Collective Construction of Environmentally-Adaptive Structures

Justin Werfel* jkwerfel@eecs.harvard.edu
Donald Ingber† donald.ingber@childrens.harvard.edu
Radhika Nagpal* rad@eecs.harvard.edu

*Electrical Engineering & Computer Science
Harvard University
Cambridge, MA 02138

†Department of Pathology, Harvard Medical School
Vascular Biology Program, Children's Hospital
Boston, MA 02115

***Abstract*— We describe decentralized algorithms by which a swarm of simple, independent, autonomous robots can build two-dimensional structures using square building blocks. These structures can (1) exactly match arbitrary user-specified designs, (2) adapt their shape to immovable obstacles, or (3) form a wall of given minimum width around an environmental feature. These three possibilities span the range from entirely prespecified structures to those whose shape is entirely determined by the environment. Robots require no explicit communication, instead using information storage capabilities of environmental elements (a form of "extended stigmergy") to coordinate their activities. We provide theoretical proof of the correctness of the algorithms for the first two types of structures, and experimental support for algorithms for the third.**

I. Introduction

Automated construction systems may one day enable the routine building of structures in settings inconvenient or dangerous for humans to work in, as with extraterrestrial sites or disaster areas. Such systems will need to adapt their activities to features of the environment they encounter. These may include, e.g., immovable obstacles in a workspace that make it impossible to build a desired structure exactly as originally specified, or environmental elements that define what a structure should look like (as with building a barrier around a hazardous waste spill).

In this paper we present decentralized algorithms by which a swarm of autonomous mobile robots may build environmentally-adaptive 2D structures of arbitrary user-specified design (Fig. 1). These structures may (1) exactly match a provided design; (2) match the design to the extent possible but be built around immovable obstacles present in the environment; or (3) surround specified environmental elements with a border of a given minimum width. The key contribution of this paper is the treatment of structures which are partially user-specified and partially adaptive to features of the environment.

Our framework may be described as a bipartite modular system, consisting of non-actuated modular units (square blocks) out of which the structure is built, and mobile robots that put them into place. The algorithms rely on simple and limited robot capabilities, based only on local information and not requiring capabilities like inter-robot communication or GPS access. Blocks, though non-mobile, can store information (e.g., using writable RFID tags), using

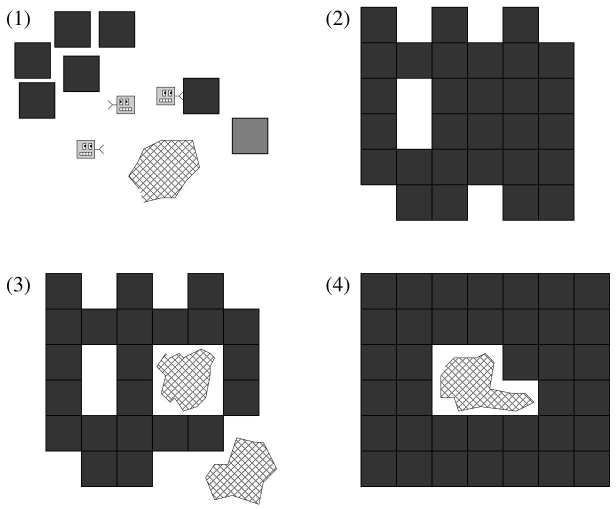

Fig. 1. Examples of the kind of framework we describe. (1) A swarm of identical robots builds using a supply of square blocks, in a workspace that may have immovable obstacles (cross-hatched), starting with a deployed 'seed' block (lighter shading). The goal is a structure whose shape (2) matches a prespecified design, (3) accomodates obstacles, or (4) surrounds an obstacle with a wall of specified minimum thickness (here, 2).

the "extended stigmergy" framework [1], [2]. We present cellular-automaton-type rules, based on occupancy of neighborhood sites, by which robots determine when to attach blocks, such that collectively they will reliably produce a desired structure. The correctness of these rules is supported by theoretical proofs or experimental evidence.

Section II reviews related work. Section III describes our framework and assumptions. Section IV describes algorithms that allow a swarm to reliably build any user-specified structure; section V addresses how the swarm can modify that design to leave room for immovable obstacles in its way. Section VI describes algorithms that allow a swarm to build an enclosure around a preexisting environmental feature. Section VII discusses considerations associated with the use of multiple robots. Section VIII describes simulation experiments. Section IX concludes.

II. Related Work

Most work on collective construction focuses on issues such as hardware design [3], communication [4], or minimalism [5]. Such studies typically have the goal either of producing structures that are wholly determined by the environment [5] or wholly prespecified [3], [4]; the latter, as a rule, require each step of construction to be specified by the user, while our system automates this task given the desired outcome.

In previous work [1], [2], we have presented decentralized algorithms for collective construction of solid 2D structures whose shape is fully prespecified, along with proofs of these algorithms' correctness and a hardware prototype implementing such a system. Here we extend that work in two major respects: (1) key elements of a structure's shape are not specified in advance but rather determined by the environment in which it is built; (2) different assumptions about physical motion constraints (§III) allow provable construction of a much larger class of structures, including shapes with arbitrary enclosed gaps.

