Justin Werfel
Radhika Nagpal
Department of Electrical Engineering and Computer Science,
Harvard University, Cambridge, MA 02138, USA
{jkwerfel, rad}@eecs.harvard.edu

Three-Dimensional Construction with Mobile Robots and Modular Blocks

Abstract

We present a decentralized algorithmic approach to automatically building user-specified three-dimensional structures from modular units. Our bipartite system comprises passive units (blocks), responsible for embodying the structure and determining where further units can legally be attached, and active units (robots), responsible for transporting passive units. The algorithmic issues are correspondingly decomposed into two parts: (1) deciding where passive units may be attached; and (2) getting them to those locations. For the first part, we give simple, scalable rules for attachment and prove that they will reliably lead to the construction of any desired structure from a large class of three-dimensional shapes. For the second part, we compare three approaches: random movement, systematic search and gradient-following; each approach is successively faster but requires more communication overhead and/or unit capabilities. The system we describe enables guaranteed construction of desired structures using very simple agent algorithms, taking a high-level specification as the only required input. The topic of collective construction is related to the problems of programmed self-assembly and self-reconfiguration in modular robots, and the rules governing block attachment presented here may be usefully applied to such systems.

KEY WORDS—Robotics in Construction, Cellular and Modular Robots, Distributed Robot Systems, Autonomous Agents, Path Planning for Multiple Mobile Robot Systems

1. Introduction

Self-reconfigurable modular robotics traditionally considers systems with uniformly self-mobile modules. This universal mobility is important in a system intended to reconfigure repeatedly or frequently, as with reconfiguration for locomotion over different terrains (Kamimura et al. 2004) or playback of dynamic three-dimensional systems (De Rosa et al. 2006). However, in a system intended for the construction of static structures (civic structures such as buildings or bridges, or smaller artifacts such as tables or toothbrushes), it may be inappropriate. Once a structure is complete and modules need not move again, the capacity for movement can be a liability: not only is it unnecessary from that point on (and the hardware and associated complexity and expense wasted thereafter), but self-mobile units are likely to be less effective as structural elements than specialized passive units.

Here we present an approach to automatically building user-specified structures in three dimensions (Figure 1 Extension 1), using a bipartite system comprising passive units (blocks), which form the structure, and active units (robots), which manipulate the passive units. "Passive" units are not self-mobile but can communicate locally with physically attached neighbors. The separation into two classes of units allows passive units to be optimized for structural capabilities (such as load-bearing strength, insulating properties, etc.) and active units to be reused elsewhere. Such systems have been considered previously, focusing on issues such as hardware design (Terada and Murata 2004), low-level control (Everist et al. 2004) or communication (Jones and Matarić 2004). Our focus is on high-level algorithm design.

We present distributed algorithms by which a system of robots and blocks can be given a blueprint or other high-level representation of a desired final structure as input, and the agents take appropriate action to produce that structure without requiring further user intervention. Previously we have described and characterized such algorithms for two-dimensional systems (Werfel et al. 2005; Werfel and Nagpal 2006; Werfel et al. 2006); here we extend the approach to three dimensions.

As with the modules, so too can the algorithmic issues can also be divided into two parts. The first part is to determine where blocks may be added to a partial structure. That is, given

The International Journal of Robotics Research
Vol. 27, No. 3–4, March/April 2008, pp. 463–479
DOI: 10.1177/0278364907084984
©SAGE Publications 2008 Los Angeles, London, New Delhi and Singapore
Figures 1, 11, 12 appear in color online: http://ijr.sagepub.com

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

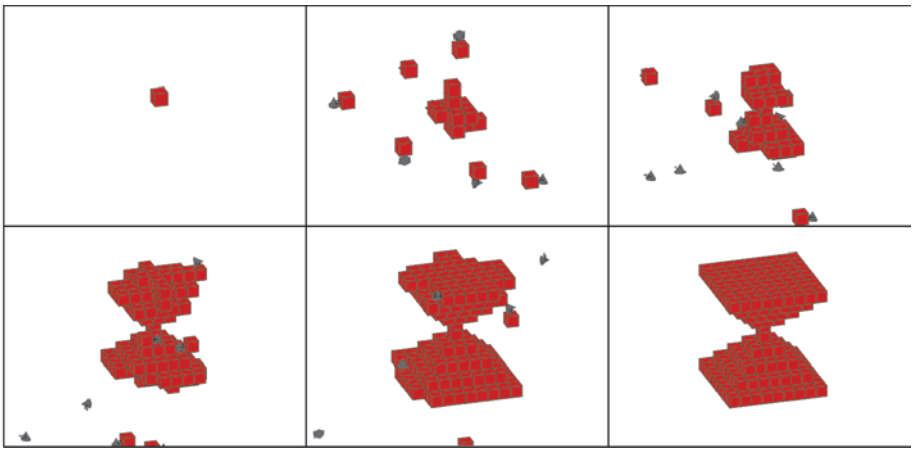


Fig. 1. Successive snapshots during the construction of an hourglass-shaped structure.

a prospective attachment site, is attaching a block there guaranteed to be a step on a path towards obtaining the desired final structure, without the possibility of getting stuck in a state of partial completion? The second part is to find those allowed attachment sites. That is, starting from some arbitrary location, how can a block wind up attached at an allowed site? In our approach, the blocks are responsible for the first issue and the robots are responsible for the second.

The block algorithms enable robot algorithms to be very simple while still guaranteeing completion of the desired structure. The amount of communication required between physically attached blocks scales linearly with the size of the structure, while explicit communication between robots is unnecessary. All algorithms are robust to variations in the number of agents and the order and timing of their actions, and avoid unwanted structural flaws and deadlock conditions.

In Section 2 we introduce the framework in which we consider the problem and describe our assumptions about unit capabilities and physical constraints on block movement. Section 3 gives a high-level picture of the block algorithm regarding placement and the robot algorithms regarding movement. Section 4 analyzes the block algorithm in more detail, giving a full correctness proof and experimentally evaluating the scaling of communication costs with structure size. Section 5 compares the robot algorithms in terms of their movement and communication costs. Section 6 reviews related work, and compares this framework of collective construction to self-reconfigurable modular robotics and to programmed self-assembly. Section 7 concludes.

2. General Problem and Assumptions

Our goal in designing a system for automated collective construction is to be able to deploy an unspecified number of robots into an obstacle-free workspace, along with a supply of free blocks and a single-block "seed" for the structure, and have construction proceed without further intervention. The only input we want to have to give the system is a high-level description of the desired structure (the "shape map"), and the outcome we desire is to have it reliably build that structure. Towards this end, we describe a set of rules for robots and blocks to follow that, in conjunction with the shape map, will do something appropriate to achieve the target structure without getting stuck along the way.

We assume a weightless environment. Dealing with gravity would introduce a variety of additional considerations (static stability of the partial structure, ability of the robots to climb the structure to reach target sites, etc.) that can be left for future work. Further, many strong potential applications for an automated construction system are in weightless environments, such as in outer space or underwater.

Robots are assumed to be able to move freely in any direction in three dimensions, alone or while carrying a single block, and to avoid collisions. They are able to fetch free blocks from elsewhere in the workspace and bring them to the structure in progress (which could be done as simply as with a random walk (Jones and Matarić 2004) or in a more directed way as with the use of beacons (Werfel et al. 2006)). Once at the partial structure, they can move along its surface in any direction and attach blocks at sites where permitted. Robots are not assumed to be able to communicate with other robots; in this way the system sidesteps difficulties associated with communication in *ad-hoc* mobile networks. There is no sort of centralized control; coordination is handled implicitly, via the structure being built.

Blocks are cubic[1], able to communicate with physically attached neighbors and to agree on a shared coordinate system,

1. We use cubic blocks for convenience in a Cartesian coordinate system and because human-made structures are almost always built to a rectilinear plan.

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

and to indicate to passing robots whether attachment at an exposed face is allowed. For some robot movement algorithms, blocks are required to be able to communicate additional information to passing robots, as described in Section 3.2. Blocks must also have enough static memory to store the shape map and several bytes of dynamic memory. The shape map may be as simple as an occupancy grid, or some more compressed format such as a collection of overlapping cuboids whose superposition defines the desired structure (Støy and Nagpal 2004).

As the precise alignment of blocks is essential in construction applications and because manipulating physical objects can be a difficult task even for sophisticated mobile robots, we envision that implementations of the sort of system we consider will involve blocks being equipped with some form of self-aligning connectors (Balaguer et al. 2002; White et al. 2005). Robots then need only maneuver blocks to near the right position and the connectors will ensure precise alignment. This self-correction will permit construction without problematic accumulated error for large structures without loops as we consider here. The grid embodied by the assembled blocks acts as a reference allowing robots to keep their position estimates accurate to within a scale smaller than the block size, and to correct small position errors as they move. These considerations make the uncertainty inherent in physically instantiated systems manageable, as we have demonstrated with a prototype of a two-dimensional system (Werfel et al. 2006).

