Auton Robot (2012) 33:323–336
DOI 10.1007/s10514-012-9305-0

Construction with quadrotor teams

Quentin Lindsey · Daniel Mellinger · Vijay Kumar

Received: 23 September 2011 / Accepted: 5 June 2012 / Published online: 22 June 2012

Abstract We propose a new paradigm for construction in which teams of quadrotor helicopters assemble 2.5-D structures from simple structural nodes and members equipped with magnets. The structures, called Special Cubic Structures (SCS), are a class of 2.5-D truss-like structures free of overhangs and holes. Quadrotors equipped with grippers pick up, transport, and assemble the structural elements. The design of the nodes and members imposes constraints on assembly, which are incorporated into the design of the algorithms used for assembly. We show that any SCS can be built using only the feasible assembly modes for individual structural elements and present simulation and experimental results for a team of quadrotors performing automated assembly. The paper includes a theoretical analysis of the SCS construction algorithm, the rationale for the design of the structural nodes, members and quadrotor gripper, a description of the quadrotor control methods for part pickup, transport and assembly, and an empirical analysis of system performance.

Keywords Robotic assembly · Aerial vehicles · Aerial manipulation

1 Introduction

There is a small and growing class of applications in which robots are used in assembly and construction of structures (Joo et al. 2007). These applications usually require extremely structured and expensive environments, which are typical in industrial automation used by automotive, electronics and packaging industries. This paradigm relies on a static environment in which absolute positions and orientations of parts and fixtures remain unchanged so that industrial robots can be programmed to pick, place and assemble with relatively simple and extremely reliable position and hybrid controllers with very little adaptation or planning (Groover 2007).

There are many assembly/construction applications in which the environment is less structured, e.g. ship building or aircraft assembly industry. These applications require a fixed-position layout where resources are brought to the product. While robots may be involved in such tasks as making long continuous welds along the hull of the ships or in the manufacture of prefabricated sections, human workers are closely involved in operating machinery used for assembly or installation of components.

Rotorcraft are used in construction work, especially for aerial lifting or transport to hard-to-access sites including downtown skyscrapers, mountainous terrain and oil rigs, or tasks requiring assembly of tall towers. However, aerial vehicles are operated manually even though recent work (Bernard and Kondak 2009; Henderson et al. 1999; Michael et al. 2011) suggests that robotic helicopters can outperform even the most skilled human pilots in many applications.

In this paper we will explore the assembly of three-dimensional structures similar to those involved in construction of scaffolds, tower cranes, skyscrapers, and high-voltage towers using autonomous aerial robots. As in any manufacturing application, it is necessary to employ basic design for assembly principles (Boothroyd and Knight 1993) and to ensure that the parts must be matched to the robots and end-effectors that are assembling them. As shown in Fig. 1, the design process yields part designs with speci-

Q. Lindsey (✉) · D. Mellinger · V. Kumar
220 S. 33rd Street, Philadelphia, PA 19104, USA
e-mail: quentinl@seas.upenn.edu

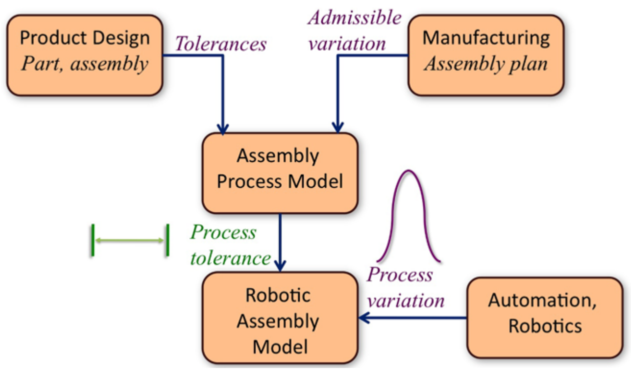


Fig. 1 The admissible process tolerance resulting from product designs and manufacturing plans must be matched to the process variation expected of robotic assembly

fied tolerances and an assembly plan that should be consistent with these tolerances. The resulting assembly process has an associated process tolerance, which in turn should be compatible with the inevitable process variation introduced by robotics. Thus, naively applying robotics technology to an automation problem is fraught with difficulties as the American automotive industry learned in the 1980's.

However, the process tolerance is considerably bigger in construction applications. While the challenges of unstructured environments are more formidable, the tolerances for assembling beams or lowering booms are in the range of millimeters to centimeters. In contrast, the assembly of products in manufacturing plants often require sub millimeter to micron level precision. Thus robotics would seem to be viable in construction applications, even though there may be significant process variation in unstructured environments.

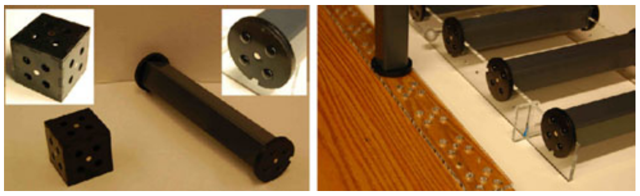
Rather than designing robots for specific construction applications, we consider structures that lend themselves to assembly with off-the-shelf aerial robots. We believe this is justified given the state of the art in aerial robotic assembly. In addition, we consider part designs suitable for robotic assembly (Galloway et al. 2010) and design truss-like elements that can easily snap together and end-effector that are suitable for aerial grasping, transport and assembly. We show that a team of aerial robots (quadrotors) are able to construct structures automatically from simple structural nodes and members using carefully-designed assembly modes compatible with the part design. The paper includes a theoretical analysis of the construction algorithm, a description of the quadrotor control methods for part pickup, transport and assembly, and an empirical analysis of system performance with multiple quadrotors.

While the work on assembly by aerial robots is quite limited, this paper builds on extensive robotics literature in the areas of robotic assembly (Sanderson et al. 1990), robotic grasping (Pounds and Dollar 2010), autonomous helicopters (Bouabdallah 2007; Shen et al. 2011) and modular robotics (Galloway et al. 2010). In the most relevant work (Galloway et al. 2010), the authors create a system in which stationary robotic manipulators are used to build truss-like 3-D structures from parts that are fed to them. Instead of climbing the structure to add additional parts, each 2-D level is automatically built and elevated to a higher level. A benefit of this is that the system can be transplanted to any environment without reconfiguration.

Recent work has examined using local rules and stigmergy to build 2-D structures with groups of robots. Robots in this work encircle the current structure placing blocks at locations determined exclusively from current state of the structure. With increased block capabilities, speed and robustness of construction can be increased while the capabilities of the robots can be reduced (Werfel and Bar-yam 2006; Werfel et al. 2007; Petersen et al. 2011). Purely stochastic systems also use local rules derived from interaction rates to dictate how parts are received (Napp and Klavins 2010) or whether parts are assembled or disassembled (Matthey et al. 2009). While these systems offer robustness to failure due to environmental or mechanical causes, they provide deterministic probabilistic guarantees of correctness at the expense of large part/robot population as well as multiple tuning parameters.

Our work is related to the two bodies of work described above but differs in two important ways. First, our approach to construction is deterministic. The structure blueprint is automatically translated into a deterministic plan for assembly of simple cubic structures. Second, and perhaps more importantly, all the assembly is performed by aerial robots, which introduces constraints unique to the coordination of aerial robots, aerial grasping and assembly with low-complexity grippers. We also note that this work builds on our own previous work. In particular, the experimental platform used here and the basic infrastructure including the underlying control software was described in Michael et al. (2010). In addition, part of this paper was presented in a conference (Lindsey et al. 2011).

This paper will be organized as follows: In Sect. 2, we describe our approach to construction including a description of an algorithm used to build a subset of cubic structures and another algorithm, which overcomes some of the shortcomings of the previous algorithm. The experimental infrastructure used for experimentation including the controllers and planners used for the quadrotors is described in Sect. 3 . The system performance is discussed in Sect. 4 including such measures as robustness and efficiency. Finally, we will discuss some advantages of our approach highlighting the main contributions and the main limitations of the work in Sect. 5 which point to obvious directions for future research.