A research area with a very similar goal is that of programmed self-assembly, where tiles are designed to have edge-binding properties such that they self-assemble into a desired configuration when randomly mixed together [6], [7]. The mixing is effectively performed in our framework by the mobile robots. Many studies in programmed self-assembly are susceptible to crystalline defects, where flaws such as internal gaps appear in undesired locations, often as a result of neglecting to take into account physical restrictions on tile movement. The approach described here is explicitly designed to avoid such defects, and so could be usefully applied to self-assembling systems in situations where environmentally-adaptive assemblies are desired.

III. Framework

The system we consider consists of three types of elements: mobile robots, movable building blocks, and fixed environmental features that act as obstacles. Robots assemble a structure from a supply of blocks, subject to obstacles in the workspace. One block is initially planted in the workspace as a seed from which the structure grows.

Robots are identical and interchangeable. They can:

- move freely in two dimensions;
- locate the growing structure and a supply of free blocks as building material[1];
- once having found the structure in progress, follow its perimeter;
- and evaluate whether blocks or obstacles occupy each of the eight sites surrounding the one they occupy.

There is *no explicit communication* between robots; robots can detect one another's presence at short ranges and modify their movement accordingly, for instance to avoid collisions. Also, at least one of the following is assumed to be true: two robots going opposite directions can pass each other in a tunnel one block-length wide, and/or two robots that meet can exchange building material (see §VII).

Blocks are square, interchangeable, and can be attached to other blocks with self-aligning connectors on all four sides. Robots can write a few bytes of information to blocks, and read it back later. This information storage could, e.g., be accomplished simply and inexpensively using nonpowered, writable RFID tags.

Our previous work [1], [2] considered a more conservative set of assumptions about physical movement constraints, making lighter demands on required robot capabilities (making physical implementation of such a system easier), but restricting the class of structures it can build (with provably correct behavior only for structures with no enclosed gaps). The assumptions in the present work require more capable robots, but can build any connected structure the user might care to specify.[2] The important differences in the assumptions about component capabilities are that here, robots are assumed to be able to (1) maneuver down narrow tunnels as little as one block-length wide, (2) attach blocks at any sites they themselves are able to reach, and (3) determine the occupancy of each of the eight block-sized sites surrounding their position. (To facilitate the first two of these assumptions, blocks might be deformable or collapsable cubes, or robots might carry blocks out of the plane of the structure.)

We consider two types of structure specifications in this paper: those where the goal is to build a particular desired structure (§IV) subject to environmental obstacles (§V), and those where the goal is to build a wall around an environmental obstacle unknown in advance (§VI). In the first case, robots all have a copy of the *shape map*, a representation of the desired structure that can be used to determine, given the location in the coordinate system embodied by the assembled blocks, whether a site is ultimately intended to be occupied by a block or left empty. The desired structure must be a single connected unit, and include the initially present seed block.

The overall approach in all cases is for robots to follow the procedure given by the following pseudocode (Fig. 2):

```
1: loop
2:   if carrying building supplies then
3:     if shape map specifies block at this site and state
         of surrounding sites satisfies CA rules then
4:       attach block at this site
5:       write information to block
6:     else
7:       follow perimeter to next empty site
8:   else
9:     fetch additional supplies and return to perimeter
```

The "CA rules" referenced in line 3, and the information to be written to the block in line 5, are elaborated in the sections below.

IV. Prespecified structures, obstacle-free environment

In this section and the next we describe rules for robots to follow that will result in correct construction of particular

[1]These locating tasks could be implemented as simply as with a random walk [4], or by using beacons to mark the locations of the structure and caches of blocks [1]. We assume robots have nothing like GPS access.

[2]Structures consisting of two or more disconnected parts are considered to be separate structures.

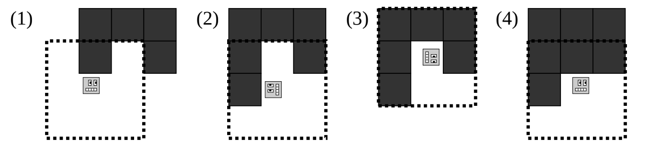

Fig. 2. Four steps executed by a robot using the CA rule of Fig. 3 (Rule A). (1) The 8-cell neighborhood (dotted line) does not match either template, so the robot will attach a block at this site and move to an adjacent site. (2) Here the neighborhood matches the second template, so the robot will proceed along the perimeter without attaching a block. (3) Neither template matches, so a block will be attached here. (4) The robot revisits the site where attachment was previously forbidden; the template no longer matches, and attachment is now allowed. (Material carried by the robot is not shown.)

