on vertical contiguous strips of cubes in $\mathcal{C}$ called subpillars. More precisely, a *subpillar* is a subset of $\mathcal{C}$ of the form $\{x\} \times \{y\} \times \{z_b, \ldots, z_t\}$. In the remainder of this paper, we denote this subpillar by $\langle x, y, z_b \,..\, z_t \rangle$. The cube (x, y, z_t) is called the *head*, and the remainder $\langle x, y, z_b \,..\, z_t - 1 \rangle$ is called the *support*. A *pillar* is a maximal subpillar, that is, a subpillar that is not contained in any other subpillar. Note that there can be multiple pillars with the same x- and y- coordinate above each other, as long as there is a gap between them. Two sets S and S' of cubes are *adjacent* if S contains a cube adjacent to a cube in S'. All omitted proofs can be found in the full version.

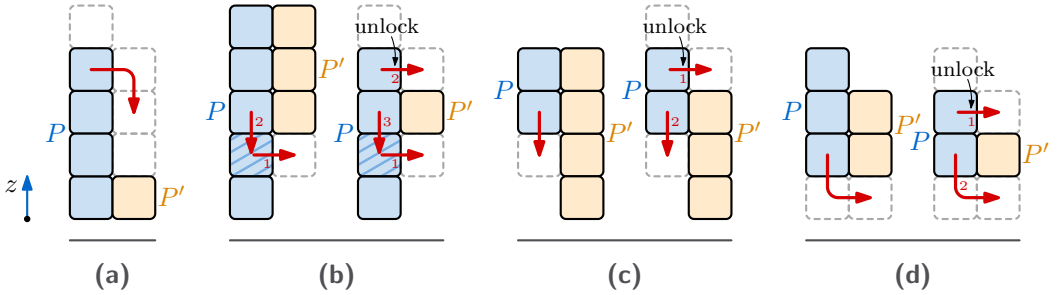


Figure 2 Examples of operations **(a–d)**; hatched cubes are non-cut and dashed outlines indicate cells that must be empty. Each case admits a move sequence that reduces $Z_{\mathcal{C}}$.

3 Algorithm

For a set of cubes $S \subseteq \mathcal{C}$, let (X_S, Y_S, Z_S) denote its *coordinate vector sum* $\sum_{(x,y,z)\in S}(x, y, z)$. Let $\mathcal{C}_{>0}$ be the subset of cubes $(x, y, z) \in \mathcal{C}$ for which $z > 0$, and $\mathcal{C}_0$ be the subset of cubes for which $z = 0$. Let the *potential* of a cube $c = (c_x, c_y, c_z)$ be $\Pi_c = w_c(c_x + 2c_y + 4c_z)$, where the *weight* w_c depends on the coordinates of c in the following way. If $c_z > 1$, then $w_c = 5$; if $c_z = 1$, then $w_c = 4$. If $c_z = 0$, then w_c depends on c_y. If $c_y > 1$, then $w_c = 3$; if $c_y = 1$, then $w_c = 2$; lastly, if $c_z = c_y = 0$, then $w_c = 1$. We aim to minimize the *potential function* $\Pi_{\mathcal{C}} = \sum_{c\in\mathcal{C}} \Pi_c$. From now on, let $\mathcal{C}$ be an unfinished nonnegative configuration. We call a sequence of m moves *safe* if the result is a nonnegative instance $\mathcal{C}'$, such that $\Pi_{\mathcal{C}'} < \Pi_{\mathcal{C}}$ and $m = O(\Pi_{\mathcal{C}} - \Pi_{\mathcal{C}'})$. This means that the sequence of moves reduces the potential by at least some constant fraction of m by going from $\mathcal{C}$ to $\mathcal{C}'$. We show that if $\mathcal{C}$ is unfinished, it always admits a safe move sequence.

The main idea is as follows. For a configuration $\mathcal{C}$, whenever possible, we try to reduce $Z_{\mathcal{C}}$. If that is not possible, the configuration must admit a *pillar shove* where a complete pillar is moved to a different x- and y-coordinate. By reducing either the z-coordinate of cubes, or the x- or y-coordinate, we guarantee that eventually every cube becomes finished.

Local Z reduction. Let $P = \langle x, y, z_b .. z_t\rangle$ be a subpillar of $\mathcal{C}$. We refer to the four coordinates $\{(x-1, y), (x+1, y), (x, y-1), (x, y+1)\}$ as the *sides* of P. On each side, P may have one or more adjacent pillars. We order these by their z-coordinates; as such, we may refer to the top- or bottommost adjacent pillar on a side of P. We say that a set of cubes $S \subseteq \mathcal{C}$ is *non-cut* if $G_{\mathcal{C}\setminus S}$ is connected or empty.