Once a block is attached at a site, it is never detached thereafter; this approach is appropriate for construction applications where building materials are to be fastened together permanently for strength and durability (as with bricks and mortar). Under this assumption, some restrictions on when and where blocks may be attached become necessary, to avoid situations where a desired structure cannot be completed as intended.

The only assumption we make about physical constraints on block movement is that a space one block wide directly between two other blocks is too narrow to require that a robot be able to maneuver a block into it (Figure 2(A)). As an important corollary, this restriction prevents more complicated situations where intended attachment sites become inaccessible, or where robots might have to travel down difficult "tunnels".

3. Algorithmic Overview

The problem of building an arbitrary user-specified structure can be considered in two parts: (1) determining whether a block may legally be attached at a given site; and (2) ensuring that robots find all sites where attachment is permitted. In this section we introduce our approach to these two parts (Sections 3.1 and 3.2, respectively); Sections 4 and 5, respectively, elaborate more fully.

3.1. Block Rules

The blocks of the structure are responsible for determining where additional blocks may be attached. Blocks each have a copy of the shape map and store their position in a shared coordinate system. The structure begins with a single "seed" block, which has in its memory the initially single copy of the shape map and coordinates $(0, 0, 0)$. As each block is added to the structure, it receives from its neighbors a copy of the shape map, its own coordinates and information about where relevant blocks have already been attached. Blocks then determine at which of their faces attachment should be allowed and grant or deny permission to robots accordingly.

Knowing the shape map and the current position in it makes it trivial to determine whether a given site should eventually be occupied. However, it is also necessary to impose a partial ordering on block attachment to prevent dead-end states, which can occur because of the physical constraints on block movement. For example, if an intended attachment site becomes completely surrounded by blocks, no robot can reach it to attach a block there, and the structure can never be completed as specified.

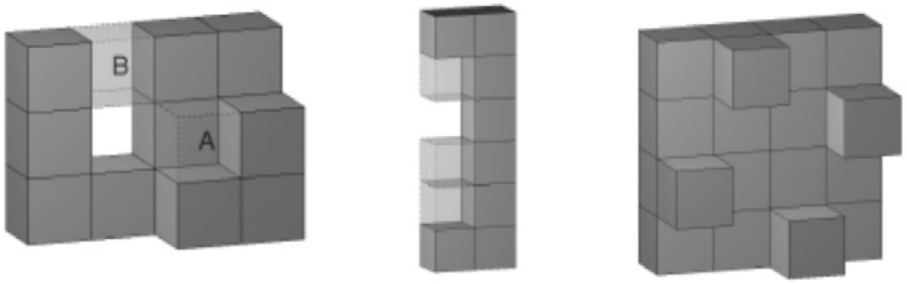
Necessary restrictions follow from the assumption about spaces one cell wide being unfillable. To start with, two blocks must never be attached in the same row (along any of the three axes) if all of the spaces between them are intended to be occupied (the *row rule*). Otherwise, an unfillable gap will eventually result (Figure 2(B)).

In two dimensions, this restriction is sufficient to achieve any desired structure without holes: attaching blocks freely at any sites the shape map specifies should be occupied, as long as separated blocks are not inappropriately attached in the same row, will provably result in the desired structure without the possibility of dead-end states (Werfel 2006). In three dimensions, that single restriction is not sufficient. As Figure 2(C) shows, it is possible to reach a state, always obeying the row rule, where the addition of any further blocks is impossible without breaking that rule.

As a result, we add the following second restriction, which can be seen as a higher-dimensional extension of the first. Consider any two-dimensional slice through the shape map, perpendicular to one of the Cartesian axes. In that plane, sites intended to be occupied form one or more contiguous groups. For any such group, blocks may not be added if separated from previously attached blocks in that group (the *plane rule*). To restate both restrictions, the row rule says that a contiguous group of blocks in the same row can start with its first block attached anywhere, but successive blocks must be attached contiguously from there; the plane rule says that a contiguous group of blocks in the same plane can originate anywhere but thereafter must grow from that point of origin.

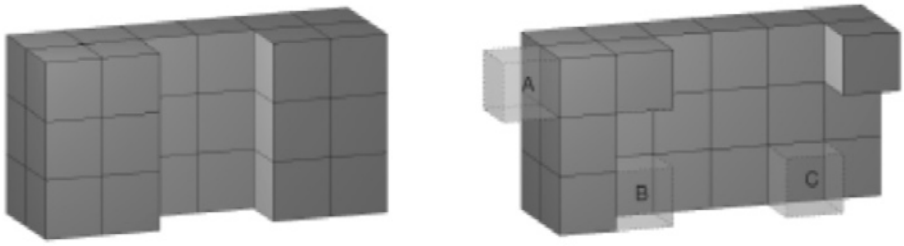
In summary, attachment is permitted at a site if and only if all of the following hold (Figure 3): (1) the shape map specifies that the site is meant to be occupied; (2) attachment at the

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016



(A) (B) (C)

Fig. 2. (A) Physical restriction on block placement. A block could be attached, e.g., at site A. Site B, although having fewer neighbors, cannot accommodate a block because neighboring blocks occupy sites at opposite faces. (B) If two separated blocks are attached in the same row, then as blocks are added between them (shaded), an unfillable gap will ultimately result. (C) This structure can result without violating the row rule; however, no further block can be attached in the front plane without violating that restriction.



(A) (B)

Fig. 3. (A) A desired structure and (B) a possible intermediate stage in its construction. Sites A, B, and C are three sites at which attachment is forbidden. Site A should be left empty according to the shape map. Attachment at site B satisfies the plane rule but violates the row rule: there is a block above it and the sole intermediate site is meant to be occupied. Attachment at site C satisfies the row rule (because the shape map specifies that the center region is to be left empty) but violates the plane rule: it would not be attached contiguously with the block already attached in the locally contiguous group in one plane.

site obeys the row rule; and (3) attachment obeys the plane rule.

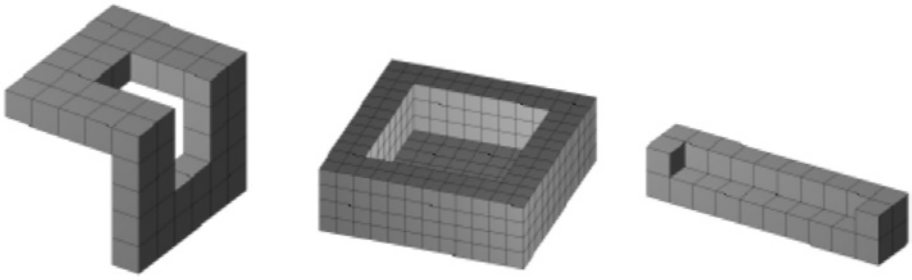
3.2. Robot Rules

Robots are responsible for locating allowed attachment sites along the surface of the partial structure and transporting free blocks there. Having blocks be responsible for maintaining the block rules just described makes it possible for robots to use very simple algorithms to transport blocks to their destinations. We consider three different approaches by which robots find allowed sites once they have brought a free block to the structure.

1. *Random movement*. Robots repeatedly move one space on the grid formed by the blocks, in any available direction along the structure surface, with no memory of where they have come from or what sites they have visited already. This approach has the advantage of simplicity and low communications overhead: the only information blocks need to communicate to robots is whether attachment at a given site is allowed, which could be implemented as simply as by turning on a light. However, the approach can require a great deal of unnecessary robot movement, especially for structures with large surface area, as shown in Section 5.

2. *Systematic search*. In two dimensions, if robots circle the structure perimeter in a given direction, they will find all available attachment sites. The most straightforward extension to three dimensions is to treat the structure as a stack of layers and each layer as a two-dimensional structure to be built. Robots circle the structure at a fixed height; if they return to a previously vis-

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016



(A) (B) (C)

Fig. 4. (A) A structure with a loop. (B) A structure which is topologically equivalent to a sphere but contains loops in horizontal planar slices. Note that the structure in (A) has a loop globally but not in any planar slice. (C) A "couch". Empty sites are meant to be left empty.

ited site, indicating that no attachment was possible at that level, they move up or down one level and continue with the next layer. This systematic search eliminates some unnecessary movement and guarantees that all sites are visited in finite time. However, it is more complicated in terms of required robot capabilities and/or block–robot communication and may restrict the class of possible structures further, as discussed in Section 5.

3. *Gradient-following*. This approach has robots receive explicit directions from the blocks of the structure. Those blocks with faces where another could be attached send out a directed numerical gradient (confined to the surface) to their neighbors (Støy and Nagpal 2004). Robots follow the gradient to the nearest available attachment site. Confining the gradient to the surface prevents robots from becoming caught in local minima. This approach can easily involve an order of magnitude (or better) less robot movement along the surface of the structure. The cost is an order of magnitude (or worse) more communication within the structure.

4. Block Algorithm

In this section we prove the correctness of the attachment rules of Section 3.1 for building any structure from a specified class (Section 4.1). We next consider communication requirements for a decentralized implementation (Section 4.2), showing that they scale linearly with the size of the structure.