(a) A node (left) and a member (right) with close up views (b) Part bins containing parts

Fig. 2 Parts used to construct special cubic structures

2 Construction

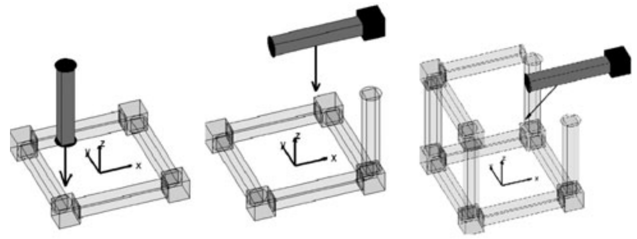
2.1 Part geometry

In this work the basic units of construction are nodes and members as shown in Fig. 2(a). The nodes and members used in this work were inspired by Galloway et al. (2010). Each node is a small 6-sided cubiold that can be attached to six members, each of which is a rectangular prism. A single node attached to a single member constitutes a module (Fig. 10(a)).

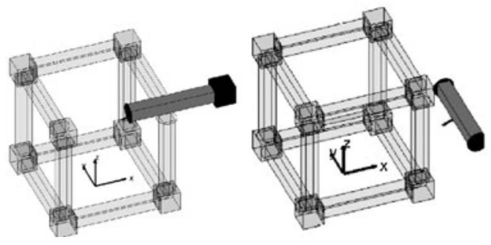
2.2 Special cubic structures

We consider 2.5-D tower-like structures consisting of strata of identical cubes with two constraints. First, we do not allow overhangs (cubes, which are not fully supported from below) since this requires cantilevered members to support more weight than is possible at each joint. Second, because of assembly constraints, we further require that each layer of cubes of has no holes. We define a hole as the specification of a group of connected cubes in a layer to be empty, which results in interior boundary. Note that even a single cube in the interior of the structure, which is specified to be empty in the blueprint, results in a hole. We will call such structures Special Cubic Structures (SCS). The scope of the paper is limited, for the most part, to SCS, although we will relax the no-holes assumption later.

In all cases, horizontal members and vertical members are called beams and columns, respectively while a beam attached to a node is called a module. These components are placed onto the SCS using the five assembly modes shown in Fig. 3. Assembly mode 1 (M_1) is used to place the vertical columns required for each layer. Once the columns are placed, assembly modes 2–5 (M_2–M_5) are used to construct squares in a 2-D stratum as shown in Fig. 4. Due to the design of the hardware, some assembly modes are not possible. For this particular hardware, it is not possible to assemble a horizontal member between two nodes that are already assembled to columns or a module between a node and a horizontal member. The algorithm for constructing SCSs described next takes such constraints into account.

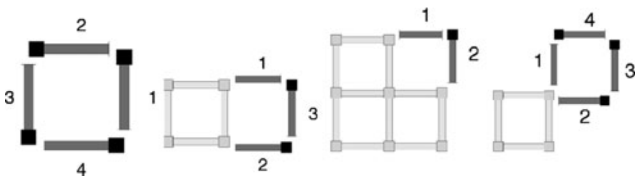


(a) Assembly Mode 1 (b) Assembly Mode 2 (c) Assembly Mode 3



(d) Assembly Mode 4 (e) Assembly Mode 5

Fig. 3 The five assembly modes used to construct a stratum in a SCS



(a) Assembly Primitive 1 (b) Assembly Primitive 2 (c) Assembly Primitive 3 (d) Assembly Primitive 4

Fig. 4 Assembly primitives used to complete squares in a partially-built stratum given that some squares have already been assembled and order of part assembly

2.3 *Wavefront raster* algorithm

We now describe our *Wavefront Raster* (WFR) algorithm for building each 2-D stratum for any SCS. The algorithm builds and completes 2-D squares one at a time, which is attractive from a structural standpoint. By limiting the number of partially-built squares to one, we get better robustness to external disturbances with fewer cantilevered members. This property also requires that parts can only added sequentially. The algorithm is called the Wavefront Raster algorithm because we first mark all the squares that expand the *wavefront* by 1-hop and then build them in the order of a *raster* scan, from west to east and south to north as shown in the pseudocode in Algorithm 1, where modules aligned with the x-axis are referred to as East/West (EW) and beams/modules aligned with the y-axis as North/South (NS). The starting square of each layer is arbitrary because any choice of start square results in a constant order of square placements. For the sake of consistency we choose the southmost square in the westmost column as the start position. The order in which parts are assembled is illustrated using simple examples in Fig. 5.

Algorithm 1 Wavefront Raster (WFR) Algorithm

```
1: build any square in the 2-D region
2: while not finished do
3:    mark squares, which are 8-connected, to already built region
4:    for (westmost column) to (eastmost column) do
5:       build marked squares in column from south to north
```

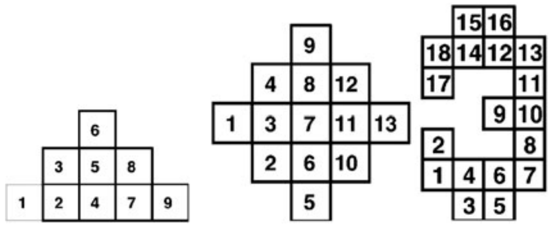


Fig. 5 Several 2-D structures constructed using the WFR algorithm (Algorithm 1). The *number* indicates the order in which a particular square is added to the structure

The Special Cubic Structure (SCS) construction algorithm couples the WFR algorithm with the necessary column placements to build any SCS.

2.3.1 Algorithm properties

Here we point out several properties of the algorithms used for building structures. Recall that the WFR algorithm builds and completes 2-D squares one at a time. Because of this it is not possible to assemble parts at different points in the structure concurrently using these algorithms. However, if for any stratum (and for subsequent strata in the 2.5-D SCS), the structure has disconnected regions and the distance between them is such that the quadrotors can concurrently assemble parts without interference, the assembly plan can be split into separate feasible plans for each disconnected region.

Here we state some theoretical properties of the algorithms and show the WFR and SCS algorithm can build any SCS.

Lemma 1 *At any step of the WFR algorithm, the partially-constructed stratum is connected.*

Proof Every placed square is a neighbor to an already placed square so the built region must always remain connected. □

Lemma 2 *The WFR algorithm can build any* 2-*D connected region.*

Proof Since the region is connected and finite, any unbuilt square is always n edges away (in the graph representation of the squares in a stratum) from an already built square and n is finite. So the unbuilt square will be built after n or less loops in Algorithm 1. This is true for all unbuilt squares.

We must also guarantee that all squares can be built using the four available assembly primitives, P1–P4 in Fig. 4. See Appendix. □

Algorithm 2 Special Cubic Structure (SCS) Construction Algorithm

```
1: build first 2-D stratum with WFR algorithm
2: while not finished do
3:    place columns required for next 2-D stratum
4:    build all groups of connected regions in next 2-D stratum with WFR algorithm
```

Theorem 1 *The SCS construction algorithm can realize any SCS.*

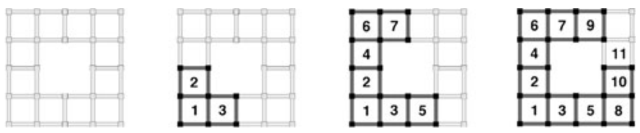
Proof All columns can be placed with assembly mode 1 and all SCS strata can be build using the WFR algorithm. □

2.4 Directional raster scan algorithm

In Sect. 2.3, we described the SCS algorithm for building a special class of cubic structures that are devoid of holes. For this class of structures, the Wavefront Raster algorithm generates a plan for building 2-D sections that are connected together to make a 3-D structures. Recall that the no-hole constraint arises from our desire to complete each square in a 2-D section, before proceeding to the next square. By completing each square, we reduce the number of cantilevered single beams, which may be disturbed by the downwash of the quadrotor or other disturbances. In structures with holes, the WFR algorithm eventually reaches the situation depicted in Fig. 6(d). Since each square adjacent to the square 11 is completed already, it is impossible to insert the two beams needed to complete the structure.

