Programmable Self-Disassembly for Shape Formation in Large-Scale Robot Collectives

Melvin Gauci, Radhika Nagpal and Michael Rubenstein

Abstract We present a method for a large-scale robot collective to autonomously form a wide range of user-specified shapes. In contrast to most existing work, our method uses a subtractive approach rather than an additive one, and is the first such method to be demonstrated on robots that operate in continuous space. An initial dense, stationary configuration of robots distributively forms a coordinate system, and each robot decides if it is part of the desired shape. Non-shape robots then remove themselves from the configuration using a single external light source as a motion guide. The subtractive approach allows for a higher degree of motion parallelism than additive approaches; it is also tolerant of much lower-precision motion. Experiments with 725 Kilobot robots allow us to compare our method against an additive one that was previously evaluated on the same platform. The subtractive method leads to higher reliability and an order-of-magnitude improvement in shape formation speed.

1 Introduction and Related Work

In nature, groups of thousands to millions of units can self-assemble into complex structures, purely through local interactions. For example, cells self-assemble into complex organisms, colonies of ants self-assemble into bridges and bivouacs made out of their own bodies, and fish self-assemble into complex schooling patterns [1, 3]. In all of these cases, a group can dramatically change its interaction with its environment through collective shape formation. In the fields of collective and modular robotics, many groups have taken inspiration from these natural examples to create novel robotics systems that achieve higher functionality through

Melvin Gauci, Radhika Nagpal
Harvard University, Cambridge, MA, e-mail: {mgauci@g, rad@eecs}.harvard.edu

Michael Rubenstein
Northwestern University, Evanston, IL, e-mail: rubenstein@northwestern.edu

self-assembly and shape formation. For example, researchers have designed small mobile robots that are envisioned to work together in large groups, including assembling together to achieve complex tasks [8, 9, 11, 12]. Similarly, using inspiration from multicellular development, several groups have developed modular robots that can self-reconfigure into different morphologies for grasping or locomotion tasks [16]. The ultimate goal is to shrink the size of these systems to create "programmable matter", where large numbers of tiny robotic modules can rapidly self-assemble into user-specified tools such as a wrench or a key [2, 6].

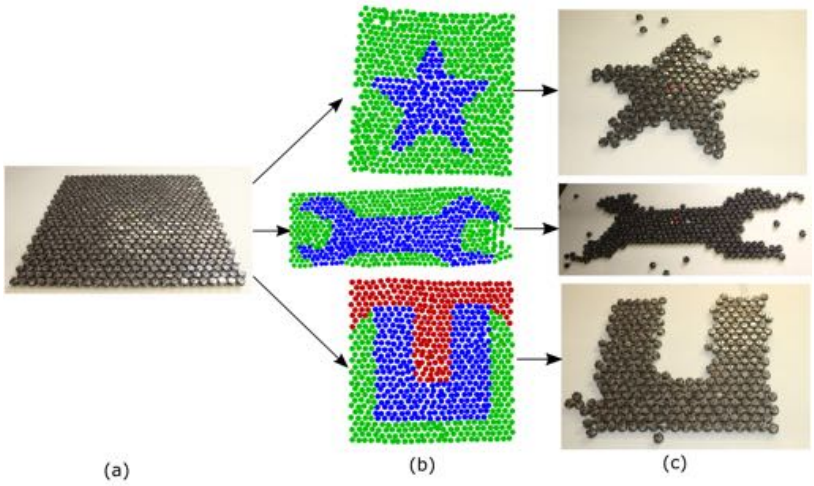

Fig. 1 Self-disassembly overview. (a): Robots start arranged in a grid. (b): A coordinate system is formed in a distributed fashion, and each robot decides if it is part of a user-specified shape (blue). If it is not, it also decides which motion type it should use to leave the configuration—phototaxis (red) or antiphototaxis (green); (c): Final state: non-shape robots have left the shape.

Self-assembling robot collectives and programmable matter systems pose a global-to-local algorithmic challenge: How does one program large numbers of agents to self-organize into a user-specified shape, when each agent can only interact locally with nearby agents at a scale much smaller than the global outcome? There is also the hardware challenge: How do such algorithms scale to large numbers of real robots, where individual errors can amplify into global errors?