4.1. Analysis

Attaching freely at any sites designated by the shape map, subject to the restrictions of the row and plane rules, will provably result in the reliable construction of any structure meeting the following criteria.

1. It contains no loops or holes, neither topologically nor in any plane (Figure 4(A) and (B)).

2. It contains no more than two "couches", where a couch is defined as a row of sites meant to be left empty, capped at each end by a site that is meant to be occupied, the whole bordered along two adjacent sides by sites that are meant to be occupied (Figure 4C). We discuss the relaxation of this criterion at the end of this section.

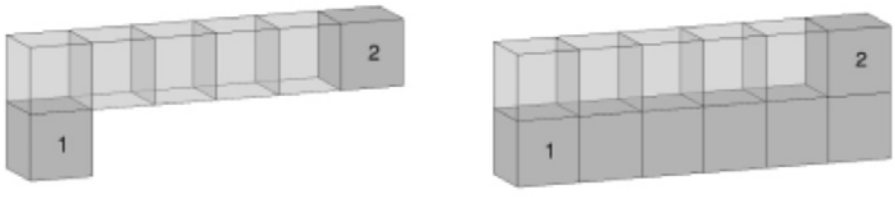
3. Any places where two parts of the surface approach each other (as in the legs of a U-shaped structure) must be far enough apart not to interfere with the movement of robots over the surface. The extent to which this limitation restricts allowable structures will depend on the hardware implementation.

To prove the correctness of these rules, we start with the following lemmas about partially completed structures built consistently with the rules and restrictions stated.

Lemma 1. *In a configuration like that of Figure 5(A), there must exist a row of already-attached blocks like that shown in Figure 5(B).*

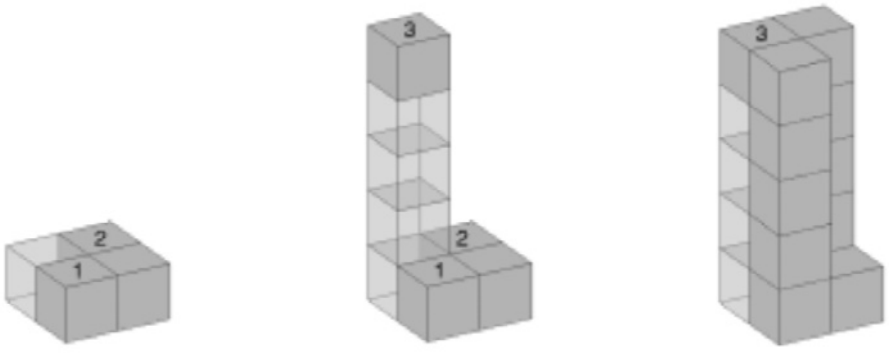
Proof. The row of intended blocks (i.e. sites where blocks are meant to be attached but are not yet present) indicates that blocks 1 and 2 are intended to be part of a connected group in the plane of the page. Thus, to avoid having violated the plane rule, they must already be connected in that plane. The chain of existing blocks connecting them cannot run above the row of intended blocks, or the structure would violate either the row rule or the rule against loops. Likewise, unless the chain of existing blocks runs immediately below the row of intended blocks, the structure will violate one of those two rules. □

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016



(A) (B)

Fig. 5. Illustration of Lemma 1. In any configuration such as that of (A), there must exist a row of already – attached blocks such as that shown in (B). Dark sites indicate blocks already present; shaded sites indicate those that are currently empty but are meant to be occupied; empty sites are currently empty and may or may not be intended to be occupied.



(A) (B) (C)

Fig. 6. Illustration of Lemma 2. If anywhere in a structure there is a configuration such as that of (A), there is a site somewhere where a block may legally be attached. If attachment is not permitted at the shaded site in (A), then the configuration must be as shown in (B) and (C). Shading is as in Figure 5.

Lemma 2. *If anywhere in a structure there is a configuration such as that of Figure 6(A), there is a site somewhere where a block may be attached legally.*

Proof. If a block can be attached at the shaded site, the proof is trivial. If not, the plane rule cannot be the problem, because the shaded site is adjacent to existing blocks in all three associated planes. Thus, the problem must be a block in the same row as the shaded site in the direction orthogonal to the existing blocks shown, with all intermediate blocks meant to be occupied (Figure 6(B)). Now consider the plane containing blocks 1 and 3. By Lemma 1, there must exist a column of blocks from block 1 to the site adjoining block 3 (Figure 6(C)). The same argument applies to the plane containing blocks 2 and 3. Finally, consider the site just below block 3. This site is meant to be occupied and if the partial structure does not already violate either the row or plane rule, adding a block at this site cannot lead to the violation of either of those rules. Thus, attachment at this site is allowed. □

We are now ready to prove the following result.

Theorem 1. *Adding blocks in any order anywhere, subject to the attachment rules given in Section 3.1, will reliably produce any desired structure from the class described at the beginning of this section.*

Proof. We prove this by contradiction. Suppose that an incomplete stage of construction can be reached where no further attachment is possible. That is, for every site at which the shape map specifies a block should be attached, some rule forbids attachment, either because of a separated block in the same row or one in the same plane. We show that there must in fact be attachment possible somewhere, considering all possible cases in turn.

- *Case A*. If the problem is separation in a row, then there is a configuration such as that of Figure 5(A). By Lemma 1, the structure must then have a configuration

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

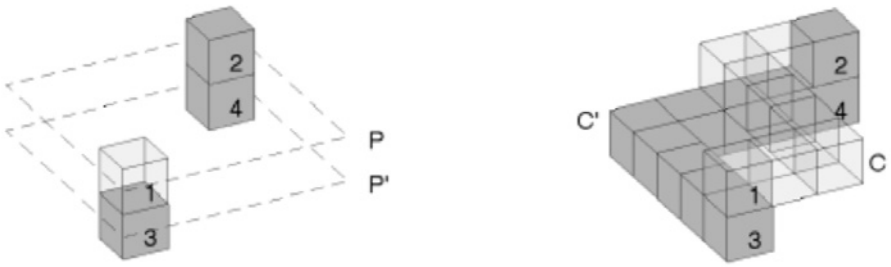


Fig. 7. A block cannot be attached at site 1 because block 2 in the same plane P means the plane rule would be broken. Shading is as in Figure 5. (A) Site 1 and block 2 are adjacent to blocks 3 and 4, respectively, in the plane P' adjoining P. (B) A chain of intended blocks C in P connects site 1 with block 2, and a chain of existing blocks C' in P' connects blocks 3 and 4.

such as that of Figure 6(A). By Lemma 2, there then exists a site where a block can be attached.

- *Case B*. Suppose instead that the problem is separation in a plane (Figure 7(A); labels in the following refer to this figure). That is, in a plane P there is a site 1 where a block is meant to be attached but cannot be because of a block 2 in the same plane, with some chain of intended blocks connecting the two in the plane. This case is more complicated than the first and requires several subcases to be considered.

  Site 1 must be adjacent to an existing block 3 in the adjoining plane P' (if block 3 were in P, the structure would already violate the plane rule). As the structure is assembled starting from a single initiation point, blocks 2 and 3 must be connected by some chain of existing blocks which at least passes through the plane P' (and may pass into planes beyond). Label as 4 the block in P' adjacent to[2] block 2. If the structure is not to have loops, there must be some manifold of intended blocks that connects the chain from 1 to 2 with that from 2 to 3. Where this manifold passes through P', its projection constitutes a chain of intended blocks in P' connecting blocks 3 and 4. As blocks 3 and 4 are meant to be connected in the plane P', they must already be connected in that plane.

  To summarize so far: there is an intended chain C between site 1 and block 2 in plane P, and an existing chain C' between blocks 3 and 4 in the adjacent plane P' (Figure 7(B)).

  To avoid loops, chain C must lie directly on top of some intended chain C'' in P'. There are two possibilities, which we next consider as separate subcases: (B1) C'' has already been built; or (B2) C'' has yet to be constructed.

  In case B1 (Figure 8(A)), the block in C'' adjacent to block 4, together with blocks 2 and 4 and the intended attachment site in C adjacent to block 2, form a configuration such as that of Lemma 2, and so a block may be attached somewhere.

  In case B2, there is more construction to be done in P' to build the chain C'' on top of which the chain C in P will be built. To avoid loops, some of this construction must be carried out directly next to the existing chain C'. There are again two possibilities, which we consider in turn as sub-subcases. The first case (B2i) is that there are bends in C' with construction intended on the inside of the bend (Figure 8(B)). In this case there will be attachment possible somewhere by Lemma 2.

  The final possibility (B2ii) is that there are no bends in C', or construction is intended only on the outside of any bends (Figure 8(C)). In such a case, the intended structure will contain at least one "couch" configuration.

  We have now shown that for all cases except (B2ii), attachment somewhere is possible. For case (B2ii), we can consider any one of those sites in P' adjacent to C' where attachment is desired and not allowed, and apply the entirety of the preceding argument again, now taking that new site to be site 1. The result will be that attachment is found to be possible somewhere, unless the situation is again that of this final case, where attachment is not allowed because of separation in a plane (B), with an intended chain in the adjoining plane which is not yet built (2) and an existing chain in the adjoining plane which has no bends where attachment is intended on the inside (ii), with a second set of sites, blocks, chains and

2. If block 2 has no adjacent block in P', then a different block in P that does have a neighbor in P' can be found, and the blocks relabeled accordingly.