Relaxing the ‘completing the square’ constraint allows us to build structures with holes. This is done by avoiding deadlocks that occur when there is an empty slot for an unassembled beam between two assembled nodes. Thus any algorithm for automated assembly would have to avoid this deadlock condition. In this section, we present the Directional Raster Scan (DRS) algorithm (Algorithm 3) which does exactly this. As before the DRS algorithm only constructs 2-D sections, and must be used in conjunction with the SCS Algorithm (Algorithm 2) in which the WFR algorithm is replaced by the DRS algorithm.

The DRS algorithm is a two part algorithm, which builds modules that are aligned along the x-axis followed by beams/modules aligned with the y-axis. We will refer to modules aligned with the x-axis as East/West (EW)



(a) Stage 1 (b) Stage 2 (c) Stage 3 (d) Stage 4

Fig. 6 Snapshots of the WFR algorithm (Algorithm 1) failing to complete a simple holed structure. In order to complete the structure, a beam must be placed between two already placed nodes, which is not physically possible. The *number* indicates the order in which a particular square is added to the structure

and beams/modules aligned with the y-axis as North/South (NS). The DRS algorithm proceeds by completing EW rows using the East/West algorithm (Algorithm 4). When there are no free nodes on the current row, to which a module can be attached, the NS algorithm places all the bottom NS beams/modules of the lowest EW row with NS beams that need to be placed if any. If there are no bottom NS beams/modules, North/South algorithm builds the upper beams/modules on that row. The main constraint of the NS algorithm is that once a continguous row contains a node, no other node can be introduced. We define a continguous row as a row in the blueprint, which can be traversed without crossing gaps. Thus if there is no node in the subsequent continguous row, a NS module is placed; otherwise, a NS beam is placed. From the one node on the subsequent row, the EW algorithm attaches EW modules to this node expanding this row to its extents.

Figure 8 illustrates the DRS algorithm by building the same example, which the WFR algorithm failed to complete. After the initial module placement shown in Fig. 8(b), there are no free nodes on that row to attach another module. This situation results in Algorithm 5 placing all the NS beams/modules that are possible (Fig. 8(c)), which makes a node on the previous row available. From Stage 4 to 9 (Figs. 8(d)–8(i)), the EW algorithm will always complete the row before advancing to Algorithm 5. In fact, this is the case for situations after the first initial part placement. From Fig. 8(i) it is simple to see that completing the hole closure row results in no complications. Depicted in Fig. 9, a more complicated structure illustrates another behavior, which the DRS algorithm demonstrates. Assembly of the structure proceeds similarly to Fig. 8 up to Stage 5 (Fig. 9(d)). The EW algorithm completes the row albeit longer than the previous example. Since there are available beams attached to the bottom of this row, the NS algorithm first completes these NS beams/modules and the subsequent row (Figs. 9(f)–9(g)) before proceeding as normal.

2.4.1 *Algorithm properties*

Like the WFR algorithm, the DRS algorithm also has several noteworthy properties. First, as designed, this algorithm can

Algorithm 3 DRS Algorithm (*RIDX* is the current row being built, and *CIDX* is a list of beams in *RIDX*, which needs to be built.)

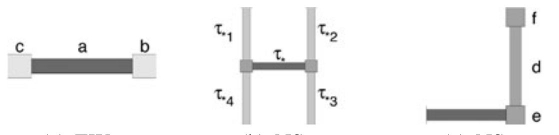
```
1: RIDX ← Row with southmost EW beams
2: CIDX ← {Westmost EW beam on row RIDX}
3: while not finished do
4:     EAST/WEST Algorithm
5:     NORTH/SOUTH Algorithm
```

Algorithm 4 EAST/WEST Algorithm

```
Require: CIDX, RIDX
1: while CIDX is not empty do {Refer to Fig. 7(a)}
2:     $a$ ← Westmost beam in the list CIDX
3:     if c is occupied then
4:         Build module $ab$ {EW Rule 1}
5:     else
6:         Build module $ac$ {EW Rule 2}
7:     CIDX ← CIDX \ a
8:     CIDX ← CIDX ∪ {Beams, which need to be built,
       with adjacent placed node in row RIDX}
```

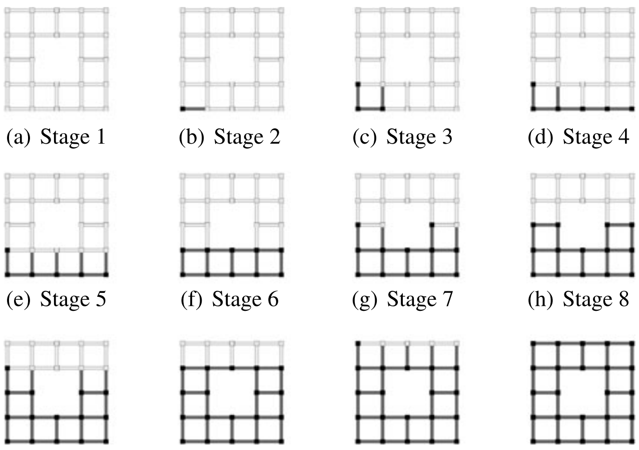
Algorithm 5 NORTH/SOUTH Algorithm (*TOPBEAMS*, *BOTBEAMS*, and *BEAMS* are lists of beams, which will be built.)

```
1: TOPBEAMS ← {}
2: BOTBEAMS ← {}
3: IDX ← {All EW beams, which have been placed}
4: for all $\tau_*$ in IDX do {Refer to Fig. 7(b)}
5:     TOPBEAMS ← TOPBEAMS ∪ {All NS beams
       ($\tau_{*1}$,$\tau_{*2}$) that are not placed but must be placed}
6:     BOTBEAMS ← BOTBEAMS ∪ {All NS beams
       ($\tau_{*3}$,$\tau_{*4}$) that are not placed but must be placed}
7: if BOTBEAMS is not empty then
8:     BOT ← Row containing southmost beam in list BOT-
       BEAMS
9:     BEAMS ← {Beams in list BOTBEAMS on row
       BOT}
10: else
11:     TOP ← Row containing southmost beam in list TOP-
        BEAMS
12:     BEAMS ← {Beams in list TOPBEAMS on row
        TOP}
13: for all $d$ in BEAMS do {Refer to Fig. 7(c)}
14:     if $e$ is not occupied then
15:         Build module $de$ {NS Rule 1}
16:     else if row with $f$ does not contain a node then
17:         Build module $df$ {NS Rule 2}
18:     else
19:         Build beam $d$ {NS Rule 3}
20: CIDX ← {All unbuilt EW beams with an adjacent
    placed node}
21: RIDX ← Row containing southmost beam in CIDX
22: CIDX ← {Beams in the list CIDX in row RIDX}
```



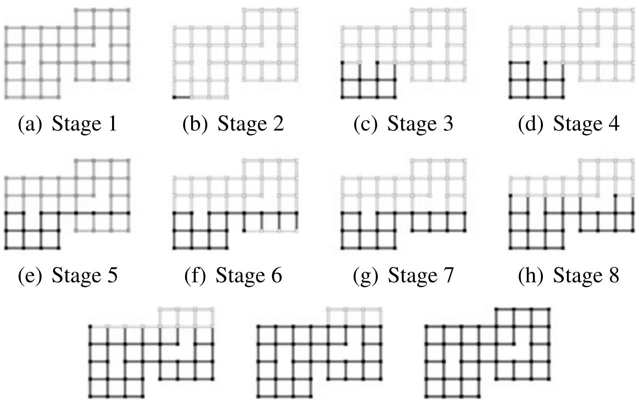
(a) EW (b) NS (c) NS

Fig. 7 Figures (**a**) and (**c**) show the possible arrangements of nodes and beams attached to a beam T_*, which is being currently placed. These figures are used with the rules of the North/South and East/West algorithms. Figure (**b**) illustrates the NS beams, which may be attached to a EW beam T_*



(a) Stage 1 (b) Stage 2 (c) Stage 3 (d) Stage 4 (e) Stage 5 (f) Stage 6 (g) Stage 7 (h) Stage 8 (i) Stage 9 (j) Stage 10 (k) Stage 11 (l) Stage 12