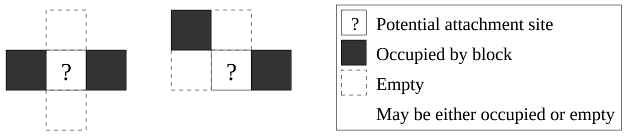

Fig. 3. CA rule ("Rule A") for structures without holes. If robots attach at any sites specified by the shape map, unless their local neighborhood matches either of these patterns (or rotations or reflections thereof), then they will reliably build structures matching the shape map.

desired structures, specified by a shape map. Correct construction requires a partial ordering on block attachment, to avoid situations where sites intended to be occupied remain forever empty because they have become inaccessible to robots. That ordering should not be more restrictive than necessary, in order to take maximum advantage of the parallelism of the swarm and achieve the fastest possible construction.

Robots along the perimeter of the structure can determine their position in the shape map's common coordinate system by storing coordinate information in the blocks. The seed block starts out with its coordinates already stored; robots come to the structure, read their position from any block, and, upon attaching a new block to the structure, write its new coordinates to it.

Here we consider an obstacle-free environment, discussing specified structures without internal holes (§IV-A) and those intended to contain holes (§IV-B).

A. *Structures without holes*

If robots refrain from attaching a block when their local 8-neighborhood matches either of the two templates shown in Figure 3, and attach otherwise at any sites specified by the shape map, they will provably and reliably produce the structure specified by the shape map.

This rule (call it Rule A) is based on the robots' ability to travel down any tunnel. As a result, a robot can reach any site bordering the perimeter of the partial structure, as long as any path to reach that site exists. The only way to eliminate such a path is by completing an unbroken wall between two previously connected areas, thus changing the topology of the space. Rule A is constructed so as to prevent such changes.

As a result, sites along the perimeter of the partial structure will come to be occupied; as the structure grows, more sites become accessible, until the structure is complete. At no point do any empty sites become inaccessible, and so the structure will reliably be completed to match the shape map.

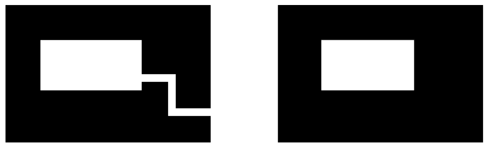

Fig. 4. An example of how a structure with no internally enclosed areas (left) can be converted to one with an internal hole (right) by filling in a tunnel.

The Appendix gives a complete proof of the correctness of Rule A.

B. *Structures with holes*

The assumptions about robot mobility make it straightforward to extend the above approach to handle structures designed to have internal holes. Consider how a structure with no internally enclosed areas can be converted to one with an internal hole (Figure 4).

This figure suggests important points about closing off internal areas. First, the enclosure is accomplished by filling in some tunnel connecting the internal region with the outside space. The tunnel may be of any shape, not necessarily a single block-length wide as in the figure, but topologically the transition occurs in this way.

This tunnel could potentially be filled in starting at any point along its length. However, the starting point should be unique. Multiple starting points, initiated independently by different robots, would result in additional closed-off areas. Robots outside such closed segments of the tunnel would be unable to get in to contribute to filling them in; and robots inside would be unable to get out, ending up trapped in tiny, unfillable pockets of their own making. Similar reasoning leads to the conclusion that the unique starting point should be at the end of the tunnel adjacent to the internal hole. Otherwise at least part of the tunnel would need to be filled in by robots in the region being sealed off; they would unavoidably end up trapped in the hole, and if they lacked sufficient building material, the tunnel (and hence the structure) would remain incomplete.

Observations of local configurations of blocks cannot distinguish between an interior region and the outside workspace. For this reason, the shape map needs to specify which sites are meant to be left empty as parts of internal holes, as distinct from empty sites that can always be reached from the outside workspace. Alternatively, robots can be left to calculate this distinction for themselves. In the following we assume that a robot knows whether a space intended to be left empty is part of an internal hole or connected to the outside workspace.

A modification to Rule A that provably extends it to handle structures with arbitrary internal holes is as follows: Any neighboring site which is meant to be left empty as part

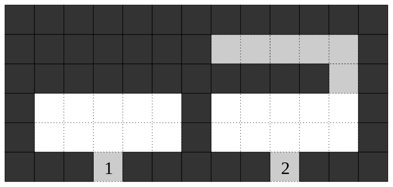

Fig. 5. Attachment is permitted at site 1 under Rule A′, because the site is the last empty one bordering an internal hole, so that neighboring empty sites belonging to that hole are treated as occupied and the neighborhood ceases to match a Rule A template. Attachment is not permitted at site 2, because other sites bordering the associated internal hole remain empty. Blocks have dark shading; sites meant to be occupied have light shading.

of an internal hole is to be treated as an occupied site in the templates of Rule A, if the potential attachment site in question is the last empty site bordering that hole (Fig. 5). Call this new version Rule A′.