There has been considerable progress on answering these algorithmic questions. Inspiration is often drawn from conventional engineering manufacturing processes, which can be broadly divided into two classes: additive and subtractive. In additive processes, material is added and fused to form the desired shape; examples include injection molding and 3D printing. In subtractive processes, material is removed from an initial bulk until only the desired shape remains; examples include lathing and milling. It is noteworthy that both additive and subtractive processes can also

be observed in nature, such as the growth of limbs (additive) and the removal of inter-digit webs in the hand (subtractive) [15].

In additive self-assembly methods, the shape starts with a seed module that defines the origin and orientation of a coordinate system. The remaining modules then localize relative to the seed and form the shape in layers. This approach is often referred to as "directed growth" [13] and has been implemented both in mobile collectives where the robots move freely in space [8, 11, 12], and in self-reconfigurable systems where the robots are constrained to move along docking sites [16, 17]. In some cases, instead of actively mobile agents, the system uses agents that move passively in the environment, based on some source of random motion such as the mixing of a fluid or a low friction surface [2, 9, 14]. Although many additive self-assembly algorithms have been explored, most have only been validated in simulation or in at most tens of hardware agents. One recent exception is a demonstration by our group on a 1024 robot collective, namely the Kilobot system [12]. Using cooperative error correction, this approach was able to robustly self-assemble large shapes with diameters of up to 45 times the radius of local interactions without human intervention. However, it has limited parallelism in robot motion, and requires this motion to be highly precise. These factors lead to long shape formation times.

The subtractive self-assembly approach (hereafter called *self-disassembly*) has received considerably less attention so far, but a few examples do exist. In this approach, the system starts with the agents in a large, dense, stationary group. A seed module then initiates the formation of a coordinate system, and finally, non-shape agents remove themselves from the system. In the Pebbles system [6], modules are arranged in a 2D rectangular lattice, and non-shape modules release themselves and move away by exploiting surface vibrations. In the Miche system [7], larger modules are arranged into a 3D rectangular lattice, using magnetic connections to maintain the structure. The removal of non-shape modules is achieved by exploiting gravity. Both the Pebbles and the Miche systems attempt to replace active precision mobility with external actuation, in order to be able to shrink the size of modules and bring the system closer to the vision of programmable materials.

The subtractive approach has several potential advantages over the additive approach, including a high level of motion parallelism, and a low level of required motion precision, both of which become more attractive as the scale of the collective increases. The self-disassembly examples discussed above were both implemented on modular robots that can be arranged into a regular lattice structure. In this case, coordinates can be easily assigned since the modules can exploit directional communication [6, 13]. However, requiring modules to align in perfect lattices makes the hardware design more complex and any manufacturing imperfections can negatively affect the perceived locations of modules. Moreover, these systems have so far only been tested on limited numbers of modules (≤ 30).

In this paper, we present an approach to programmable self-disassembly that extends the conceptual ideas from the Miche and Pebbles systems to a fully-autonomous mobile robot collective that operates in free space, and uses only local sensing and interactions. The robots distributively form a continuous coordinate system through distance sensing and multilateration, allowing for a larger tolerance

to the localization errors of individual robots [10, 12]. The robots then use a decentralized consensus algorithm to transition from the coordinate system formation phase to the self-disassembly phase. Finally, the non-shape robots autonomously remove themselves from the shape without distorting it, using a single external light source as a motion guide. We present theoretical results to show which classes of shapes can be formed by our approach, and how the shape class determines the self-disassembly procedure. We then validate our approach on the Kilobot platform, using 725 robots, and present experimental results demonstrating high reliability and accuracy despite the noise and failures present in large-scale collectives. We also present a direct comparison of additive and subtractive algorithms, as experimentally tested on the same hardware platform, and show that while the additive approach is able to form larger shapes for a given number of robots, the subtractive approach achieves much higher motion parallelism, as well as tolerance to simpler and noisier motion. This results in a significant improvement in shape formation speed.

2 Self-Disassembly Algorithm

2.1 Robot Model

Here we describe the model for a Kilobot robot; note, however, that its feature set matches closely with those of many other robots designed for large-scale collective behaviors. More details on the Kilobot platform can be found in [12]. An individual Kilobot has the following capabilities: external light sensing, computation (via a microcontroller), and non-holonomic locomotion using vibration motors. Each robot can also communicate with nearby neighbors within a radius of 3 body lengths (at most around 36 robots) via shared channel wireless broadcast; the group thus acts as a large multi-hop wireless network. Furthermore, each robot can measure its distance to its neighbors based on the communication signal strength, with a resolution of 1 mm. However, it cannot measure the bearings or headings of its neighbors.