Fig. 8 Snapshots of the same structure, which the WFR algorithm failed to complete in Fig. 6. The *darker parts* represent parts that have already been placed while the *lighter parts* are those which will be placed

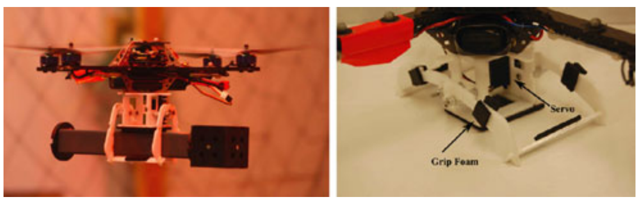


(a) Stage 1 (b) Stage 2 (c) Stage 3 (d) Stage 4 (e) Stage 5 (f) Stage 6 (g) Stage 7 (h) Stage 8 (i) Stage 9 (j) Stage 10 (k) Stage 11

Fig. 9 Snapshots of a more complex structure during various stages of its assembly using the DRS Algorithm. The *darker parts* represent parts that have already been placed while the *lighter parts* are those which will be placed

construct structures with holes in any 2-D stratum. This feature comes at the additional risk of accidental failures due to a fallen part. Unlike the WFR algorithm, we note that there is a limited ability to parallelize the assembly task. During the placement of the NS beams/modules as described by the NS algorithm, the order of beams/modules placement is arbitrary as long as the node restriction is maintained. Thus we can place beams/modules simultaneously provided that their placement does not causes the quadrotors to collide. However, the DRS algorithm also suffers from increased loss of structural integrity. In the WFR algorithm there is at most one cantilevered member. This member is susceptible to falling due to the downwash of the quadrotor. Since the DRS algorithm allows multiple cantilevered members, the likelihood of one of these members falling is greatly increased.

Finally, like the WFR algorithm we can provide theoretical properties and guarantees of the DRS algorithm. We use a similar approach as that taken in Sect. 2.3.1. First, we assume that the structure is connected throughout construction because parts are only attached to previously placed parts. Since the NS algorithm (Algorithm 5) precludes the assembly of multiple independent nodes on the same contiguous row, the DRS algorithm will not have any inconsistencies (instances when a part cannot be placed as described in Sect. 2.2). This also applies to loop closure of holed structures. Since the structure is connected and the completion of a row or column will eventually make every row or column accessible, the structure will be completed in finite time.



(a) Quadrotor with Part (b) Gripper

Fig. 10 (**a**) A Hummingbird quadrotor carrying a module. (**b**) A single degree of freedom gripper made of acrylic actuated by a servo motor with a layer of foam to facilitate grasping

3 Experimental infrastructure

3.1 Robots

We use the Hummingbird quadrotors sold by Ascending Technologies, GmbH[1] for experimental validation of the basic concepts. The quadrotor is approximately 55 cm in diameter, weighs approximately 500 g including battery, while providing approximately 20 minutes of operation with no payload. The maximum payload is around 500 grams.

Each robot is equipped with a gripper (see Fig. 10(b)) specially designed for the parts used for the SCSs. The

[1] Ascending Technologies, GmbH. http://www.asctec.de.

single-degree of freedom gripper consists of a pair of fingers driven by a simple slot mechanism powered by a Hitec HS-82MG servo, which has 2.8 kg-cm of torque and a mass of 19 g. The gripper is fabricated from acrylic and a layer of foam tape adds to the coefficient of friction for a more stable grasp.

3.2 Parts and bins

Stated in Sect. 2.1, the basic units of construction in this work are nodes and members. Each face of a node has four circular slots with complementary protrusions at the two ends of each member to provide features for assembly. Magnets are embedded at the center of each face to allow for a snap fit connection. The key differences from Galloway et al. (2010) are a reduction in the number of magnets and the mass of the parts. For transport by quadrotors, each node has a mass of 60 g and each member has a mass of 119 g so that the largest payload (a module) has a mass of 179 g. For robust assembly, the central magnet configuration reduces the likelihood of an incorrect stable part connection typical of the four magnet configuration.

Parts are stored in bins before assembly as shown in Fig. 2(b). Columns are stored in a bin consisting of a horizontal plate designed with two rows of holes that accommodate a face of the column. In the center of each set of holes is a small magnet, which is strong enough to support the columns vertically in the downwash of the quadrotors but is weak enough that the quadrotor can easily lift the column from the bin. Horizontal members (or beams) and modules are stored with their longitudinal axis in a horizontal position in a different bin with evenly spaced notches. The notches are spaced far enough apart so that the gripper when fully opened will not interfere with another part. Further, they are elevated from the surface such that when a quadrotor lands upon the part, the gripper can close around the member.

3.3 State estimation

In our work we rely on the VICON motion tracking system[2] for state estimation for the aerial vehicles, and for estimating the position and orientation of the part bins and base of the desired SCS to be built. The VICON system provides position feedback at 150 Hz with marker position accuracy on the order of a millimeter. The workspace of the tracking system is 6.7 m × 4.4 m × 4.0 m. Since it is impractical to place VICON markers on every part, each bin is designed to be a pallet and parts are stacked so that the position and orientation of a part with respect to the bin is known. While quadrotors use state feedback for assembling individual parts, there is no direct estimate of the state of the SCS. Since the initial position and orientation of the base of the SCS are known, the quadrotors keep track of the number and types of parts that have been assembled and are able to infer the state of the SCS at any point.

[2]Vicon Motion Systems. http://www.vicon.com.

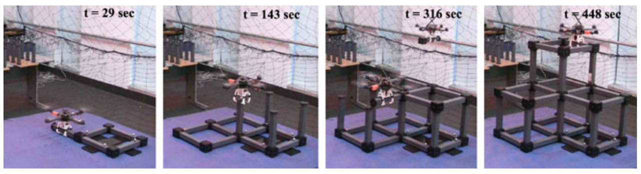


Fig. 11 Intermediate snapshots of a pyramid-like SCS being built by three quadrotors

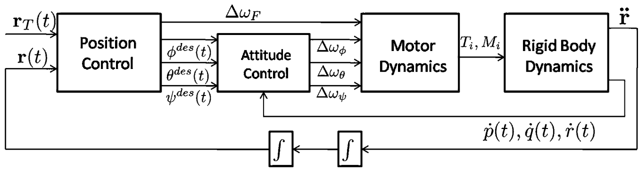


Fig. 12 Control Loops for position and attitude control

3.4 Control for robotic assembly

This section will briefly describe the two levels of the control for each quadrotor. Low level controllers described in Sect. 3.4.1 are used to execute three maneuvers. First, the quadrotor can hover at any specified position. Second, the quadrotor can execute a specified trajectory between any two desired points. Third, the quadrotor can apply open-loop yaw moments to test successful assembly of a part.

At a higher level, multiple quadrotors are coordinated using a finite state automaton to perform the assembly of a specified SCS efficiently and safely (Sect. 3.4.2) like in Fig. 11.

3.4.1 Quadrotor control

The quadrotor controller is illustrated in Fig. 12. Since the quadrotors operate at near hover conditions, we use controllers derived from the linearized equations of motion defined in Michael et al. (2010) where the roll and pitch angles, ϕ and θ, are proportional to accelerations in x and y. An inner loop controls the attitude of the robot similar to the approach used in other work (Bouabdallah 2007; Lupashin et al. 2010; Hoffmann et al. 2011; Michael et al. 2010). An outer position control loop prescribes the desired roll and pitch angles required to achieve the desired accelerations.