A proof of the correctness of Rule A′ can be outlined as follows. Treating a deliberately-empty site as being occupied will sometimes (1) make a robot's neighborhood match a template it otherwise would not have, and forbid attachment that would have been allowed under Rule A; and sometimes (2) keep the neighborhood from matching a template it otherwise would have, and allow attachment that would have been forbidden under Rule A. Enumerating all such possible situations reveals, for these two cases: (1) There is no situation in which Rule A′ forbids attachment that Rule A would have allowed. (2) The only time Rule A′ will allow attachment that separates two regions (the eventuality Rule A was constructed to prevent) is when one region is an internal hole for which construction is entirely completed, and no robot will ever again need to enter that region. Thus Rule A′ will reliably construct any desired structure with any set of internal holes.

Determining that a site bordering an internal hole is the last remaining empty one involves gathering nonlocal information. Robots can easily acquire this information as they follow the structure perimeter, keeping track of whether they return to a previously visited border site without encountering any other empty spaces along the border. Because the framework discussed here involves only adding building material, never removing it, no other robot's actions can lead to an inappropriate attachment event based on outdated information.

A final consideration has to do with 'peninsulas' one site wide where blocks should be attached, jutting into concave internal holes, as in the top-left hole in the structure of Fig. 6 (outlined in yellow). The site at the base of the peninsula (purple) is the only one to which blocks of that peninsula can be attached, and so must not be the last site left empty along the external border of the hole: a block must be attached there before both of its neighbors along the external border (blue) are occupied. Again, this nonlocal information can easily be obtained by a perimeter-following robot.

When a robot attaches a block at a site closing off an internal hole, it should do so such that it ends up on the side of the outside workspace, to keep from being trapped inside the hole. The issue of avoiding trapping other robots in internal holes is discussed in Section VII.

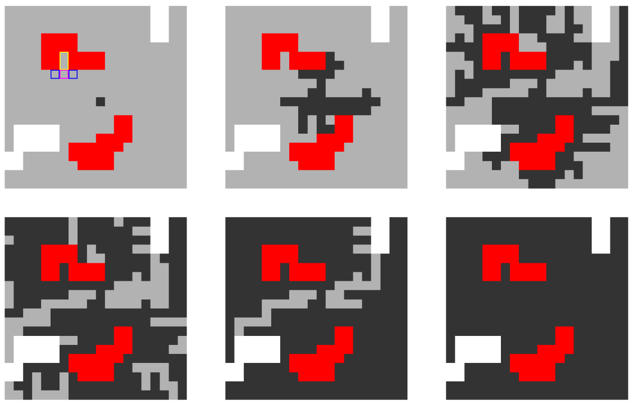

Fig. 6. Successive snapshots during the construction of an example structure. Sites intended by the shape map to be occupied have light shading; blocks have dark shading; internal holes are marked in red. Outlined sites in the first snapshot are discussed in the text.

V. Prespecified Structures, Environment with Obstacles

Suppose the shape map calls for a structure of a given design, but existing, immovable obstacles in the environment (e.g., boulders scattered about the workspace) are in the way of attaching blocks at some desired sites. One approach to such a situation is to build the structure to the greatest extent possible, leaving holes in the original design where immovable obstacles are present. This approach can be directly achieved using Rule A′ discussed above.

The idea is for robots to modify the shape map according to existing obstacles. When a robot is in a site neighboring an obstacle, it reclassifies that site in its copy of the shape map as being part of an internal hole. As construction and robot movement continue, robots will individually update their shape maps such that obstacles become located within a border of sites marked as belonging to an internal hole. That border increases the effective size of the obstacle, allowing robots to maneuver around it while following the perimeter of the hole. Construction then works exactly as above, whether or not an internal hole has an impenetrable obstacle in the middle of it (Fig. 7A).

Because Rule A′ depends only on the occupancy states of sites in a robot's local neighborhood, no separate stage of construction is needed to survey the workspace for obstacles and modify the shape map globally before building starts. Further, the rule only treats internal-hole sites as occupied if the structure surrounding the hole is all but completed; determining that that's the case requires a robot to have circled the entire perimeter of the hole and thus the entire obstacle, and so lack of knowledge of which sites contain obstacles will never lead a robot to attach blocks inappropriately.

Note that if obstacles cut off a part of the structure to make it noncontiguous with the seed, that part of the structure will

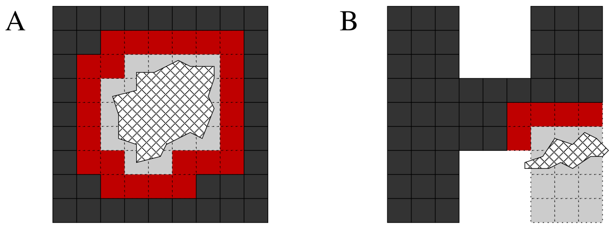

Fig. 7. A: An obstacle (cross-hatched) occupies sites intended to be part of the structure (light shading); robots mark sites bordering the obstacle (red) as being part of an internal hole.
B: An obstacle separates the lower-right leg of the desired structure from the rest, and so prevent it from ever being constructed, although it contains sites not themselves bordering any obstacle.