The collective is inherently decentralized, multi-hop, and asynchronous. In addition, all aspects experience noise and manufacturing variation. The light sensor has a limited resolution and accuracy. Open-loop robot motion drifts quickly with time and the robots do not have internal odometry capabilities. Messages are communicated via a shared wireless channel which is asynchronous and experiences message loss. Distance measurements based on wireless signal strength are inherently noisy. Even after calibration, there are significant variations between robots in all aspects.

2.2 Algorithm and Implementation

The goal of the algorithm is to form a user-specified shape, starting from a configuration where all the robots are in a tightly packed group. We assume that the size of the desired final shape fits within the starting configuration. Three pre-localized *seed* robots are placed within the initial configuration, representing the origin and orientation of the coordinate system. All the robots in the configuration are given an identical program, which includes a representation of the shape to be formed and the location of a light source (with respect to the seed robots) that will serve to guide the motion of non-shape robots. The algorithm proceeds in three steps: (i) Distributed Coordinate System Formation - each robot localizes itself within the collective (with respect to a seed) and decides whether or not it is part of the desired shape; (ii) Transition using a Consensus Algorithm: the robots collectively determine when every robot has localized in order to transition to the next step; (iii) Self-Disassembly: robots that are in the desired shape remain stationary while non-shape robots move away from the shape.

Distributed Coordinate System Formation

Since Kilobots can measure distance to and communicate with neighbors, a robot can localize itself with respect to 3 or more other localized neighbors using multilateration. If robot i has n localized neighbors with distances $d_1,\ldots,d_n$ away from it and positions $\mathbf{p}_1,\ldots,\mathbf{p}_n$, then robot i computes its best-guess position as:

$$\mathbf{p}_i^* = \arg\min_{\mathbf{p}_i} \sum_{j=1}^{n} \frac{\left|d_j - \|\mathbf{p}_i - \mathbf{p}_j\|\right|}{d_j}, \tag{1}$$

where $|\cdot|$ denotes the absolute value and $\|\cdot\|$ denotes the Euclidean norm. The d_j in the denominator serves to implement inverse variance weighting, giving less importance to larger measured distances, which tend to be less accurate. The robots localize in a 'wave' that propagates outward from the 3 pre-localized seed robots.

Although distributed coordinate system formation is a simple process at an abstracted level, it is well-known that in practice, it can be highly sensitive to initial conditions and several types of noise [10]. As such, at large scales, it requires significant additional constraints to operate reliably. As shown in the experimental section, we have developed a highly-robust distributed coordinate system formation algorithm by using the following measures.

(i) In order to account for asymmetry in distance sensing, robots exchange measured distances with each other for two-way averaging. Let d_{ij} be the distance to robot j as measured by robot i, and vice-versa for d_{ji}. Then both robots estimate the distance between them as being $\frac{1}{2}(d_{ij}+d_{ji})$. Robot i only takes robot j into account for multilateration purposes (i.e. Eq. 1) if: (a) both d_{ij} and d_{ji} are available, (b) $\frac{1}{2}(d_{ij}+d_{ji}) \leq 3$ body lengths, and (c) $\left|d_{ij}-d_{ji}\right| \leq 1$ body length. (ii) A robot only considers localizing after it has at least one set of 3 neighbors that meets the following criteria: (a) the minimum angle of the triangle formed by the positions of

these 3 neighbors is larger than 35°, and (b) 2 of these neighbors are within 1.25 body lengths and the other is within 2.5 body lengths. The first criterion avoids localizing with respect to almost-collinear neighbors, in which case the best-guess position might have a flip ambiguity. The second criterion ensures that the estimated distances of the robots making up this triangle are reasonably accurate. After condition (ii) has been met, a robot waits up to 30 seconds to check if it can acquire more neighbors that satisfy condition (i). Each time a new such neighbor is acquired, this timer is reset and the waiting starts again. This ensures that each robot uses as many neighbors as possible for multilateration, increasing the amount of averaging in the process, and thus improving consistency.