Let $\mathbf{r}_T(t)$ and $\psi_T(t)$ be the trajectory and yaw angle we are trying to track. The command accelerations in the ith direction, $\ddot{r}_i^{des}$, are calculated from PID feedback of the position error, $r_{i,T} - r_i$, using the equation:

$$\left(\ddot{r}_{i,T} - \ddot{r}_i^{des}\right) + k_{d,i}(\dot{r}_{i,T} - \dot{r}_i) + k_{p,i}(r_{i,T} - r_i) + k_{i,i}\int (r_{i,T} - r_i) = 0. \tag{1}$$

Note that the desired velocities and accelerations are given by $\dot{r}_{i,T} = \ddot{r}_{i,T} = 0$ for hover. Here the integral control terms,which result in the integral force component $F_{s,i}$, constantly adapt to the changing mass and center of mass of the system due to the changing payload. The desired roll and pitch angles are then calculated from the first two components of the desired acceleration while $\psi^{des} = \psi_T$ is specified for each task.

$$F_{s,i} = \left(A\left(k_{i,i}\int (r_{i,T} - r_i)\right) + B\right)^2$$

$$A = 0.0198\sqrt{\frac{\text{kg}\cdot\text{s}^2}{\text{m}}} \qquad B = 2.2676\sqrt{\text{N}}. \tag{2}$$

The attitude control block generates motor speed differentials $(\Delta\omega_\theta, \Delta\omega_\phi, \Delta\omega_\psi)$ according to PD control on the Euler angles and the angular velocities. The fourth motor speed differential, $\Delta\omega_F$, is derived from the desired acceleration in the z-direction. The four desired motor speeds, ω_i^{des}, are calculated from four rotor speed differentials and the nominal rotor speed required to hover, ω_h, through a constant linear transformation:

$$\begin{bmatrix}\omega_1^{des}\\ \omega_2^{des}\\ \omega_3^{des}\\ \omega_4^{des}\end{bmatrix} = \begin{bmatrix}1 & 0 & -1 & 1\\ 1 & 1 & 0 & -1\\ 1 & 0 & 1 & 1\\ 1 & -1 & 0 & -1\end{bmatrix}\begin{bmatrix}\omega_h + \Delta\omega_F\\ \Delta\omega_\phi\\ \Delta\omega_\theta\\ \Delta\omega_\psi\end{bmatrix}. \tag{3}$$

As in Michael et al. (2010), the high-level position control loop runs on a control computer that receives the quadrotor pose estimates from VICON. Interprocess communication on the control computer is handled by ROS (Quigley et al. 2009) and a ROS-MATLAB bridge.[3] The control computer sends inputs to the ARM7 processor on the quadrotors via ZIGBEE at a fixed rate of 100 Hz, which runs the low-level attitude control loop and computes the desired motor speeds. With the VICON estimates and quadrotor controller, we can achieve hover performance of ±1.5 cm in xy-plane and ±0.5 along the z-axis.

3.4.2 *Finite state automaton*

In this section we will describe the Finite State Automaton that coordinates the concurrent action of multiple quadrotors to enable multiple quadrotor experiments.

[3]ROS-Matlab Bridge. http://github.com/nmichael/ipc-bridge.

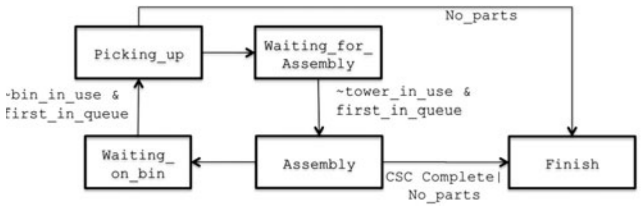


Fig. 13 The finite state automaton for picking up and assembling parts using multiple quadrotors

States in the FSA We use a FSA with five states as shown in Fig. 13. We require that only one quadrotor is retrieving parts from the part bins and that only one quadrotor is assembling parts to the SCS. `Waiting_on_Bin` and `Waiting_for_Assembly` have FIFO (First In, First Out) queues, where the quadrotors hover in place until the part bins or the SCS become available. At the conclusion of the experiment, each quadrotor transitions to the `Finish` state.

To avoid collisions between the quadrotors, we design the layout for assembly accordingly. The part bins, the SCS, the hover positions for `Waiting_on_Bin` and `Waiting_for_Assembly` are located around the perimeter of a loop such that no two paths taken by the quadrotors are close to each other at any given time. Furthermore, we add delays between state transitions to ensure that a quadrotor serving the part bins or assembling the structure has sufficient time to leave the area before another quadrotor enters that same area.

Picking up parts Columns are stored vertically in one bin while horizontal members and modules are stored horizontally in a different bin. To pick up columns, the quadrotor approaches and hovers in place above the specified column. It subsequently descends to a height such that when the grippers are closed, the face of the column is supported by the gripper from below. A similar procedure is used for horizontal members and modules. These parts are grasped by having the fingers close around the longitudinal axis. The quadrotor descends slowly to a point above the specified part and cuts its thrust while controlling to a zero pitch and roll angle to 'land' on the member. By landing on the member and grasping, we can ensure that the grippers will fully close around the part.

With part in gripper, the quadrotor then ascends and hovers in place for a specified time interval to the load. During this period the commanded thrust required to compensate for the load is used to determine if the quadrotor has grasped the part successfully. If the commanded thrust does not exceed the nominal thrust required for hover with the weight of the expected payload, the robot knows the grasp was unsuccessful and tries to grasp the part again. On repeated failures, the robot abandons that column and moves to the next available column.

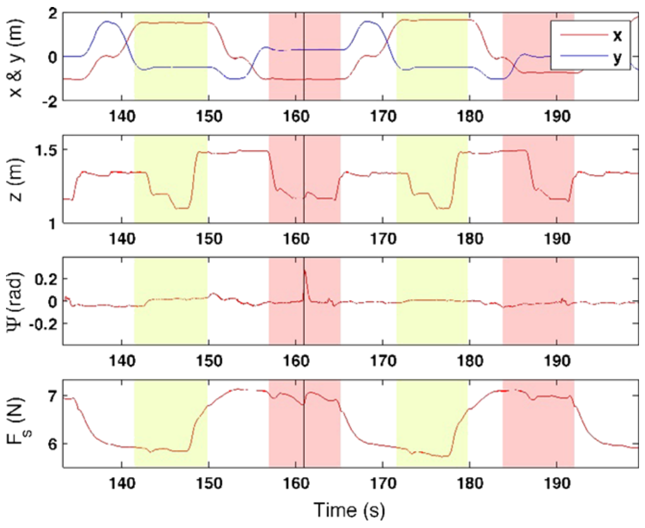


Fig. 14 Two cycles of pickup (*shaded yellow*) and placement (*shaded red*). The position history is shown for horizontal motion in the *top frame* (x in *red*, y in *blue*) and for vertical motion in the *second frame*. The yaw motion and the commanded thrust in the *third* and *fourth frame* are used to confirm pick up and assembly. The *vertical line* at $t = 160$ secs signifies an unsuccessful part placement (Color figure online)

This is illustrated in Fig. 14. The figure shows two cycles of picking up and placing parts. The two segments shaded yellow correspond to the `Picking_up` state. In both, the quadrotor quickly ramps up the required thrust to hover illustrated by the bottom plot. As expected, the required thrust to hover with a part is drastically larger than that required for no load conditions. The nominal thrust to hover as described above is extracted from these no load time periods and part pick up is confirmed by the rise in thrust.

Assembly The strategy for placing parts on the structure is illustrated in Fig. 15. For each assembly mode, the quadrotor first hovers in place at a point P_1 (the location of each part in Fig. 3). It then descends in a straight-line trajectory to P_2, the desired position of the part on the structure, along the direction shown in Fig. 3. Then the quadrotor hovers in place at P_2 for a fixed time until the magnets snap into place. We have found that the slight fluctuations about the position setpoint inherent in the position control and the influence of the magnet are normally enough to cause the part to snap into place.

Although this technique is quite robust, we use a simple error detection method to determine if the assembly was successful. After the fixed time of hovering in place at P_2, the quadrotor executes a specified open-loop yaw trajectory, $\psi^{des}(t)$. If the yaw angle, ψ, of quadrotor results in a large error, $\psi_{error} = |\psi_{des} - \psi| > \psi_{max}$, which is indicative of a unsuccessful assembly, the quadrotor ascends to P_1 to retry. If the error is small, it means the part has been fixed in place and the robot infers successful assembly and proceeds to the next task.