Let $P = \langle x, y, z_b .. z_t\rangle$ be a non-cut subpillar, and let $P' = \langle x', y', z'_b .. z'_t\rangle$ be a pillar adjacent to P. We define a set of operations of at most three moves within P which locally reduce $Z_{\mathcal{C}}$ (see Figure 2). Because P is non-cut, $\mathcal{C} \setminus P$ is connected. Therefore, if cubes of P move in such a way that each component (of cubes originating from P) remains adjacent to a cube of $\mathcal{C} \setminus P$, then the result of that operation is a valid configuration. These different operations **(a–d)** can be seen in Figure 2. For a complete definition of these operations **(a–d)**, see the full version of this paper.

Pillar shoves. Next, we consider a longer move sequence **(e)** that still involves a single subpillar. We call this operation a *pillar shove*, which takes as parameters a subpillar $P = \langle x, y, z_b .. z_t\rangle$ and a side (x', y') of P. The result of the pillar shove is the set of cubes

$$\text{shove}(\mathcal{C}, P, (x', y')) \;:=\; (\mathcal{C} \setminus P) \;\cup\; \langle x', y', z_b .. z_t - 1\rangle \;\cup\; \{(x, y, z_b)\},$$

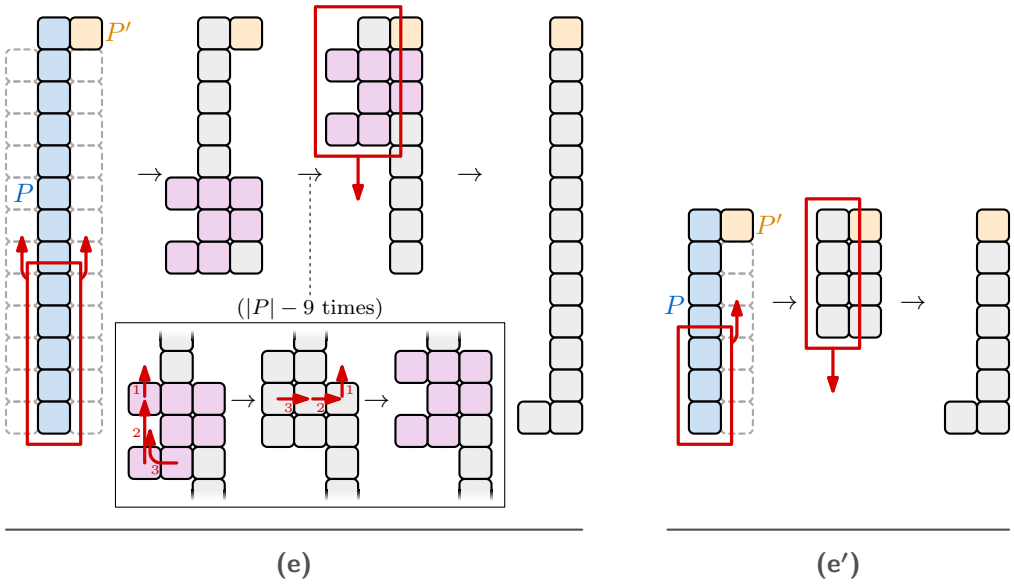


Figure 3 Examples of pillar shoves for a long pillar **(e)** and a short pillar **(e′)**.

in which the support is effectively shifted to the side (x', y'), and the head is effectively moved from (x, y, z_t) to (x, y, z_b). Although shove$(\mathcal{C}, \langle x, y, z_b \,..\, z_t\rangle, (x', y'))$ is well-defined, it is not necessarily a connected configuration, let alone safely reachable from $\mathcal{C}$.

Let $P = \langle x, y, z_b \,..\, z_t\rangle$ be a non-cut subpillar, and assume that on at least two sides (x', y') and (x'', y'') of P, no cube except possibly the head (x, y, z_t) has an adjacent cube. Moreover, assume that $(x', y', z_t) \in \mathcal{C}$. Then the pillar shove can be done without collisions and while keeping connectivity (see Figure 3). There are two cases: one where P has at least 9 cubes, in which case we take $O(|P|)$ moves (left side of Figure 3), and the case where P has fewer than 9 cubes takes $O(1)$ moves and does not require the existence of the second side (x'', y'') (right side of Figure 3). A pillar shove reduces $Z_{\mathcal{C}}$ by $z_t - z_b$ and takes $O(z_t - z_b)$ moves, so it is safe. For a complete definition of **(e)**, see the full version.

Lastly, we define an operation **(f)** that performs any move of $\mathcal{C}$ that moves a cube of $\mathcal{C}_{>0}$, reduces the potential, and results in a nonnegative instance. In summary, the moves **(a–f)** are designed to reduce the z-coordinate of a cube. If this is not directly possible, the pillar shove moves a complete pillar such that the head of that pillar can still reduce its z-coordinate.