Once localized, a robot determines if it is part of the shape. The shape is specified in the robots' program as a polygon with an arbitrary number of vertices (≥ 3), and a simple point-in-polygon algorithm is used to decide if the robot's position lies inside its perimeter. At this point, if a robot is not part of the shape, it also decides based on geometric considerations which type of motion it will execute when it is its turn to start moving (this will be explained further in Sec. 2.3).

Transition using a Consensus Algorithm

Once all the robots have localized, the collective must autonomously transition to the next phase where the non-shape robots remove themselves from the configuration. This is done using a simple gossip-based consensus algorithm, as follows. Each robot transmits a consensus value. For a non-localized robot, this value is always 0. A localized robot, on the other hand, determines its value by adding 1 to the minimum of its neighbors' values (all neighbors within the communication range are used for this algorithm). As long as there is at least one non-localized robot, its 0 value will hold down the values of all the other robots; a 'gradient' pattern will form across the group, with the maximum possible value of any robot being the maximum distance in the configuration (in terms of the communication range). When there are no longer robots transmitting a value of 0, all the robots start to count up. Once a robot's value exceeds a predefined threshold (chosen to be larger than the maximum possible hop count with an unlocalized robot still present), the robot decides that consensus has been reached. This algorithm is sensitive to cases where a single robot fails to localize permanently, due to not meeting one of the conditions described above. In our experiments, this happened rarely. Nevertheless, in the future, this phase of the algorithm can be improved by using a quorum algorithm in place of a consensus algorithm, which would 'only' require a high percentage (but not 100%) of the robots to have localized.

Self-Disassembly

To remove themselves from the shape, non-shape robots rely on three motion types: collision avoidance, phototaxis, and antiphototaxis. All moving robots perform collision avoidance if they are too close to their neighbors, but each robot will only perform one of phototaxis and antiphototaxis, depending on its initial position rela-

tive to the shape. After a robot has localized, its program casts a ray from its position to the light source; if this does not intersect with the shape, the robot decides on performing phototaxis; otherwise, it decides on performing antiphototaxis.

The self-disassembly process starts from the edge of the initial configuration, and proceed inwards towards the shape. In this manner, robots that are completely locked by other robots do not needlessly start moving, which could potentially distort the shape. Each robot only starts moving when it has had 4 or fewer neighbors (stationary or moving) within 1.25 body lengths for a predefined amount of time (30 s). After starting to move, robots execute the following behavior.

A robot does collision avoidance if it perceives a stationary neighbor with 2 body lengths (i.e. one that is either part of the shape, or is not but has not yet started moving). This is done to minimize the chances of moving robots colliding with the shape and distorting it, at the cost of a slower self-disassembly process. When a robot senses no stationary localized robots within 2 body lengths, it switches to phototaxis or antiphototaxis. A robot that is doing phototaxis or antiphototaxis only reverts to collision avoidance if it perceives a stationary neighbor within 1.5 (rather than 2) body lengths. This hysteresis prevents robots from continuously switching between behaviors. Note that, moving robots do not avoid each other, allowing them to move quickly and aggressively, even in the face of congestion.

The motion behaviors are implemented as follows. Collision avoidance starts with the robot turning, and looking at its distance to its closest stationary neighbor. The robot searches for a valley in the profile of this distance versus time, i.e., a transition from a negative to a positive rate of change. This indicates that the robot is facing away from its stationary neighbor, and at this point, it switches to straight motion. In phototaxis, the robot starts turning and looking at its light sensor reading. It searches for a valley in the intensity profile versus time, which indicates that it is facing towards the light source (the light sensor is on the back of the robot). At this point, the robot keeps rotating in its current direction for a small time period, then reverses its turning direction, and starts looking for a new valley. Due to the configuration of the robot's legs, this oscillatory turning causes a net motion towards the light. Antiphototaxis is implemented in the same way, but the robot now looks for peaks rather than valleys in the intensity versus time profile.

2.3 Achievable Shape Classes

In order to successfully form a desired shape, all non-shape robots must move away from the shape without disturbing it. Our current system relies on phototaxis and antiphototaxis, which have the advantage of being fast and highly tolerant of robot sensory and actuation noise. However, this imposes a constraint on the system: all (or in practice most) of the robots must be able to leave the shape using one of these two motion types. We now describe which shape types are solvable by self-disassembly under this constraint. We use polygon representations for the shapes, and divide polygons into 3 classes. An example from each class is provided in Fig. 2.