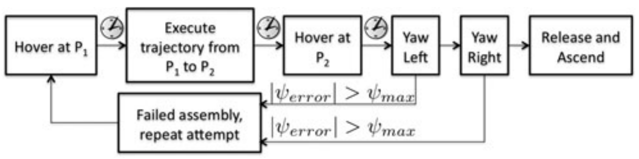


Fig. 15 Composition of (**a**) hover controller; (**b**) trajectory controller; and (**c**) yaw controller for assembling a part to a partially-completed SCS

This error recovery is in the segment shown in red in Fig. 14. In general, monitoring position histories $((x(t), y(t), z(t), \psi(t))$ does not give any information about successful assembly as can be seen from the top three panels. However, the open-loop yaw motion test helps the robot confirm assembly. For example, at $t = 160$ secs, the quadrotor recognizes an unsuccessful part placement using the yaw motion test described above. This error can readily be seen in the orientation subplot, where the yaw angle ψ spikes briefly. After detecting the error, the quadrotor ascends slightly in order to attempt the procedure again. If the part had been assembled successfully, it would have been impossible to execute this motion.

4 System evaluation

4.1 Empirical evaluation and assessment

Here we present experimental results for six trials, two for each of three representative SCSs, in Fig. 16 and Table 1. Snapshots of the assembly process for a pyramid-like SCS are shown in Fig. 11. Trials for a fourth structure (the castle) were investigated in simulation because of the limitation on the number of parts available and the battery life of the quadrotors.

Our simulation models the grasping, transport and assembly process. While attempts to assemble parts may fail in experiments, there are no failures in the assembly process in the simulation. The number of repeated assembly attempts (shown in Table 1) explains the discrepancy between the time to completion in simulation versus experimentation (*Note*: The time to execute a retry is not constant). Throughout all experimental trials there were two unsuccessful assemblies where the quadrotor determined it had assembled a part correctly but the part was actually pinned in an incorrect position.

4.2 Process variation and assembly tolerance

In robotic assembly, it is necessary to ensure that the process variation (errors in position and orientation of each part and the position of the partially assembled structure) is smaller

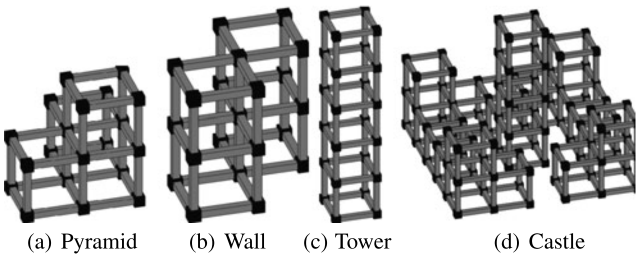


Fig. 16 Representative special cubic structures (SCS) tested in simulation and experiments

Table 1 Time to completion (in seconds) and success rate for SCS in Fig. 16 for two experimental trials and in simulation using the WFR algorithm

	Pyramid	Wall	Tower	Castle
Number of parts	32	34	40	192
Successful assemblies	32	33	40	
(Trial 1/2)	32	34	39	
Time (Trial 1/2)	449.6	486.6	588.2	
	450.7	486.2	587.3	
Column retries (Trial 1/2)	5/5	3/1	8/3	
Beam retries (Trial 1/2)	4/5	2/2	5/1	
Time (in simulation)	443.6	480.4	581.9	2642.0

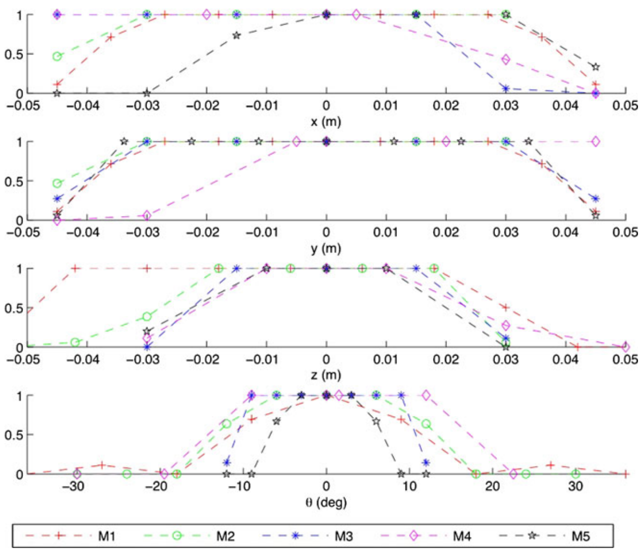


Fig. 17 Empirical evaluation of the success rates of placing parts for each assembly mode depicted in Fig. 3

Table 2 Empirical evaluation of >95 % success rate for picking up vertical and horizontal parts. For horizontal parts, the y-axis is along the length of the part. The orientation error of vertical parts is omitted because the part in its vertical configuration is rotation invariant. Additionally, the z error for horizontal parts is omitted because the procedure for picking up horizontal parts requires the quadrotor to cut its thrust and land on the part in question

Type	x (cm)	y (cm)	z (cm)	θ (deg)
Vertical	± 4.2	± 3.8	$[-2.5, 2.0]$	–
Horizontal	± 2.5	± 4.2	–	± 25

than the admissible tolerance required for pick up or for assembly. We empirically evaluated the allowed process variation for our assembly process with several trials for relative positioning errors of up to ±5 cm and orientation errors of up to 30°. Since the WFR and DRS algorithms both place parts in the same way, not necessarily in the same order, the analysis of process variations and admissible tolerance is equivalent. Figure 17 shows the probability of successful assembly of a part as a function of the error in the position and orientation of the structure during the five assembly modes depicted in Fig. 3. It can readily be seen that if the position errors are within ±3 cm in the horizontal plane, ±1 cm in the vertical plane, or ±5 deg in orientation, the probability of successful assembly is almost 1.0 in all assembly modes. Larger position and orientation error can be accommodated by repeated attempts as shown in Fig. 15.

A similar set of experiments showed that the system is even more robust in picking up parts as shown in Table 2. >95 % success rates are achieved for much larger position and orientation errors than for assembly. Note that since grasping columns is invariant to rotations about the yaw axis, that entry in the column is missing.

4.3 Effect of number of quadrotors

For the experimental setup described in Sect. 3.4, increasing the number of quadrotors quickly leads to diminishing returns using the WFR algorithm. Figure 18 shows the completion time and distribution of time spent in different states for a representative SCS consisting of two horizontal cubes requiring 16 part placements when using the WFR algorithm. For experiments with 3 or more quadrotors, the completion time remains a constant. This behavior is a direct consequence of our decision to have only one robot serve the part bins and one robot perform assembly on the structure at a given time.

Unlike the WFR algorithm, the DRS algorithm can be parallelize to small degree. Since the degree of parallelization depends on the structure in question, only qualitative remarks can be made. It is clear that for longer rows where a large number of NS beams/modules must be placed, that multiple quadrotors can place parts simultaneously. So for certain structures, the effect of multiple quadrotors will tend to have a larger impact for $n > 3$ quadrotors in comparison to the negligible gains shown in Fig. 18. This effect, however, will eventually dissipate when it becomes impossible for quadrotors to place parts due to other quadrotors in the immediate vicinity.

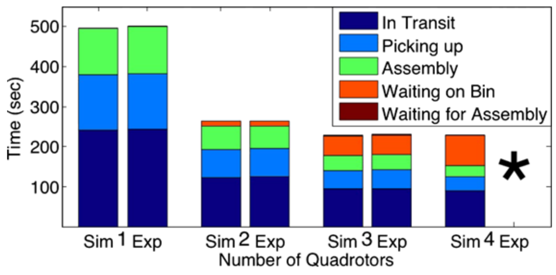


Fig. 18 Time spent on different states (Fig. 13) during the construction process using the WFR algorithm. '*' signifies an experiment, which was not conducted

5 Discussion

Main contribution of the paper The construction algorithms and control schemes outlined in Sects. 2.3, 2.4 and 3.4 have yielded a reliable system that can construct any Special Cubic Structure (SCS). This work is unique because it represents, to our knowledge, the first autonomous aerial robot construction system, and demonstrates a proof-of-concept system that can correctly and reliably construct a SCS for the construction of towers, scaffolds, trusses and commercial buildings. The extension to structures with holes is accomplished by replacing the WFR algorithm with the DRS algorithm. Of course, a number of issues will arise when we scale up to build real-world structures which we address below.