Low and high components. Suppose that **(a–f)** do not apply. Let $\mathcal{LH}_{\mathcal{C}}$ be the bipartite graph obtained from $G_{\mathcal{C}}$ by contracting the components of $G_{\mathcal{C}_0}$ and $G_{\mathcal{C}_{>0}}$ to a single vertex (see Figure 4). We call $\mathcal{LH}_{\mathcal{C}}$ the *low-high graph* of $\mathcal{C}$, and we call the vertices of $\mathcal{LH}_{\mathcal{C}}$ that correspond to components of $G_{\mathcal{C}_0}$ and $G_{\mathcal{C}_{>0}}$ *low* and *high* components, respectively. The full version of this paper proves the following lemma.

▶ **Lemma 3.1.** *Assume $\mathcal{C}$ does not admit any operation of type* **(a–f)**. *If H is a high component such that $\mathcal{C} \setminus H$ is connected, then every pillar of H is part of a pillar of $\mathcal{C}$ starting at $z = 0$, H consists entirely of finished cubes, and H contains $(0, 0, 1)$.*

We pick a vertex R of $\mathcal{LH}_{\mathcal{C}}$ that we call the *root* of $\mathcal{LH}_{\mathcal{C}}$. If $(0, 0, 0) \in \mathcal{C}$, pick R to be the low component that contains $(0, 0, 0)$. Otherwise, pick R to be an arbitrary low component.

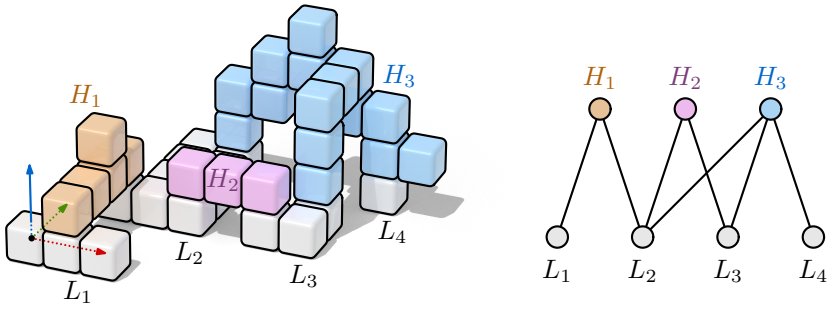


Figure 4 An example configuration $\mathcal{C}$ and its low-high graph $\mathcal{LH}_\mathcal{C}$. This configuration does admit moves of type **(a–f)**

We call a low component L *clear* if $\mathcal{C} \setminus L$ is connected, $L \neq R$, and L is connected to a non-cut pillar P in $\mathcal{C} \setminus L$. We call such a pillar P a *clearing pillar*. The full version proves that such a clear low component always exists, unless $\mathcal{LH}_\mathcal{C}$ contains only the root and possibly a single high component.

We now define operation **(g)**: pick a clear low component L, and perform any move of $\mathcal{C}$ that moves a cube of L, reduces the potential, and results in a nonnegative instance. This does the same as operation **(f)**, but now on $\mathcal{C}_0$ instead of $\mathcal{C}_{>0}$. When operations of type **(g)** are executed, one of three special events could occur:

(1) The component connects to a different low component, merging them.
(2) The component connects to the root, and becomes part of the root.
(3) The component reaches the origin (at which point it becomes the root).

If none of the operations **(a–g)** are available then L is too small to reach the origin. We would like to move the clearing pillar P and do a pillar shove. However, it could be that there are cubes around P, or that moving P would disconnect the low component. For this specific case we define two last operations. Operation **(h)** applies when the bottom of P is completely surrounded. It moves the cube at the bottom of P via $z = -1$ to a positive cell that is closer to the origin. The second operation **(i)** applies when the bottom of P is not sufficiently connected to L and moving it would disconnect L. In this case, we gather cubes from the low component towards P to connect it to the low component, and perform a pillar shove on it, see Figure 5.

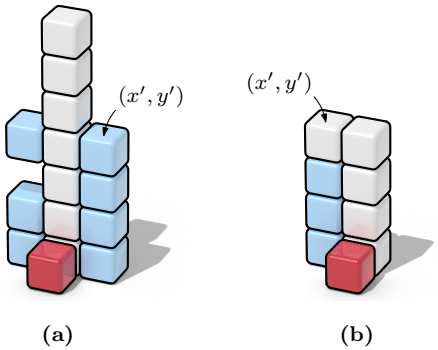


Figure 5 The start configuration for a pillar shove for a clearing pillar. The white pillar is the clearing pillar. The red cube is part of L. The blue cubes are required and need to be gathered. (a): clearing pillar of height at least 5. (b): The configuration for a pillar shove of height at most 4.