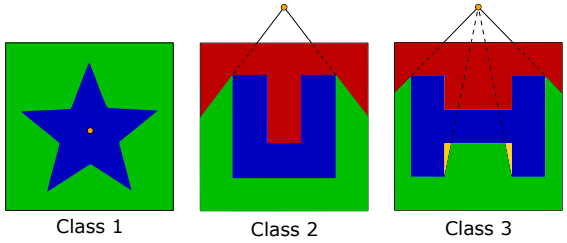

Fig. 2 Examples of shapes in the 3 classes (see the text for details). Orange dots indicate the light source location; blue indicates the shape; red and greed indicate regions from which phototaxis and antiphototaxis are possible, respectively; yellow indicates blocked regions.

- **Class 1**: *Solvable by antiphototaxis alone*: "Star-shaped polygons" is a well-known class of polygons that can be defined as follows: There is at least one point within the polygon such that any ray cast from this point intersects the polygon's perimeter exactly once. For our application this means that, if the light source is placed at such a point, any robot outside the shape can move away from it in a straight line (i.e., antiphototaxis) without encountering the shape. Note that convex polygons are a subclass of star-shaped polygons, having the additional property that *every* point inside the polygon satisfies the light placement condition. In general, a linear-time algorithm exists for determining which points inside a given polygon, if any, satisfy this condition; if none are found, the polygon is not Class 1 [5].
- **Class 2**: *Solvable by a combination of phototaxis and antiphototaxis*. We define this class of polygons according to the following property: There is at least one point outside the polygon such that every ray cast from this point either does not intersect the polygon's perimeter, or intersects it exactly twice. If a ray does not intersect the perimeter, both phototaxis and antiphototaxis are viable from every point on this ray; we prioritize phototaxis over antiphototaxis due to its better reliability on Kilbots. If a ray intersects the perimeter twice, then phototaxis is viable from points that lie on the segment between the light source and the first intersection, while antiphototaxis is viable from points beyond the second intersection. In the supplementary material [5], we supply a linear-time algorithm that determines the set of feasible light source positions, if any, for a Class 2 shape; if none exist, the polygon is not Class 2.
- **Class 3** *Not solvable by a combination of phototaxis and antiphototaxis alone*. A polygon is in Class 3 if it is not Class 1 or Class 2. For a Class 3 polygon, wherever the light source is located, there will always be regions in the environment from which a robot can do neither phototaxis nor antiphototaxis without encountering the shape. In general, self-disassembly into Class 3 polygons is not possible under our constraints. In practice, some robots might still manage to navigate away from blocked regions using collision avoidance, but this depends on the depth of the blocked regions. It is a promising avenue for future research to investigate how our method can be augmented for such shapes.

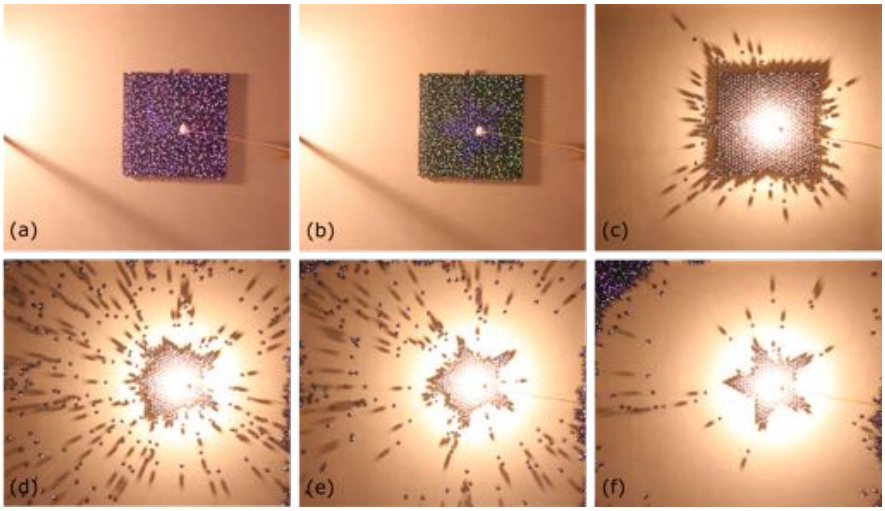

Fig. 3 Experimental self-disassembly of a starfish shape. *(a)* Initial configuration; only the seed robots are localized. *(b)* $t = 40$ min: All the robots are localized, and each robots has decided if it is part of the shape (blue) or not (green). *(c)* $t = 50$ min: The non-shape robots start leaving the shape. *(d)*, $t = 55$ min, *(e)* $t = 63$ min. *(f)* 90 min: Final state.