Localization without motion capture systems Although we depend extensively on VICON, we have demonstrated that this system can perform in less than ideal situations with errors in positioning and orienting. In Fig. 17 and Table 2, we showed the effect of position and orientation errors on part assembly and part pickup. Clearly our present system provides errors that are general smaller than the maximum allowable errors. With the current state of the art laser and vision based onboard localization systems boasting accuracies of better than 3 cm in position and 1° in yaw (Shen et al. 2011), it is possible to perform the individual tasks discussed here without the motion capture system.

Power The scale of the structures that we can currently build is mainly limited by battery life. While an unloaded quadrotor hovering in place can stay aloft for approximately 20 mins, a quadrotor loaded intermittently with the parts has a maximum flight time of 10 mins. To mitigate this effect, the batteries would need to be changed during the experiment. Certainly, this can be solved by integrating charging stations such as those described in Valenti et al. (2007) so that the longer assembly tasks can be completed, but at the cost of increased system complexity.

Parallelization of assembly tasks The current system implements assembly plans in a serial fashion. From Fig. 18, we can see that the benefit of multiple quadrotors quickly saturates. The algorithms presented here ensure that the SCS can be built using primitives that can be implemented with our subset of feasible assembly modes. In order to realize additional benefits with multiple quadrotors, we must solve two problems. First, we must develop new strategies of creating feasible assembly plans that can place parts without introducing intermediate structures that result in a deadlock condition (i.e., without resulting in states that preclude further assembly). Second, the system must allow several quadrotors to retrieve parts simultaneously from multiple supply bins and would need to incorporate more sophisticated path planners to resolve conflicts and to avoid collisions. Both these problems are future research directions.

Heterogeneity In this work we were limited with the constraints of an aerial robotic team. By integrating ground robots into this system, we can potentially build subassemblies on the ground and have quadrotors cooperatively lift and place them in a manner similar to the cooperative lifting described in Mellinger et al. (2010).

Part design and fasteners While the snap-fit parts improve the ease and reliability of assembly, real structures will require stronger joints. This may be done using smarter parts with built in fasteners that include anchor screws and toggle bolts that are automatically deployed on assembly or by attaching fasteners manually, as is currently done in automated building construction. Smarter parts can also be used to facilitate better fault detection by recognizing their neighbors and the state of the partially built structure and communicating with the robots during the assembly process.

Summary In conclusion, construction of structures with aerial robots represents an exciting area of research with many potential applications. This paper represents the first step in this direction with a proof-of-concept, provably correct approach to assembling Special Cubic Structures from parts with embedded magnets. As our discussion above illustrates, there are many ways to extend this work with a plethora of new research challenges for the robotics community.

Acknowledgements The authors would like to acknowledge Yash Mulgaonkar for assistance with the experiments and the fabrication of the parts and Professor Nathan Michael for help with the software interfaces and infrastructure.

Appendix: Proof of Lemma 2

In order to prove Lemma 2 we will consider constructing an individual square during the WFR algorithm in space 0

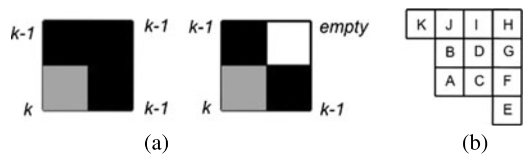


Fig. 19 (**a**) Inward Pointed L-shape (*left*) and Inward Pointless L-shape (*right*) and (**b**) square placement reference figure

during wave k as shown in Fig. 21(a). There are 8 immediate neighbors of space 0 so there are 256 possible situations that can occur. We will show that many of the 256 possible situations are impossible and that the only ones remaining can be constructed using the construction primitives P1–P4 shown in Fig. 4.

We introduce *Inward L-shapes* as illustrated in Fig. 19(a) and define them with the aid of Fig. 19(b). We define an *Inward Pointed L-shape* as a situation where square A was built during wave k and squares B, C, and D were built during wave $k-1$. We define an *Inward Pointless L-shape* as a situation where square A was built during wave k and squares B and C were built during wave $k-1$ and space D is to be left empty. Note that *Inward L-shapes* can be rotated versions (90°, 180°, and 270°) of the ones shown in Fig. 19(a). We first show that *Inward L-shapes* imply locally disconnected regions of squares during a previous construction wave.

Lemma 3 *For SCSs the Inward Pointless L-shape shown in Fig.* 19(a) *implies that spaces E, F, G, I, J, and K must contain a set of disconnected squares built during wave $k-2$.*

Proof No squares were built in E, F, G, I, J, or K prior to wave $k-2$ because then B or C would have been built before wave $k-1$. The fact that square B (C) was built during wave $k-1$ and A during wave k implies that one or more squares were built in spaces I, J, and K (E, F, and G) during wave $k-2$. Spaces G and I cannot both contain squares because this would imply a hole in the original structure at D. Therefore, E, F, G, I, J, and K must contain a set of disconnected squares built during wave $k-2$. □

Lemma 4 *For SCSs the Inward Pointed L-shape shown in Fig.* 19(a) *implies that for some wave number w there exists a set of squares of the shape of the region E–K that are disconnected and filled during wave w.*

Proof We can use the same logic as in the proof of Lemma 3 to show that if G and I are not both occupied then E–K are disconnected and $w = k-2$. If G and I are occupied and H is not then Lemma 3 tells us that during wave $k-3$ we must have a set of nodes shaped like E–K that are disconnected, meaning $w = k-3$. The only other possibility is that G, H, and I are all filled. In this situation we again have an *Inward

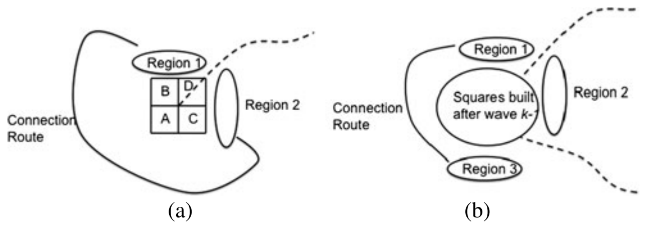


Fig. 20 (**a**) Region 1 and 2 cannot connect through some *curve* illustrated by the *dotted line*. (**b**) Region 2 is disconnected from both Region 1 and 3

Pointed L-shape*. So, if we repeat the logic we eventually result in a disconnected region shaped like E–K or a region shaped like G–I with all squares filled. However, we cannot continue to have three squares built at every wave because the WFR algorithm must start at wave 1 with a single built square. Therefore, for some wave number, w, we must have a region shaped like E–K that is disconnected. □

We now show that *Inward L-shapes* imply globally disconnected regions.