This last operation moves cubes that are not part of the clearing pillar $P = \langle x, y, z_b .. z_t \rangle$. However, the move is still safe. Recall that the *potential* of a cube $c = (c_x, c_y, c_z)$ is $\Pi_c = w_c(c_x + 2c_y + 4c_z)$, where w_c is the *weight* of c. The potential of the complete configuration is the sum of potential of the individual cubes. Because the low component cannot reach the origin, its size is at most $O(x + y)$. Therefore, moving a constant number of cubes towards this pillar also only takes $O(x + y)$ moves. This is charged to the head of P: since it goes from $z > 1$ to $z = 1$, or from $z = 1$ to $z = 0$, its weight decreases by 1, paying for the gathering of these cubes.

This algorithm terminates when no clear low component (and hence only the root low component) remains. We are left with two cases. Either no high component remains, or there is at most one high component, which consists of entirely finished cubes.

All of the moves **(a–i)** not only work in 3D, they also work in 2D when instead of prioritizing reducing the z-coordinate, we prioritize reducing the y-coordinate. Moreover, these moves never move the origin. Therefore, we can now run the exact same moves on the bottom layer in 2D, until the root component is finished. If there is still a high component, it stays connected via the origin. We end up with a finished configuration.

Recall that a sequence of m moves on an instance $\mathcal{C}$ is *safe* if the result is a nonnegative instance $\mathcal{C}'$, such that $\Pi_{\mathcal{C}'} < \Pi_{\mathcal{C}}$ and $m = O(\Pi_{\mathcal{C}} - \Pi_{\mathcal{C}'})$. Since each of our operations is safe, the total number of moves our algorithm performs is $O(\Pi_{\mathcal{C}}) = O(X_{\mathcal{C}} + Y_{\mathcal{C}} + Z_{\mathcal{C}})$.

4 Conclusion

We presented an in-place algorithm that reconfigures any configuration of cubes into a compact canonical shape using a number of moves proportional to the sum of coordinates of the input cubes. This result is asymptotically optimal. However, just as many other algorithms in the literature, our bounds are amortized in the sense that we make use of a number of dedicated cubes which help other cubes move by establishing the necessary connectivity in their neighborhood. This is in particular the case with our pillar shoves, that need some additional cubes to gather at the pillar, to then move up and down the pillar to facilitate moves. These extra moves are charged to one cube in the pillar reducing its coordinates. In the literature such cubes are referred to as *helpers*, *seeds*, or even *musketeers* [2, 4, 6, 7]. Hence, an interesting question is whether it is possible to arrive at sum-of-coordinates bounds without amortization?

References

1 Zachary Abel and Scott Duke Kominers. Universal reconfiguration of (hyper-)cubic robots. *ArXiv e-Prints*, 2011. URL: https://arxiv.org/abs/0802.3414v3.

2 Hugo A. Akitaya, Esther M. Arkin, Mirela Damian, Erik D. Demaine, Vida Dujmovic, Robin Y. Flatland, Matias Korman, Belén Palop, Irene Parada, André van Renssen, and Vera Sacristán. Universal reconfiguration of facet-connected modular robots by pivots: The O(1) musketeers. *Algorithmica*, 83(5):1316–1351, 2021. doi:10.1007/S00453-020-00784-6.

3 Hugo A. Akitaya, Erik D. Demaine, Matias Korman, Irina Kostitsyna, Irene Parada, Willem Sonke, Bettina Speckmann, Ryuhei Uehara, and Jules Wulms. Compacting squares: Input-sensitive in-place reconfiguration of sliding squares. In *Proc. 18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2022)*, volume 227 of *LIPIcs*, pages 4:1–4:19, 2022. doi:10.4230/LIPICS.SWAT.2022.4.

4 Matthew Connor and Othon Michail. Centralised connectivity-preserving transformations by rotation: 3 musketeers for all orthogonal convex shapes. In *Proc. 18th International Symposium on Algorithmics of Wireless Networks (ALGOSENSORS 2022)*, volume 13707 of *LNCS*, pages 60–76. Springer, 2022. doi:10.1007/978-3-031-22050-0_5.

5 Adrian Dumitrescu and János Pach. Pushing squares around. *Graphs and Combinatorics*, 22:37–50, 2006. doi:10.1007/s00373-005-0640-1.

6 Othon Michail, George Skretas, and Paul G. Spirakis. On the transformation capability of feasible mechanisms for programmable matter. In *Proc. 44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *LIPIcs*, pages 136:1–136:15, 2017. doi:10.4230/LIPICS.ICALP.2017.136.

7 Frederick Stock, Hugo Akitaya, Matias Korman, Scott Kominers, and Zachary Abel. A universal in-place reconfiguration algorithm for sliding cube-shaped robots in quadratic time. In *Proc. 40th International Symposium on Computational Geometry (SoCG 2024)*, 2024. To appear.