3 Experimental Validation

3.1 Setup and Protocol

In order to validate our algorithm, we performed 3 experiments with two shapes in Class 1 and on shape in Class 2. The Class 1 shapes constituted a 5-point starfish and a wrench, both with the light source placed in the center such that all robots could perform antiphototaxis. The Class 2 shape was a U shape, with the light source placed above the *U*'s cavity such that robots in the cavity could perform phototaxis and the rest could perform antiphototaxis. We used the experiments to look at the time efficiency and the accuracy at each stage of the algorithm.

The experimental protocol was as follows. The robots were initially arranged in an approximate hexagonal grid. For the starfish and the U shapes, this consisted of 29 rows by 25 columns (= 725 robots), which yields a roughly square arrangement since $29 \times \sqrt{3}/2 \approx 25$. For the wrench shape, the initial configuration consisted of 17 rows by 42 columns (= 714 robots), which yields a rectangular arrangement. In all cases, the seed robots were placed in the middle of the initial configuration.

All the robots were simultaneously programmed with the algorithm described previously. At the beginning of each experiment, the light source was placed in the appropriate location, and recording was started on an overhead camera. A run signal was then issued to all the robots, and the formation of the coordinate system was monitored by the operator based on the robots' LED colors. In order to collect data

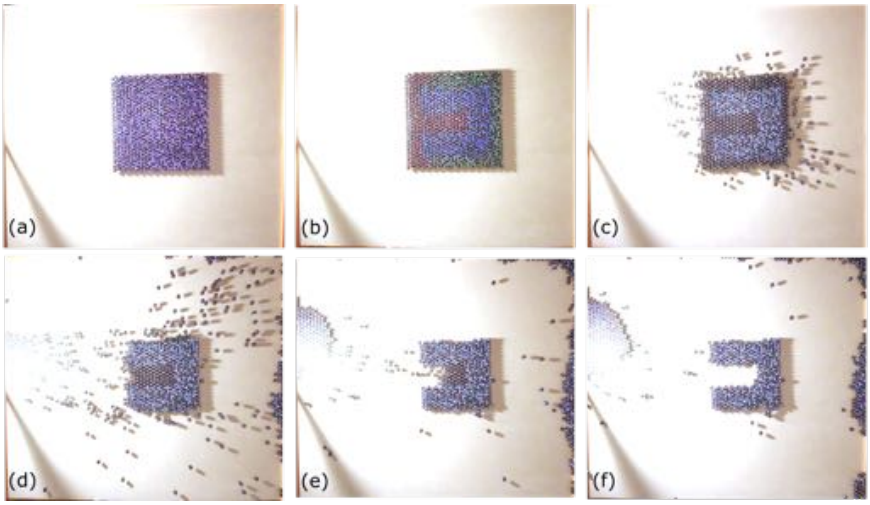

Fig. 4 Experimental self-disassembly of a U shape. *(a)* Initial configuration; only the seed robots are localized. *(b)* $t = 40$ min: All the robots are localized, and each robots has decided if it is part of the shape (blue), and if not, whether it should leave the shape using phototaxis (red) or antiphototaxis (green). *(c)* $t = 50$ min: The non-shape robots start leaving the shape. *(d)*, $t = 55$ min, *(e)* $t = 63$ min. *(f)* 90 min: Final state.

on how well the robots were estimating their own coordinates, we used an extra robot that was programmed to receive messages from localized robots and report their positions to a PC. This robot was waved over the group of robots when the coordinate system formation process had finished, and before the transition happened. During the self-disassembly phase, no manual intervention was made, including in cases of dead or stuck robots. The experiment was stopped after one hour had elapsed from the start of the self-disassembly phase.