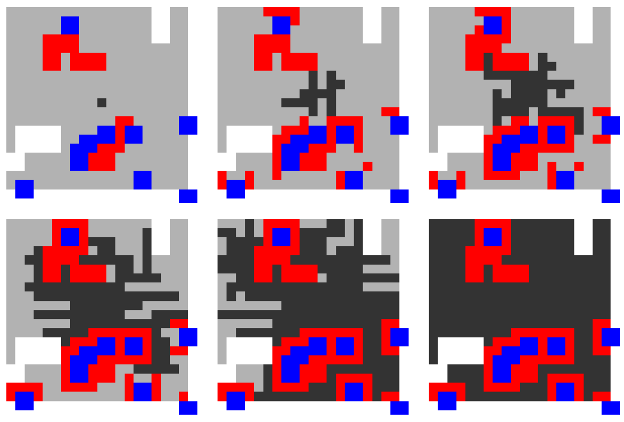

Fig. 8. Successive snapshots during the construction of the structure of Fig. 6 in the presence of obstacles (blue). Sites intended to be occupied have light shading; blocks have dark shading; internal holes, and sites that at least one robot has found to be adjacent to obstacles, are marked in red.

never be built under this framework (Fig. 7B).

Figure 8 shows snapshots during the construction of an example structure in the presence of obstacles.

VI. Fully Adaptive Structures

Here we consider the case where a structure's shape is determined, not by a prespecified design, but by environmental features and high-level functional requirements. Specifically, we address the problem of surrounding an unknown environmental feature with building material out to at least a specified thickness t, as with building a containment barrier around a hazardous waste spill. We assume that a one-block seed is initially planted adjacent to such a feature.

In such a case, no predefined shape map can be specified, and coordinate information is not relevant to the structure. Rather, the important value to be associated with blocks is related to their distance from the obstacle. The seed is given the value 1. The desired final state is for all empty sites outside the wall to be separated from the obstacle by a Manhattan distance of more than t.

First we define Rule A$''$, a modified version of Rule A to prevent closing off unintended holes when a structure is to be built right up to the edge of an obstacle. Rule A$''$ uses the same templates as Rule A (Fig. 3) to forbid attachment,

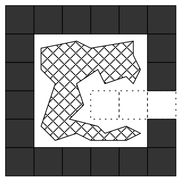

Fig. 9. Example of an obstacle (cross-hatched) with a concavity too narrow to accomodate more than a single block. A wall completely lining the obstacle's perimeter should involve blocks at all three empty sites shown, each marked as having distance 1. However, the left and center sites have no structure adjacent to attach blocks to. Attachment should then be allowed at the rightmost site, where Rule A would otherwise forbid it, to keep construction from stalling.

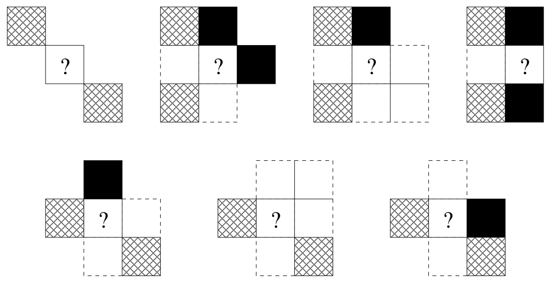

Fig. 10. Templates for Rule A$''$. Treating sites containing obstacles as though they contained blocks, a robot uses Rule A (Fig. 3) to determine where not to attach blocks. However, if the local neighborhood matches any of these 7 templates (or rotations or reflections thereof), the robot makes an exception and attaches at a site that would otherwise be forbidden by Rule A. ? marks the potential attachment site; cross-hatched sites contain obstacles; solid sites may contain either obstacles or blocks; sites outlined by dotted lines are empty; unmarked sites may be occupied or empty.

treating sites into which the obstacle extends as though they were occupied by blocks. Since blocks can only be attached to other blocks, not directly to the obstacle, exceptions to these templates must be made for cases where an obstacle has concavities wide enough for only a single block, too narrow for robots to build the structure into or through such openings (Fig. 9). Fig. 10 shows the templates necessary and sufficient to handle all such cases (proof omitted).

We then define rules for attachment (Rule B) as follows:

1) If the obstacle extends into any of the eight neighboring sites, plan to attach a block here and label it with value 1.
2) Otherwise, if there are at least two neighboring blocks with the same value:
    a) Let v be the smallest value that appears more than once among all neighboring blocks. If two neighboring blocks labeled with v are adjacent to one another, plan to attach a block here and label it one greater than the minimum value of all neighboring blocks.
    b) Otherwise, plan to attach a block here and label it with value v.
3) Refrain from attaching if Rule A$''$ forbids it, or if the intended label value is greater than t. Otherwise, attach as planned.