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

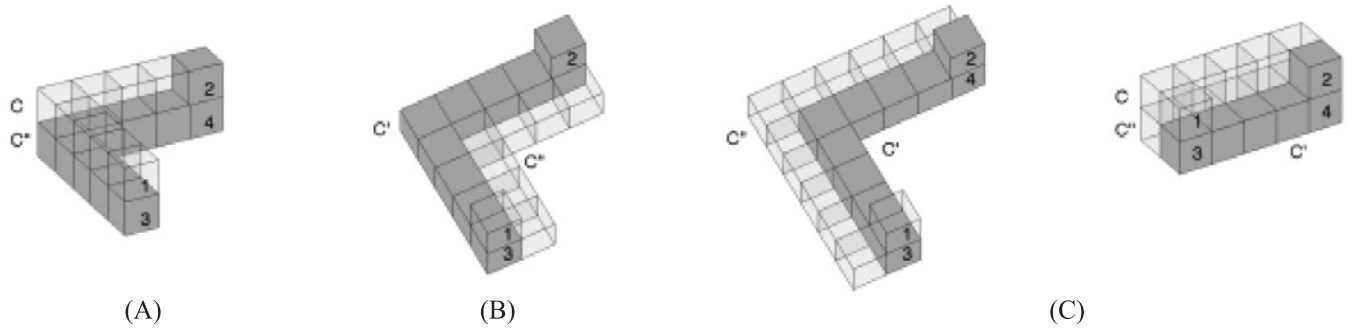


Fig. 8. Three possibilities for chains C (running between site 1 and block 2 in plane P), C' (running between blocks 3 and 4 in plane P') and C'' (matching C, but in plane P'). Shading is as in Figure 5. (A) Intended chain C lies on top of existing chain C'' (identical to C' in this example). (B) Intended chain C'' involves building on the inside of existing, bent chain C'. (C) Intended chain C'' involves building on the outside of existing, bent chain C' (left) or alongside a straight C' (right). In the latter case, the "couch" entailed by this configuration is clear to see. The former case entails two couches.

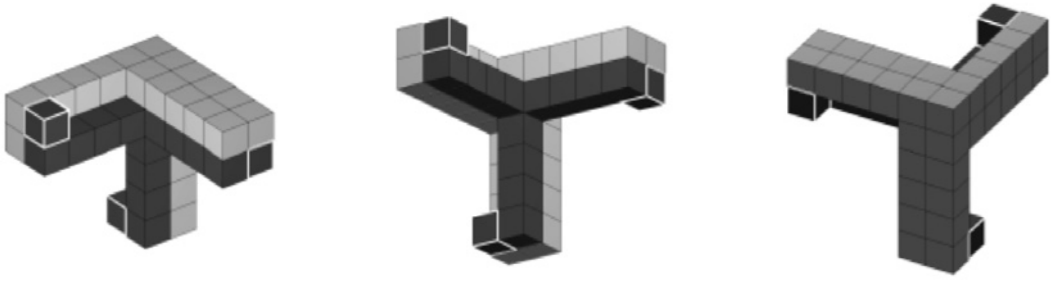


Fig. 9. Multiple views of a structure with three "couches", built consistently with the row and plane rules, in an incomplete state where no further blocks can be added legally. Dark shading indicates blocks that have already been attached; light shading indicates empty sites that the shape map specifies should be occupied; empty sites are intended to be left empty. Each remaining attachment site is forbidden as a result of being in the same plane as, but separated from, one of the three blocks marked with white edges.

planes. That configuration will entail at least a second couch. Now the entire argument can be applied a third time to yet another of the sites where attachment is intended but not allowed. As a third couch is not permitted in the desired structure, the argument terminates and we must have one of the other cases where a block can be attached somewhere. This completes the proof. □

The second restriction on the class of permissible structures, limiting them to have no more than two couches, is necessary for the sake of a strict guarantee on correct completion. A structure with three couches can be specified for which robots obeying the row and plane rules can conceivably get stuck in a dead-end state of partial completion (Figure 9). However, it is very difficult to wind up in such a configuration in practice. Even when building the example structure of that figure, experiments repeated 1,000 times for each of the three robot movement approaches considered in this paper resulted in the structure being completed successfully every time, never reaching a deadend. This result suggests that the algorithms presented here, while not strictly guaranteed to complete structures with more than two couches, can successfully be used for such structures with high probability.

4.2. Block Communication

Blocks can obtain the necessary information to enforce the attachment rules with local communication. When a new block is attached to the structure, it receives information from its neighbors: the shape map, its own location in the shared coordinate system and whether blocks have been attached in neighboring rows and planes. In addition, when it is attached, its neighbors disseminate information to other blocks in neighboring rows and planes about that new site being filled. In

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

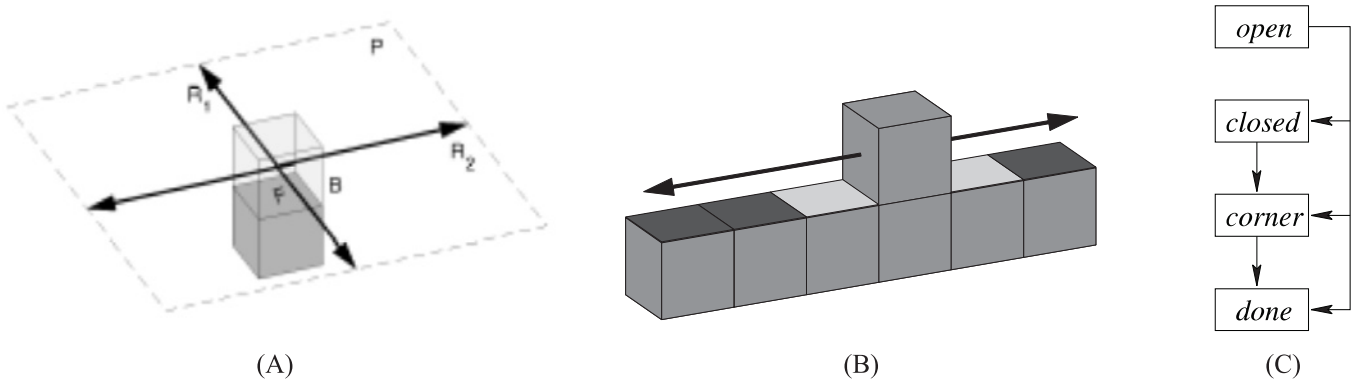


Fig. 10. (A) Each block face F has three state variables, which together determine whether a block B may be attached to it. Each variable is associated with one of three regions: the adjoining plane P or one of the two rows R_1, R_2. Before other blocks have been attached in such a region, associated face states are *open*. (B) When a block B is attached in such a region, faces associated with neighboring sites enter a *corner* state (light shading) and more distant faces are *closed* (dark shading). (For clarity, only variables associated with the indicated row are shown.) (C) The state machine for variables associated with such regions.

this way, structure blocks can maintain up-to-date information about the state of the relevant subset of the structure in progress, and designate a neighboring unoccupied site fit or unfit for attachment. They can then grant or withhold permission from robots seeking to attach blocks at that site. Communication costs are kept minimal by disseminating information only as far as it is needed.

4.2.1. Implementation of the Algorithm

The above approach can be implemented with a straightforward extension of the two-dimensional algorithm for block state and communication described by Werfel et al. (2006). The algorithm can be outlined as follows (Figure 10). Each face F of each block has state that reflects whether a new block B could legally be attached to F. Such a block B would be in a plane P parallel to F and in two orthogonal rows R_1, R_2 within P. The row and plane rules specify that B cannot be attached if there are non-contiguous blocks already in any one of $\{P, R_1, R_2\}$. Thus, F maintains three state variables, associated with those three regions, reflecting whether and where blocks in those regions have already been attached.

Before any blocks have been attached in such a region, the corresponding state variables of associated faces of already-present blocks will be in an *open* state, indicating that attachment is permitted anywhere in the region. When the first block in a region is attached, block faces associated with sites in that region that are adjacent to the newly added block enter a *corner* state (so called because those sites are at the corner of two blocks). Other block faces in the region enter a *closed* state. Faces bordering sites that the shape map specifies should be left empty are always *closed*. A face to which a block is actually attached becomes *done*.

Robots may attach blocks to faces for which all three state variables are in *open* or *corner* states. To prevent communication delays within the structure from permitting multiple robots to attach conflicting blocks near-simultaneously, an *open* plane or row should become (reversibly) locked after a robot requests permission to attach and before permission is granted. When a new block is attached to the structure, it sets its state variables based on those of its neighbors, and its neighbors change their own state as appropriate based on the attachment. Details are omitted here (see Chapter 3 of Werfel (2006) for details).