Lemma 5 *For SCSs the WFR algorithm cannot result in the Inward Pointless L-shape.*

Proof From Lemma 3 we know that spaces E, F, G, I, J, K contain two disconnected regions. As shown in Fig 20(a), we can draw a curve outward from square A that must separate the two regions of squares (Region 1 and 2). If they are completely disconnected this contradicts Lemma 1 so they must be connected in some way along the *Connection Route*. The Connection Route implies at least three rotated *Inward L-shapes* must exist. However, theses *Inward L-shapes* must also eventually result in locally disconnected regions like that shown in Fig. 20(a). Consider any one of the *Inward L-shapes*, it must eventually separate Regions 2 and 3 as shown in Fig. 20(b). Region 2 is then completely disconnected from region 1 and 3 which contradicts Lemma 1. Therefore, the WFR algorithm cannot result in an *Inward Pointless L-shape*. □

Lemma 6 *For SCSs the WFR algorithm cannot result in the Inward Pointed L-shape.*

Proof According to Lemma 4 we must eventually result in two disconnected regions in a shape like E–K. Then the same logic used to prove Lemma 5 can be used to prove this lemma. □

Now we consider building a square in space 0 during wave k as shown in Fig. 21(a) and the 256 possible situations that can occur. Here we note that any of the spaces 4, 5, 6, and 7 that are filled at the time of construction may

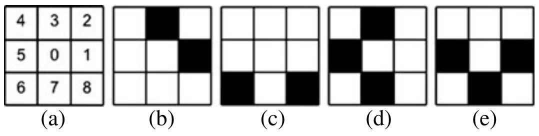


Fig. 21 Neighbor reference figure (**a**) and examples of situations that cannot occur (**b**–**e**)

have been filled during wave k or $k-1$ and any of them not filled at this point will never be filled. Blanks in spaces 1, 2, 3, or 8 may or may not be filled during wave k. Any squares built in spaces 1, 2, 3, or 8 must have been built during wave $k-1$.

Lemma 7 *At no point during the WFR algorithm for a SCS can a hole exist in the constructed structure.*

Proof Since the original structure does not contain holes, any hole in the structure must contain spaces that are to be filled. Any hole that is to be filled must contain a *Inward L-shape* in the top right corner. Since *Inward L-shapes* are impossible the WFR algorithm cannot result in a structure with holes. □

We can now eliminate all of the impossible situations and prove Lemma 2.

Proof of Lemma 2 We first eliminate any situation that contains squares at both spaces 1 and 3 like shown in Fig. 21(b). These squares must have been filled during wave $k-1$ meaning these situations are *Inward L-shapes* which are impossible according to Lemmas 5 and 6. This eliminates 64 situations.

We next consider situations where prior to construction we have (at least) two disconnected regions in spaces 1–8 like the example shown in Fig. 21(c). These may be connected outside of the spaces 1–8 but if this is true then building square 0 results in a hole in the structure. This contradicts Lemma 7 and we can dismiss 115 more situations as impossible.

The next situation to consider is one with squares built in spaces 3, 5, and 7 and not at 1, see Fig. 21(d) as an example. Square 3 must have been built during wave $k-1$. If Square 5 was built during wave $k-1$ then we would have an *Inward L-shape* so it must have been built during wave k. Square 7 must have been built during wave k or wave $k-1$. If it was built at wave k then one or more of the spaces directly below 6, 7, and 8 must have been filled during wave $k-1$. Either way square 3 must be connected to squares at the bottom. If they are connected around space 1 then the structure contains a hole, which is impossible. If they are connected around space 5 at least two *Inward L-shapes* are required which is impossible due to Lemmas 5 and 6. We can then dismiss 16 more situations.

The final situation to consider is one with squares in spaces 1, 5, and 7 and not in 3, see Fig. 21(e) as an example. The same logic used above can be used to show that blocks on the left must somehow be connected to blocks on the right which implies that this situation is impossible. We can then dismiss 16 more situations.

The remaining 45 situations can all be built using the constructions primitives P1–P4. □

References

Bernard, M., & Kondak, K. (2009). Generic slung load transportation system using small size helicopters. In *Proc. of the IEEE international conference on robotics and automation* (pp. 3258–3264).

Boothroyd, G., & Knight, W. (1993). Design for assembly. *IEEE Spectrum*, *30*(9), 53–55.

Bouabdallah, S. (2007). *Design and control of quadrotors with applications to autonomous flying*. PhD thesis, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland.

Galloway, K. C., Jois, R., & Yim, M. (2010). Factory floor: a robotically reconfigurable construction platform. In *Proc. of the IEEE international conference on robotics and automation* (pp. 2467–2472).

Groover, M. P. (2007). *Automation, production systems, and computer-integrated manufacturing* (3rd ed.). Upper Saddle River: Prentice Hall.

Henderson, J. F., Potjewyd, J., & Ireland, B. (1999). The dynamics of an airborne towed target system with active control. In *Proc. of the institution of mechanical engineers, Part G: Journal of aerospace engineering* (Vol. 213).

Hoffmann, G. M., Huang, H., Waslander, S. L., & Tomlin, C. J. (2011). Precision flight control for a multi-vehicle quadrotor helicopter testbed. September 2011.

Joo, H., Son, C., Kim, K., Kim, K., & Kim, J. (2007). A study on the advantages on high-rise building construction which the application of construction robots take. In *Proc. of the international conference on control, automation and systems (ICCAS)* (pp. 1933–1936).

Lindsey, Q. J., Mellinger, D., & Kumar, V. (2011). Construction of cubic structures with quadrotor teams. In *Proc. of robotics: science and systems conference (RSS)*.

Lupashin, S., Schollig, A., Sherback, M., & D'Andrea, R. (2010). A simple learning strategy for high-speed quadrocopter multi-flips. In *Proc. of the IEEE international conference on robotics and automation*, Anchorage, AK (pp. 1642–1648).

Matthey, L., Berman, S., & Kumar, V. (2009). Stochastic strategies for a swarm robotic assembly system. In *Proc. of the IEEE international conference on robotics and automation* (pp. 1953–1958).

Mellinger, D., Shomin, M., Michael, N., & Kumar, V. (2010). Cooperative grasping and transport using multiple quadrotors. In *Proc. of the international symposium on distributed autonomous systems*, Lusanne, Switzerland.

Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The GRASP multiple micro UAV testbed. *IEEE Robotics and Automation Magazine*, *17*(3), 56–65.

Michael, N., Fink, J., & Kumar, V. (2011). Cooperative manipulation and transportation with aerial robots. *Autonomous Robots*, *30*, 73–86.

Napp, N., & Klavins, E. (2010). Robust by composition: programs for multi-robot systems. In *Proc. of the IEEE international conference on robotics and automation* (pp. 2459–2466).

Petersen, K., Nagpal, R., & Werfel, J. (2011). Termes: an autonomous robotic system for three-dimensional collective construction. In *Proc. of robotics: science and systems conference (RSS)*.

Pounds, P., & Dollar, A. (2010). Hovering stability of helicopters with elastic constraints. In *Proc. of the ASME dynamic systems and control conference*.

Quigley, M., Gerkey, B. P., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., & Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*.

Sanderson, A. C., Homem de Mello, L., & Zhang, H. (1990). Assembly sequence planning. *AI Magazine*, *11*(1), 62–81.

Shen, S., Michael, N., & Kumar, V. (2011). Autonomous multi-floor indoor navigation with a computationally constrained MAV. In *Proc. of the IEEE international conference on robotics and automation*, Shanghai, China.

Valenti, M., Dale, D., How, J. P., de Farias, D. P., & Vian, J. (2007). Mission health management for 24/7 persistent surveillance operations. American Institute of Aeronautics and Astronautics.

Werfel, J., & Bar-yam, Y. (2006). Distributed construction by mobile robots with enhanced building blocks. In *Proc. of the IEEE international conference on robotics and automation*. New York: IEEE Press.

Werfel, J., Ingber, D., & Nagpal, R. (2007). Collective construction of environmentally-adaptive structures. In *Proc. of the IEEE international conference on intelligent robots and systems*.

Quentin Lindsey received his B.S. in Mechanical Engineering and in Electrical Engineering from Yale University in 2007. He is currently pursuing his PhD in Mechanical Engineering and Applied Mechanics at the University of Pennsylvania as a member of the GRASP Lab. His research focuses are in multi-robot collaborations for manipulation and exploration tasks.

Daniel Mellinger received his B.S. degree in mechanical engineering from North Carolina State University in 2007. He is currently a Ph.D. student at the University of Pennsylvania, researching problems concerning the dynamics and control of robotic systems, in particular quadrotors.

Vijay Kumar received his Ph.D. degree in mechanical engineering from Ohio State University in 1987. He is currently the UPS foundation professor and the deputy dean in the School of Engineering and Applied Science at the University of Pennsylvania, focusing on mechanical engineering and applied mechanics as well as computer and information science. He is a Fellow of the IEEE and American Society of Mechanical Engineers.