Figure 11 shows snapshots during the construction of such

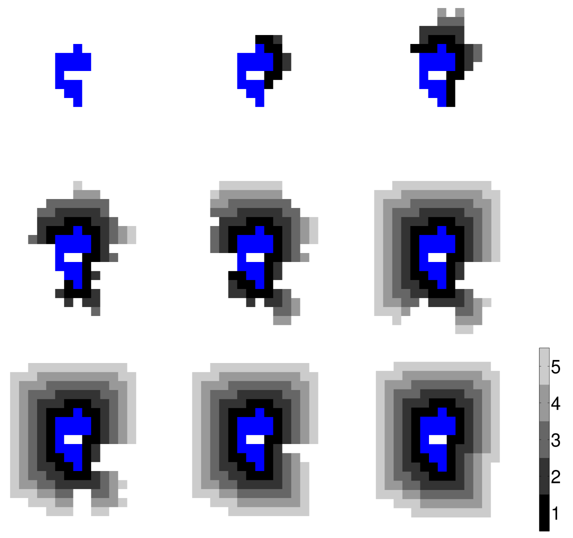

Fig. 11. Snapshots during the construction of a wall of thickness $t = 5$ around a randomly generated obstacle (blue). Blocks are colored by value (legend shown at right).

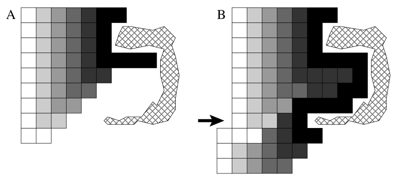

Fig. 12. A: A possible early stage of construction around an obstacle (cross-hatched) according to Rule B, with $t = 6$. Blocks attached with higher values are shown with lighter colors. B: Construction associated with the lower part of the obstacle is forced to go around the wall built in the earlier stage, resulting in an empty site (arrow) separated from the obstacle by only five blocks in the row indicated.

a wall around an obstacle.

Full theoretical analysis of Rule B will appear in future work. The rule leads to construction of a wall of at least thickness t around an obstacle in nearly all situations. The exception that can occur is shown in Fig. 12, where early construction can produce a wall near one part of an obstacle that then "shields" empty sites from a closer part of the obstacle. This exception can occur only if $t > 5$, and if the obstacle is concave (in the sense of extending into separated sites in the same row or column). Thus we conclude that Rule B will (1) always result in a wall of at least thickness t, for general obstacles and $t \leq 5$; (2) always result in a wall of at least thickness t, for any t and convex obstacles; (3) for any t and general obstacles, occasionally leave an site in the external workspace empty when it is fewer than t steps from the obstacle. Such errors are rare (§VIII).

A partial exception to the above may occur for obstacles shaped to have a bottleneck (Fig. 13). If a value of t is specified such that the width of the bottleneck is $\leq 2t$, but the bulb of the bottle contains a space wider than $2t$, the bottle will not "fill", leaving sites along the bottleneck that are contiguous with the outside workspace but fewer than t steps from some part of the obstacle. In such a case, all sites outside the convex hull of the wall will still be separated from all parts of the obstacle by at least t blocks. This situation arises because the wall called for is one with an internal hole, which Rule B will not produce. If robots have the capacity to keep track of where they have observed attached material, and conclude that an area should be sealed off, a modification to the rule like that of Rule A$'$ will allow the bottleneck to be filled and eliminate this problem.

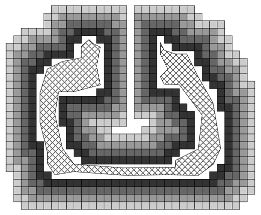

Fig. 13. If an obstacle has a bottleneck no wider than $2t$, and a courtyard wider than $2t$, Rule B will leave a passage through the bottleneck, with sites separated from the obstacle by fewer than t blocks. (Here $t = 4$.)

VII. Multiple robots

The approaches discussed above work with any number of robots, with robots acting independently and without explicit coordination. However, there are situations where it is useful to explicitly consider the interactions of multiple robots.

One issue is what happens when two robots encounter each other in a narrow tunnel heading in opposite directions. In section III, we specified that either robots should be able to pass each other in this situation, or they should be able to exchange building material. In the former case, robots can effectively "pass through" each other, from the perspective of occupying sites in the grid. In the latter case, the robots can each turn around and go back the way they came; since robots are interchangeable, this is equivalent to having them pass through each other; the exchange of building material ensures material availability everywhere. More generally, any time one robot needs to pass by another one, they can exchange roles and building material. This approach, with its lighter requirements on the spatial extent of the robots, may make their mechanical design easier.

Another case calling for consideration of other robots is when first starting to close off an internal hole: it is preferable (though not, strictly, necessary) to avoid sealing other robots inside. A sufficient way to handle this situation is for the robot to wait some time at the entrance to the hole before sealing it, acting as a gatekeeper. Other robots approaching the site from the internal side are allowed to leave; others approaching from the external side are sent back. The period the robot should wait is the time it takes for a robot to travel all the way around the perimeter of the hole; this period depends on the geometry of the hole, which is information available to the gatekeeper robot. As above, if robots are unable to physically pass one another in narrow tunnels, each robot coming from the inside of the hole and reaching the exit can take on the gatekeeper role, sending the previous one away; the last robot to exit will be the one to seal the hole.

VIII. Experiments

In this section, we discuss two types of experiments, (1) testing the reliability of Rule B in building walls of specified thickness around randomly generated obstacles, and (2) investigating how the time required to build structures under both Rules A$'$ and B scales with the number of robots.