4.2.2. Communication Cost

Passing messages over distances is needed only when the first block is being added to a new row, or to a new contiguous group in a plane. For the addition of further blocks in that row or plane, communication beyond immediate neighbors is not required: blocks can grant or deny permission to attach without communicating, and messages following attachment need only travel to and from immediate neighbors of blocks involved in the attachment. Moreover, the way structures grow in practice makes it unusual for very much of a large plane or row to be constructed before the adjoining plane or row is started. As a result, messages needing to travel further than a distance of one block are rare, and the total amount of communication needed grows no worse than linearly with the size of the structure.

This limit on communication costs is demonstrated by simulation experiments involving the construction of cubic structures of increasing size, as follows. A structure is initialized with a single block. Additional blocks are added at random, consistent with the attachment rules, until the structure is

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

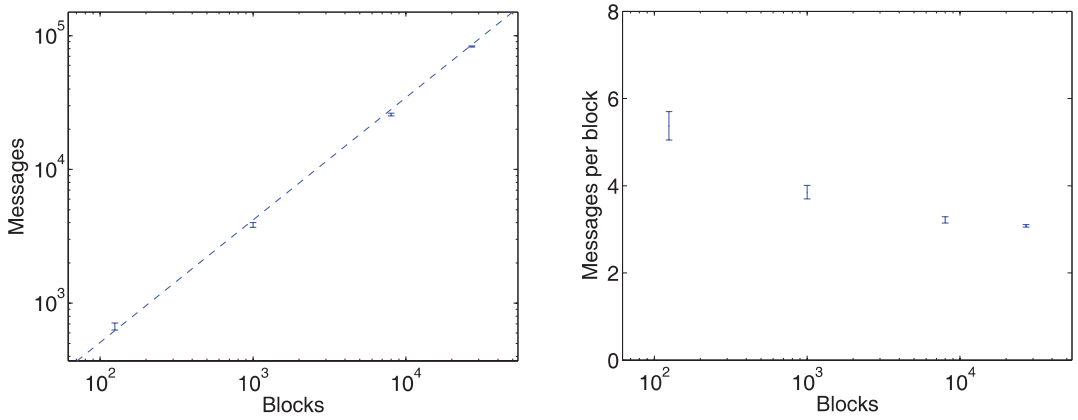


Fig. 11. (A) Total number of messages passed between blocks to check allowed attachment during construction of cubic structures of increasing size. The slope of the best-fit line is 0.92 ± 0.02. (B) Number of messages normalized by the number of blocks in the structure. The number of messages per block is roughly constant independent of structure size. The mild decrease may reflect finite-size costs less significant for larger structures. Averages are over 10 independent runs.

completed.[3] Checks for allowed attachment are implemented in a centralized way, but calculate at each step the number of messages required for a decentralized implementation. Figure 11 shows that the number of messages passed per block stays roughly constant over structures with sizes spanning over two orders of magnitude.

5. Robot Algorithms

In a two-dimensional construction, a robot can simply follow the perimeter of the partial structure and be guaranteed to visit every site along the perimeter exactly once in a full circuit. No such stateless procedure exists by which a robot starting from a random location can visit every site on the surface of a general three-dimensional structure exactly once (Werfel 2007). As a result, robots in this system need movement algorithms that involve additional state, communication and/or revisiting previously rejected sites.

Different algorithms that robots may follow to find allowed attachment sites will have different costs. In Section 3.2, we introduced three such robot algorithms based on random walk, systematic search and gradient-following. In Section 5.1 we elaborate on them in more detail, discuss their correctness and compare them qualitatively with respect to several criteria, including required agent capabilities, robot travel distance/time and communications costs. Section 5.2 gives a quantitative comparison. Section 5.3 discusses issues related to the simultaneous use of many robots.

3. More specifically, these sites are found by robots using the random walk algorithm, as described in more detail in Section 5. Communication costs associated with maintaining block attachment restrictions are not significantly different when robots use the systematic search or gradient-following approaches.

Algorithm 1 "Random walk" algorithm for robot movement.

```
while structure not finished do
  bring block to structure
  while not allowed to attach at current site do
    move along structure surface to adjacent grid site in
    random direction
  end while
  attach block
end while
```

5.1. Algorithms and Discussion

The *random walk* approach (Algorithm 1) is the most straightforward and simplest of the three. Robots need not maintain any state, and the only information they need from blocks is whether attachment at their present location is allowed. The weakness of the approach is that a robot will generally exhibit a great deal of repeated movement over the structure surface, revisiting previously rejected sites, before finding an allowed site.

A random walk on a two-dimensional lattice is recurrent, i.e. it will return to the origin with probability 1 if allowed to continue for infinite time (a property not true of random walks in higher dimensions). As the structure surface is two dimensional, a random walk will reach every site on the surface if left long enough. Thus, the random walk approach is at least guaranteed to finish the structure eventually, although there is no guarantee on how long it may take to do it.

The *systematic search* approach (Algorithm 2) involves less revisiting of previously covered ground and implies a limit on

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

Algorithm 2 "Systematic search" algorithm for robot movement.

```
while structure not finished do
  bring block to structure
  initialize vertical-direction to up or down, at random
  empty visited list
  while not allowed to attach at current site do
    if current location in visited list then
      move to perimeter of next level in vertical-direction
      if at top or bottom of desired structure then
        reverse vertical-direction
      end if
    else
      add current location to visited list
      move along structure surface clockwise at current
      level
    end if
  end while
  attach block
end while
```

the time necessary to find an allowed attachment site. However, it is more complicated than random walk with respect to a number of issues, including capabilities and behavioral complexity required of robots, and potential restrictions on the class of buildable structures.

Moving up or down a level in the structure is triggered by returning to a previously visited site. Thus, robots need enough dynamic memory to keep track of which sites they have visited. Alternatively, blocks can be responsible for keeping track of this information (as with robots leaving "breadcrumbs" behind them), at the cost of increased memory in the blocks and increased communication between blocks and robots.

The procedure of moving up or down a level can involve moving not only vertically but also inward or outward to reach the perimeter of the layer at the new level. For a structure that looks like a stack of arbitrarily shaped flat plates, moving up a level (from the perimeter of one plate to that of the next) can be accomplished in three steps: (1) if there are blocks directly above, move outward until the way upward is clear; (2) move up one step; (3) if not yet at the perimeter of the plate at this level, move inward until that perimeter is reached. This procedure requires robots to be able to establish both vertical and radial directions. Such abilities could be implemented through block–robot communication, or with external beacons (one on the structure's vertical axis), the directions to which robots could use as an additional reference.

With a more complicated structure than a stack of flat plates, moving up or down a level can be non-trivial algorithmically. For instance, a robot trying to move up a level could be caught under a ledge with an overhanging lip, forcing it to move down before it is able to move out and up. More complicated search strategies or restrictions on the class of possible structures could avoid such problems; alternatively, if robots as well as blocks have copies of the shape map, robots could plan a path to the next level.

Unless such algorithmic complexities are allowed for the robots, the class of possible structures will be restricted further with the systematic search than with the other two methods considered. The stack-of-plates case outlined above is the simplest in terms of robot capabilities, search strategies and communication; robots only need to be able to establish the vertical and radial directions, and they have a simple, reliable procedure for moving from one level to another. However, permissible structures then have the following additional constraints on each layer. (1) The blocks of the layer must form a single connected unit without holes. This constraint reduces the problem of building the layer to the previously solved two-dimensional case. (2) There must be a block directly above or below the initial seed, which furthermore must be the first block attached in that layer. This constraint ensures that a robot moving radially inwards in any layer will always find a place to attach in that layer. These limitations could be circumvented, but at the cost of increasing the complexity of the robots.

The *gradient-following* approach (Algorithm 3) requires the most direct coordination between the robots and blocks, as well as a great deal of inter-block communication to maintain gradient information. That information includes the number of steps away from the closest allowed attachment site, as well as the local direction to travel to reach it. Including direction information means that robots only need to receive messages from immediately adjacent blocks, and not from all of the neighbors of those blocks, to determine which way to go.

5.2. Quantitative Comparison

We conducted a series of simulation experiments to evaluate quantitative differences in performance of these three approaches. These considered three measures:

1. D, the total distance traveled by all robots during the full course of construction;
2. M_1, the number of messages sent from blocks to robots;
3. M_2, the number of messages passed between physically connected blocks.

The distance D reflects the time robots spend searching for allowed attachment sites, potentially involving repeated or wasted effort. Since physical movement is likely to be very much slower than communication, D is also associated with the time it takes to complete the structure. Messages M_1 include an indication of whether attachment at a given site is allowed. Messages M_2 may be of lower cost than M_1, since they involve communication via a physical connection. We do

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

Algorithm 3 "Gradient-following" algorithm for robots and blocks.

```
A: Robots
while structure not finished do
  bring block to structure
  while not allowed to attach at current site do
    move along structure surface to adjacent grid site in
    direction specified by neighboring block
  end while
  attach block
end while
B: Blocks
loop
  if just now attached to structure then
    for all faces on surface of structure do
      value ← min(values of neighboring faces)+1
      direction ← toward minimum neighboring value
    end for
    if attachment is allowed at any faces then
      for all faces where attachment is allowed do
        value ← 0
        direction ← none
      end for
      send update-gradient messages to neighbors associ-
      ated with those faces
    end if
  end if
  if received update-gradient messages from neighbors
  then
    if associated faces would then have smaller value then
      for all faces that would have smaller value do
        value ← min(values of neighboring faces)+1
        direction ← toward minimum neighboring value
      end for
      send update-gradient messages to neighbors associ-
      ated with those faces
    end if
  end if
end loop
```

not explicitly consider the quantity of information associated with each message.

An additional measure of interest might be total construction time. This quantity depends on a number of factors besides D, including: L, the time it takes to retrieve a free block and bring it to the structure; A, the time it takes to attach a block once an appropriate site has been found; N, the number of robots; and the structure's size and shape (Werfel and Nagpal 2006). The time L will depend on issues such as the layout of the workspace and the way robots find free blocks and the structure; A will depend on the hardware implementation; neither depends on the robot search algorithm. For these reasons, we focus on D as the measure affecting construction time most relevant for comparison of robot algorithms. The number of robots N is discussed briefly in Section 5.3.

Simulations were performed as follows. The structure is initialized with a single block. Ten robots, each the size of a block, are placed at random a fixed distance away. Each moves inwards (one grid space per time step) until it reaches the growing structure, then moves along its surface according to one of the three approaches. When it finds a site where attachment is allowed, it adds a block at that location, then moves instantaneously back to a random location at the original distance from the seed block, and continues with another free block. Inter-block messages relating to allowed attachment travel infinitely fast compared to the speed of robot movement. Gradient messages travel one grid space per time step; updating gradient information rapidly compared with the speed of robot movement can increase the total distance robots end up traveling (Støy and Nagpal 2004). Updates are asynchronous.

Figure 12 shows the results for several example structures. Overall, there is a clear tradeoff among these three approaches between extraneous movement and communications costs.

The random walk approach requires more robot movement than the other approaches. This difference increases for structures with greater surface area. The systematic search eliminates a great deal of repeated movement and guarantees that robots will reach every allowed attachment site in bounded time. Robots using the gradient-following approach travel much smaller distances still (for structures of the scale and complexity tested here, an order of magnitude smaller), as they are directed straight to the closest available site. However, the cost is considerably more communication within the structure (in these experiments, one to two orders of magnitude more), to establish and maintain the gradient. In addition, the (fewer) messages sent from the structure to robots will need to contain more information, specifying movement direction in addition to whether attachment there is allowed. These costs are likely to be bearable in practice for the sake of the speedup and reduced robot movement in assembly, because communication between blocks that are physically connected should be rapid, unambiguous and reliable, and hence of low cost compared with less direct communication or (most importantly) actuation.

Another important consideration is how the costs of each of these approaches scale with structure size. Figure 13 and Table 1 show the results of experiments on cubic structures of increasing size.

Robots using the gradient-following approach traveled significantly shorter distances, with more favorable scaling, than did robots using the other two approaches. Surprisingly, the random walk appears to scale more favorably than the systematic search, within the range of structure sizes tested. This result, which deserves further study in future work, may reflect a tendency for robots using systematic search to revisit sites across multiple trips to the structure to attach a series of blocks.

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

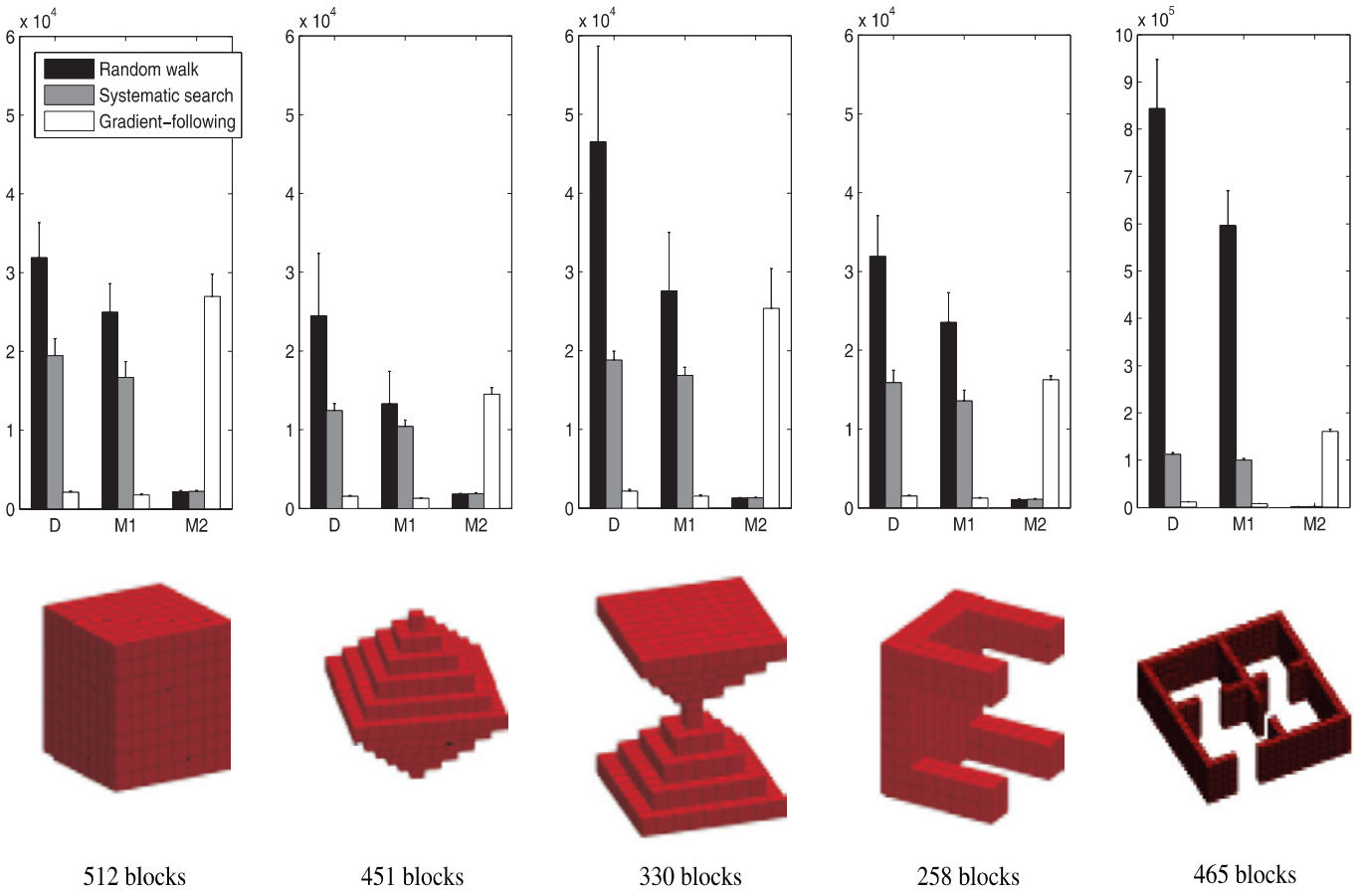


Fig. 12. Experimental results for the movement and communication required to build various structures. Here D is the distance traveled along the surface, in block-lengths, M_1 is the number of messages passed from blocks to robots and M_2 is the number of messages passed between blocks. Averages are over 10 independent runs.

The number of messages sent from blocks to robots has results very similar to robot travel distance, as expected because of the way robots receive messages from the structure as they travel over its surface.

The number of messages passed within the structure is significantly greater, and scales more poorly, for the gradient-following approach than for the other two. There is no significant difference between random walk and systematic search with respect to this measure: the only inter-block messages are associated with maintaining the row and plane rules, and as noted earlier, the amount of communication required for that purpose does not depend on the robot algorithm.

5.3. Multiple Robots and Interference

One of the primary motivations for the swarm approach is its potential for great parallelism. The approach described here enables such parallelism, both with the opportunity for multiple robots to "pipeline" tasks (fetching building material, searching the structure for allowed attachment sites) and with the multiplicity of sites where blocks may be attached at any given time. While the attachment rules require locking out parts of the structure from attachment to ensure that contiguous groups in each row and plane are assembled contiguously, there are typically many sites at a time where blocks could be simultaneously attached (Figure 14).

Naturally there are limits to the number of robots that can contribute at any one time to construction. With too many robots, not only may there be no tasks available for the extraneous robots to contribute to, but robots may start to interfere with each other's movement, slowing down overall progress as they have to spend time avoiding each other.