To evaluate the reliability of Rule B, we performed simulation experiments building a wall around an obstacle randomly generated by starting with a single site and following a random walk for 30 steps. We performed 500 independent runs for each of several cases and checked for empty sites separated from any part of the obstacle by fewer than t blocks.

For $t = 5$, no such errors were encountered. For $t = 10$ and obstacles restricted to be convex, no errors were encountered. For $t = 10$ and no restrictions on obstacles, four of the 500 runs each had a single site separated from the obstacle by $t-1$ blocks, for the reason shown in Fig. 12. These results support our claim that Rule B works reliably for $t \leq 5$ and for convex obstacles, and that errors for other cases are rare.

Fig. 14 shows how construction time scales with the number of robots, both when building prespecified structures in the presence of obstacles and when building walls around obstacles. In the former case, a shape map is generated by starting with a 10×10 square and adding ten randomly located 2×2 holes and ten randomly placed 2×2 obstacles (potentially overlapping). In the latter case, random obstacles are generated as above; we use $t = 5$. Robots are initialized at random on the perimeter of a square surrounding the seed, move inward until they reach an obstacle or the structure, follow its perimeter until they reach a site where attachment is allowed, attach a block, and move instantaneously back to the surrounding perimeter. A robot will not enter a site already occupied by another robot.

Under these conditions, the algorithms are capable of handling many robots at once with little effect of interference. The time required to complete a structure decreases as the number of robots is increased, with nearly an N-fold speedup for N robots. Diminishing returns with increasing numbers of robots are visible, as the number of sites available for attachment at any given time saturates and additional robots have less to contribute.

IX. Conclusion

We have presented decentralized algorithms by which a swarm of autonomous mobile robots may build 2D structures whose shape adapts to obstacles in their environment. These structures may match, to the greatest degree possible, an arbitrary high-level design provided by the user, or may surround environmental features whose shape determines their own. We presented proof of the correctness of the algorithm for prespecified structures, and empirical evidence characterizing the correctness of the algorithm for environmentally-determined structures. These algorithms require no communication or nonlocal knowledge from the robots, and are robust to run-time variations in the number of robots and the order and timing of their actions. Information stored in building blocks is used to coordinate the progress of the construction.

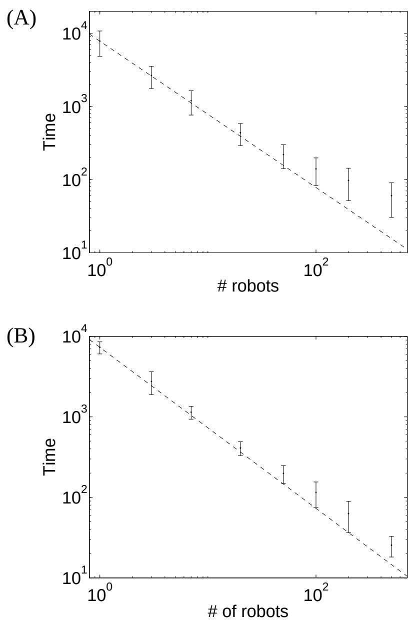

Fig. 14. Time required for N robots to build (A) a prespecified structure in the presence of obstacles (§V) or (B) a wall of thickness 5 around a randomly generated obstacle (§VI). Error bars reflect averages over 100 independent runs. The dotted line shows what a factor-of-N speedup would look like, i.e., perfect parallelism.

The approaches described here could also be implemented (more easily, but less cheaply) with blocks with embedded computation capabilities, as in the "communicating blocks" framework of [2]. For instance, such blocks used to build a wall around an obstacle could dynamically update their labels to reflect their true distance from the obstacle, making stale information a transient problem and making it easy to guarantee a wall of any thickness around an obstacle of any shape.

In future work, we plan to better characterize and refine the algorithm for environmentally-determined structures; to extend these algorithms to construction of three-dimensional structures; and to implement them on a hardware testbed [1].

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

References

[1] J. Werfel, Y. Bar-Yam, D. Rus, and R. Nagpal, "Distributed construction by mobile robots with enhanced building blocks," in *Proc. ICRA 2006*, Orlando, USA, 2006.

[2] J. Werfel and R. Nagpal, "Extended stigmergy in collective construction," *IEEE Intelligent Systems*, vol. 21, no. 2, pp. 20–28, Mar.-Apr. 2006.

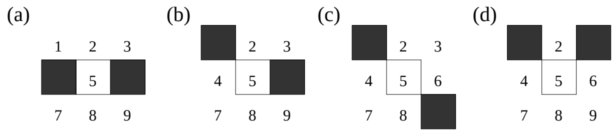

Fig. 15. Four possibilities in which a wall from the left side to the right side of the local neighborhood can be completed, separating two areas, by attaching a block in the central site (5). Rotating these cases 90 degrees gives the possibilities for a wall from top to bottom. Solid sites are occupied by blocks; unmarked sites are unspecified. Numbers label sites for the description in the text.