There is no unique number of robots ideal for any given building project. The number of robots that can usefully be employed at one time depends on factors such as the layout of the workspace, the time it takes to attach a block once a permitted site has been found and the surface area of the structure. As the last of these factors changes as construction

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

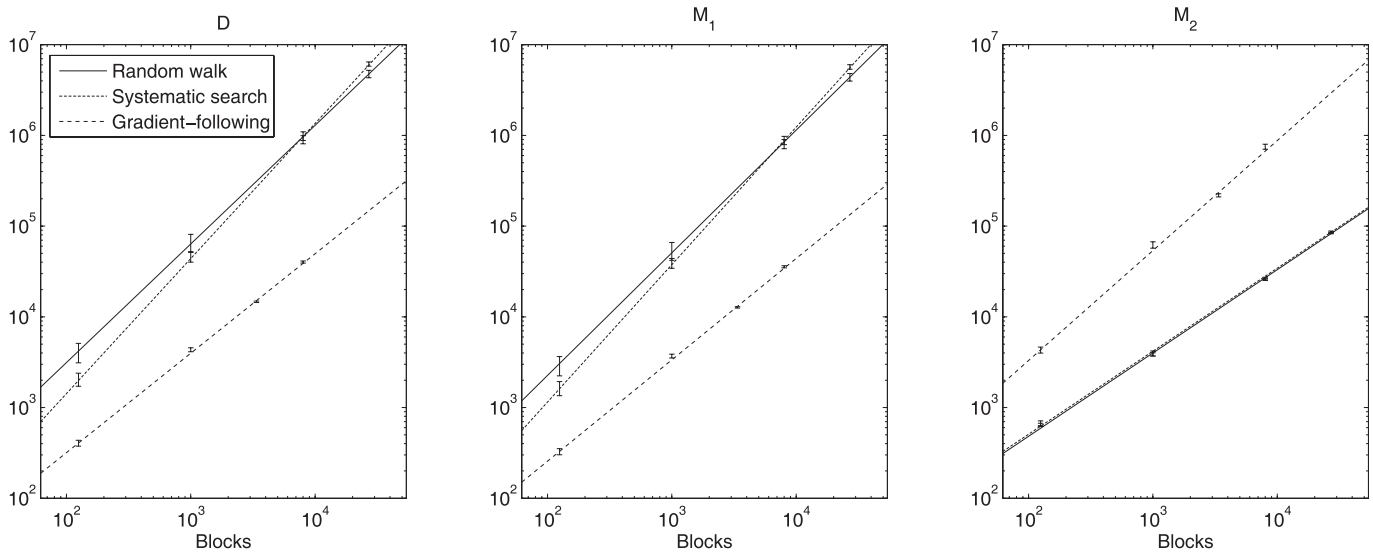


Fig. 13. Performance measures as a function of number of blocks, for cubic structures of different sizes. Here D is the total distance traveled by all robots during construction, M_1 is the number of messages passed from blocks to robots and M_2 is the number of messages passed between blocks. Averages are over 10 independent runs.

Table 1. Best-fit lines for performance experiments shown in Figure 13, building cubic structures of n blocks. Here D is measured in block-lengths, and M_1 and M_2 in the number of messages.

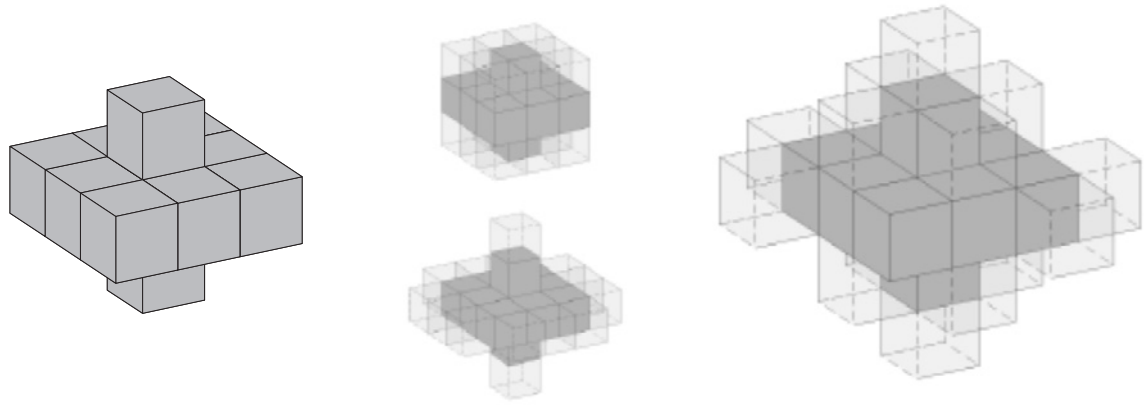
Robot algorithm	D	M_1	M_2
Random walk	$(7.6 \pm 0.5)n^{1.308\pm0.008}$	$(4.5 \pm 0.4)n^{1.352\pm0.009}$	$(7.0 \pm 1.3)n^{0.92\pm0.02}$
Systematic search	$(1.5 \pm 0.2)n^{1.49\pm0.02}$	$(1.07 \pm 0.15)n^{1.52\pm0.02}$	$(7.5 \pm 0.9)n^{0.916\pm0.013}$
Gradient-following	$(2.0 \pm 0.5)n^{1.10\pm0.03}$	$(1.5 \pm 0.4)n^{1.12\pm0.03}$	$(12 \pm 5)n^{1.21\pm0.05}$

proceeds, the number of robots the project can accommodate likewise changes. An effective distributed way to dynamically adjust the size of the swarm appropriately for the amount of work available is for each robot to note the fraction of time it spends avoiding collisions with other robots and to leave the active workspace temporarily when this fraction exceeds some threshold, rejoining the active builders at random later on (Werfel 2006). This approach lets the number of active robots increase as the structure grows, so that the density of robots searching its surface remains roughly constant. That density will depend on the threshold and also on factors such as robot size and footprint. While the choice of threshold affects the exact equilibrium density, it is not necessary to carefully tune its value in order for construction to be successful.

With some such scheme to limit robot density on the structure surface, so that traffic does not clog up completely, each of the three robot algorithms we consider lends itself easily to avoiding deadlock. Robots performing a random walk can limit their choice of step to directions unobstructed by other robots, and so clumps of robots will tend to spread out on average. All robots performing a systematic search circle the structure at any level in the same direction; the only case where two robots might meet head-on is when one is moving to a higher level and one to a lower, and some convention like passing-on-the-right will resolve such cases. Gradient-following robots building with identical blocks[4] may find two robots heading for the same site; any scheme to give one priority will resolve the conflict and let one become first in line. In weightless environments, robots can leave the structure surface after attaching a block, obviating the traffic problem of getting out of the way of material-bearing robots; if robot mobility is limited to the surface, an ability for robots to pass building material from one to another will let blocks reach attachment sites without requiring robots to be cleared out of the way first.

4. With multiple types of blocks (Werfel et al. 2005), the situation becomes more complicated. The simplest extension is for a separate gradient to be maintained for each block type. But then two different gradients may direct robots into each other, and some conflict resolution scheme is necessary.

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016



(A) (B) (C)

Fig. 14. Sites where attachment is simultaneously permitted. (A) An example 11-block structure. Assume that the shape map specifies that all sites are to be occupied. (B) Attachment is permitted at any of the 30 shaded sites (shown in two sets, for clarity). (C) As attaching a block in a row or plane locks out non-contiguous parts of that row or plane from further attachment, not every site could accommodate attachment simultaneously. Up to 14 blocks could be attached simultaneously to this structure (e.g. those shaded here).

6. Related Work

The problem of automating three-dimensional collective construction, as formulated in this paper, is related to a number of topics in self-reconfigurable modular robotics and beyond. In this section we discuss related work in collective construction, modular robotics, and programmed self-assembly, as well as the connections between these areas.

Several researchers have considered bipartite systems of mobile robots and manipulated material, often explicitly intended for construction. In most cases, either these studies do not have the goal of building a particular desired structure (Melhuish et al. 1999; Bowyer 2000) or they require specifying an explicit low-level sequence of building steps as an input to the system (Jones and Matarić 2004; Terada and Murata 2004).

A sampling of this work follows. Terada and Murata (2004) describe a robot that moves along and manipulates a substrate of passive cubic blocks, which are attached using a passive locking mechanism operated by the robot. They focus on the hardware design of the robot and blocks, and do not consider automated control of the robot. Everist et al. (2004) discuss a two-dimensional system intended for the assembly of structures in outer space, using self-mobile pucks to assemble passive struts. They demonstrate low-level strut-assembly operations. Jones and Matarić (2004) explore the use of inter-robot communication in a two-dimensional system with passive cubic blocks. Melhuish et al. (1999) consider minimalistic approaches for robots manipulating circular pucks in a plane. Bowyer (2000) proposes a scheme for a hardware system of terrestrial robots building with polymer foam. Théraulaz and Bonabeau (1995) consider a model with stateless agents moving on a cubic lattice, with rules directing them to deposit building material based on local observations of existing material. Their goal, to predict the characteristics of the resulting structures by studying the rules, is the opposite of ours, to find rules that generate a desired structure. Detweiler et al. (2006) discuss a bipartite system for self-assembling structures, primarily in two dimensions, using mobile units and passive struts. Structures in their approach are not intended to be built from struts alone; the mobile units are seen as an essential part of the final structure.