[3] Y. Terada and S. Murata, "Automatic assembly system for a large-scale modular structure: hardware design of module and assembler robot," in *Proc. IROS 2004*, Sendai, Japan, 2004, pp. 2349–2355.

[4] C. Jones and M. Matarić, "Automatic synthesis of communication-based coordinated multi-robot systems," in *Proc. IROS 2004*, Sendai, Japan, 2004, pp. 381–387.

[5] C. Melhuish, J. Welsby, and C. Edwards, "Using templates for defensive wall building with autonomous mobile ant-like robots," in *Proc. TIMR 99*, Manchester, UK, 1999.

[6] J. Bishop, S. Burden, E. Klavins, R. Kreisberg, W. Malone, N. Napp, and T. Nguyen, "Self-organizing programmable parts," in *Proc. IROS 2005*, Edmonton, Canada, 2005.

[7] D. Arbuckle and A. Requicha, "Active self-assembly," in *Proc. ICRA 2004*, New Orleans, Louisiana, 2004, pp. 896–901.

APPENDIX: PROOF OF RULE A

Here we show that Rule A of Fig. 3 will result in the correct construction of any specified structure, without either (1) unwanted holes, where remaining sites should have blocks attached but cannot be physically reached, or (2) deadlock, where remaining sites can be reached but the rule forbids attachment anywhere.

First we show that Rule A prevents the completion of any unbroken wall between two previously connected areas. Any gap of at least 1 site wide can be passed through by a robot. In the two situations of Fig. 3, then, attaching a block at the site in question will complete an impassable wall, potentially separating two areas. We now show that those two situations cover all cases where an impassable wall could be created.

From the perspective of the local neighborhood, a situation where attaching a block at the current site completes an impenetrable wall must take one of the four forms shown in Fig. 15 (plus rotations and reflections), since every step along that wall from one block to the next must be either straight or diagonal. We consider the four cases in turn, showing that each is covered by one of the two templates of Rule A.

(a) Sites 4 and 6 are occupied. If site 2 or 8 is occupied, an impenetrable wall already exists and adding a block at site 5 creates no new separations. Thus attachment is problematic only if sites 2 and 8 are empty. This gives the first template of Rule A.

(b) Sites 1 and 6 are occupied. If site 4 is occupied, we again have the situation of case (a). If site 2 is occupied, an impenetrable wall already exists and adding a block at site 5 creates no new separations. This attachment is problematic only if sites 2 and 4 are empty. This gives the second template of Rule A.

(c, d) Attachment is only possible if there is an adjacent block to attach to. Thus at least one of sites $\{2, 4, 6, 8\}$ must be occupied. Adding a block at any of those four sites in either case gives back case (b). The exception is site 2 in case (d), in which case once again an impenetrable wall already exists and adding a block at site 5 creates no new separations.

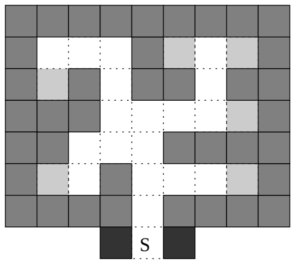

Fig. 16. Example where the occupied sites (dark shading) that make an empty site S unfillable are connected by a 'chain' of blocks, enclosing a space within which every empty site is meant to be occupied, and at least one site (light shading) may be.

Having shown that Rule A prevents the creation of unreachable empty regions, it remains to show that it is not too restrictive—that it will never lead to deadlock situations, where sites meant to be occupied remain unfilled simply because the rule forbids it. The proof is by contradiction.

Suppose that a state is reached where every remaining site the shape map specifies should be occupied is reachable by robots but forbidden by one of the two templates of Rule A. Consider any empty site S meant to be occupied. Because the structure must be built as a single continuous unit, the two occupied sites in that template must be connected by some chain of blocks (which may be arbitrarily wiggly) enclosing some finite area (Fig. 16). Because S is meant to be occupied, and the structure may not contain any holes, all empty sites in the enclosed area must be meant to be occupied.

It is possible to enumerate all possible configurations of existing blocks starting from S. The details are omitted for space considerations, but there are a finite number of templates that may describe sets of blocks spreading out from S such that all empty sites have attachment forbidden. These templates take the appearance of narrow (one block wide) tunnels, and junctions of such tunnels. Any configuration not matching one of these templates allows attachment somewhere.

The area enclosed by the chain of blocks thus takes the form of a nest of narrow tunnels. These may split and multiply to profusion, but every tunnel must end—either in a dead end or a wider open space, either of which will allow attachment. Nor is it possible for a tunnel to avoid this kind of ending by joining up with another tunnel: such a loop would enclose an isolated group of blocks, which is not possible in a connected structure. The nest of tunnels must have a tree structure. Attachment somewhere is therefore possible, giving a contradiction: deadlock cannot occur.

Thus Rule A prevents all dead ends, both those due to unreachable areas and those due to excessive restrictions on attachment, and will therefore lead to the successful completion of any requested solid structure.