Other work in self-reconfigurable modular robotics considers systems with uniformly self-mobile units, often cubic in shape, which are required to rearrange themselves autonomously into a given desired configuration in a distributed way (Vassilvitskii et al. 2002; Fitch et al. 2003; Kotay and Rus 2004; Grushin and Reggia 2006; Støy 2006). In addition to the considerations discussed at the beginning of this paper about the utility of separating mobile and structural elements, these systems often involve movement assumptions inappropriate for construction problems, e.g. modules may be able to move on their own down narrow passages. Adapting our construction approach to those systems, however, may be straightforward for modules with appropriate communication abilities. In addition, several elements developed in many self-reconfigurable systems, such as self-aligning connectors between modules and inter-module communication mechanisms, would be invaluable to blocks in our approach to construction.

Another closely related area is that of programmed self-assembly: the problem of designing a set of elements to have edge-binding properties such that, when the elements mix randomly, they bind to form desired assemblies. If these elements

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

can communicate with attached neighbors and change their binding properties dynamically, then the self-assembly problem matches the problem that blocks are responsible for in our construction framework. In both cases, the key is to establish at which empty sites attachment is and is not allowed.

A few especially relevant studies are as follows. Klavins et al. (2006) discuss a formal approach to generating grammars that can lead to a desired structure in two dimensions. Jones and Matarić (2003) present a compiler to generate rules for communicating tiles, to form desired two-dimensional shapes from a certain class of possibilities. White et al. (2005) present hardware design for a self-assembling system of cubes in three dimensions; their structures are specified with a sequence of attachment steps provided as an input to the system.

Often in programmed self-assembly, physical constraints on element motion are not considered explicitly. As a result, the assemblies are subject to crystalline flaws, with gaps resulting when elements attached in an inappropriate order prevent access to other sites. The approach we present, with the central concern of preventing such gaps, could thus be useful for self-assembly applications, guaranteeing a user correct completion of a desired assembly without requiring them to be involved in the details of the assembly sequence.

It is also worth discussing work in automating construction that does not take the swarm approach. Some proposals, for terrestrial as well as extraterrestrial construction (Khoshnevis and Bekey 2003; Buswell et al. 2005), involve what may be described as large-scale rapid prototyping approaches, extruding material from a nozzle suspended from a gantry erected above the entire construction site and building up the structure in a series of layers. Where feasible, such an approach could be very effective; but swarm systems may have advantages in many settings, particularly those where human intervention is non-trivial. Swarms are likely to be significantly easier to deploy; rather than needing to carefully set up an apparatus that is larger than the structure to be built (a particularly daunting thought in the context of construction on other planets), all that is needed is to transport a potentially disordered and easily packable set of components to the building site. Swarm systems may be more useful in settings where deposition cannot easily be used, again as in outer space or underwater. The range of buildable structures may be greater with the swarm approach: deposition machines, in order not to be limited to structures where all material is directly supported by other material immediately underneath, use a soluble auxiliary material for temporary supports. In large-scale construction scenarios, removing that material later may be difficult. Robots able to crawl over the surface of a structure, in contrast, may build complicated overhanging features directly. In addition, there is the potential for speedup through parallelism and through not being limited to building one layer at a time.

7. Conclusion

In this paper we have presented decentralized algorithms for automated construction of user-specified three-dimensional structures, separating the task into the subproblems of (1) determining whether attachment at any given site is allowed and (2) finding all sites where attachment is allowed. For the first problem, we have described simple rules for attachment, proved their correctness for a large class of structures and showed that the communication required for a distributed implementation scales favorably with the size of the structure. For the second problem, we have compared three algorithms and investigated their scaling behavior with respect to three performance measures.

The simple rules governing block attachment make very simple robot algorithms possible. In addition, these rules may be of particular value in the self-reconfigurable modular robot community as well as in the domain of construction. Many self-reconfigurable systems involve cubic modules and distributed control and need to avoid configurations where sites intended to be occupied are unreachable. The rules we discuss could be used for control in such systems and accommodate conservatively attainable module movement capabilities.

In future work, we hope to extend the class of possible structures to allow the construction of more general three-dimensional shapes including those with loops and fully enclosed regions. One approach to this goal will be to explore alternate rules for block attachment. The plane rule is more restrictive than it strictly needs to be. Two separated blocks could be attached and later connected by the addition of further blocks without causing an insurmountable problem; the problem (Figure 2(C)) arises when four blocks or groups of blocks mutually conflict. A less restrictive rule about separation in a plane may allow more general structures, as well as (because more sites can be permitted for attachment at a time) enabling faster structure completion.

We would also like to apply our approach directly to existing hardware systems for self-assembly (White et al. 2005) or construction (Terada and Murata 2004). We may demonstrate its use with simulations specialized to such systems, or would welcome collaborations with researchers interested in applying these algorithms to their own hardware.

Acknowledgment

This work is partially funded by the National Science Foundation under Grant No. 0523676.

Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Examples of a 10-robot swarm building a user-specified structure

References

Balaguer, C. et al. (2002). FutureHome: an integrated construction automation approach. *IEEE Robotics and Automation Magazine*, **9**(1): 55–66.

Bowyer, A. (2000). Automated construction using cooperating biomimetic robots. *Technical Report*, University of Bath, Department of Mechanical Engineering.

Buswell, R. A. et al. (2005). Investigation of the potential for applying freeform processes to construction. *Proceedings of the 3rd International Conference on Innovation in Architecture, Engineering and Construction*, Rotterdam, The Netherlands, pp. 141–150.

De Rosa, M. et al. (2006). Scalable shape sculpting via hole motion: motion planning in lattice-constrained modular robots. *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, Orlando, FL.

Detweiler, C. et al. (2006). Hierarchical control for self-assembling mobile trusses with passive and active links. *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, Orlando, FL.

Everist, J. et al. (2004). A system for in-space assembly. *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 2356–2361.

Fitch, R., Butler, Z. and Rus, D. (2003). Reconfiguration planning for heterogeneous self-reconfiguring robots. *Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV.

Grushin, A. and Reggia, J. (2006). Stigmergic self-assembly of prespecified artificial structures in a constrained and continuous environment. *Integrated Computer-Aided Engineering*, 13: 289–312.

Jones, C. and Matarić, M. (2003). From local to global behavior in intelligent self-assembly. *Proceedings of 2003 IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, pp. 721–726.

Jones, C. and Matarić, M. (2004). Automatic synthesis of communication-based coordinated multi-robot systems. *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 381–387.

Kamimura, A. et al. (2004). Distributed adaptive locomotion by a modular robotic system, M-TRAN II (from local adaptation to global coodinated motion using CPG controllers). *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 2370–2377.

Khoshnevis, B. and Bekey, G. (2003). Automated construction using contour crafting—applications on earth and beyond. *Journal of Rapid Prototyping*, **9**(2): 1–8.

Klavins, E., Ghrist, R. and Lipsky, D. (2006). A grammatical approach to self-organizing robotic systems. *IEEE Transactions on Automatic Control*, **51**(6): 949–962.

Kotay, K. and Rus, D. (2004). Generic distributed assembly and repair algorithms for self-reconfiguring robots. *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan.

Melhuish, C., Welsby, J. and Edwards, C. (1999). Using templates for defensive wall building with autonomous mobile ant-like robots. *Proceedings of Towards Intelligent Autonomous Mobile Robots '99*, Manchester, UK.

Støy, K. (2006). How to construct dense objects with self-reconfigurable robots. *Proceedings of the European Robotics Symposium*, Palermo, Italy, pp. 27–37.

Støy, K. and Nagpal, R. (2004). Self-reconfiguration using directed growth. *Proceedings of Distributed Autonomous Robotic Systems 2004*, Toulouse, France.

Terada, Y. and Murata, S. (2004). Automatic assembly system for a large-scale modular structure: hardware design of module and assembler robot. *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 2349–2355.

Théraulaz, G. and Bonabeau, E. (1995). Coordination in distributed building. *Science*, **269**: 686–688.

Vassilvitskii, S., Yim, M. and Suh, J. (2002). A complete, local and parallel reconfiguration algorithm for cube style modular robots. *Proceedings of 2002 IEEE International Conference on Robotics and Automation*, Washington, DC, pp. 117–122.

Werfel, J. (2006). Anthills built to order: automating construction with artificial swarms. *Ph.D. Dissertation*, Cambridge, MA, Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory.

Werfel, J. (2007). Robot search in 3D swarm construction. *Proceedings of 1st IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, Cambridge, MA, pp. 363–366.

Werfel, J., Bar-Yam, Y. and Nagpal, R. (2005). Building patterned structures with robot swarms. *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Edinburgh, UK.

Werfel, J. et al. (2006). Distributed construction by mobile robots with enhanced building blocks. *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, Orlando, FL.

Werfel, J. and Nagpal, R. (2006). Extended stigmergy in collective construction. *IEEE Intelligent Systems*, **21**(2): 20–28.

White, P. et al. (2005). Three dimensional stochastic reconfiguration of modular robots. *Proceedings of Robotics: Science and Systems I*, Cambridge, MA.

Downloaded from ijr.sagepub.com at University of Cape Town on April 26, 2016