3.2 Results and Discussion

Fig. 1 shows the (a) initial and (c) final states of the robots for both experiments, as well as (b) where each robot believes its coordinates to be, with colors indicating whether it believes to be part of the shape (blue), or should leave the shape via phototaxis (red) or antiphototaxis (green). Figs. 3 and 4 show snapshots of the self-disassembly process over time for the starfish shape and the U shape, respectively. The supplementary material [5] includes full-length overhead video recordings of the self-disassembly process for the starfish and the U shape experiments.

Qualitatively, it can be visually observed in Fig. 1 (b) that in all three cases, the coordinate system formation algorithm achieved a good accuracy in all three cases. A slight rotation of the coordinate system is present in the case of the starfish shape. Note, however, that this is a global rotation, and does not cause any significant warp

in the coordinate system; as such, the user-specified shape is not distorted. As a first quantitative measure of coordinate system accuracy, we look at the estimated distances between all pairs of robots that are adjacent in reality (i.e. the real distance is 1 body length $= 33\,\text{mm}$). We looked at the distribution statistics of the error (defined as $\text{distance}_{\text{estimated}} - \text{distance}_{\text{true}}$). In the starfish experiment, the median error was $+1.00\,\text{mm}$; the 25th and 75th percentile errors were $-1.74\,\text{mm}$ and $+4.00\,\text{mm}$, respectively; and the extreme errors were $-13.79\,\text{mm}$ and $+13.32\,\text{mm}$. In the wrench experiment, the median error was $+1.01\,\text{mm}$; the 25th and 75th percentile errors were $-1.94\,\text{mm}$ and $+4.05\,\text{mm}$, respectively; and the extreme errors were $-33.00\,\text{mm}$ and $+13.96\,\text{mm}$. In the U shape experiment, the median error was $+1.06\,\text{mm}$; the 25th and 75th percentile errors were $-0.98\,\text{mm}$ and $+3.68\,\text{mm}$, respectively; and the extreme errors were $-21.34\,\text{mm}$ and $+13.39\,\text{mm}$.

As a second quantitative measure of coordinate system accuracy, we calculated the error between the area of the estimated configuration (defined as the area enclosed by the convex hull of the estimated coordinates) and the area of the true configuration. The error is defined as $100 \times (\text{area}_{\text{estimated}} - \text{area}_{\text{true}})/\text{area}_{\text{true}}\%$. For the starfish, wrench, and U shape experiments, this error was $+0.62\%$, -0.81%, and $+2.37\%$, respectively.

As a third quantitative measure of coordinate system accuracy, we compared the error between the number of robots that considered themselves to be part of the shape according to the coordinate system and the theoretically-expected number (assuming a fully-packed hexagonal initial configuration). The error is defined as $100 \times (\text{number}_{\text{coordinate system}} - \text{number}_{\text{expected}})/\text{number}_{\text{expected}}\%$. In the starfish experiment, the actual and expected numbers were 164 and 181, respectively (error $= -9.39\%$). In the wrench experiment, the actual and expected numbers were 336 and 356, respectively (error $= -5.62\%$). In the U shape experiment, the actual and expected numbers were 284 and 298, respectively (error $= -4.70\%$).

These three quantitative measures of coordinate system accuracy together suggest that the robots are systematically overestimating the distances to their neighbors. Note that in the first measure, the median error is positive in all three cases; if there were no systematic error in the distance sensing, a median error of 0 mm would be expected. In the second measure, the estimated area is larger than the true are in two out of three cases. In the third measure, in all three cases, the number of robots that considered themselves to be in the shape according to the coordinate system is smaller than the expected number. This is because the estimated configuration is expanded, and therefore sparser.

All three runs of the coordinate system reached steady-state after around 40 min. This time mainly reflects the strict conditions imposed on when a robot can start to localize (see Sec. 2.2). We observed that relaxing some of these conditions would cause the coordinate system to form faster, but at the cost of reduced accuracy and reliability. In the starfish experiment, 2 robots at one of the edges of the configuration failed to localize. Upon inspection, it turned out that these were neighboring a robot that had switched off due to low battery. This was exacerbated by the fact that edge robots have the fewest neighbors. They were therefore failing to meet one or more of conditions to localize described in Sec. 2.2. These two robots were man-

ually removed to allow the consensus-based transition to take place. In the future, this problem could be addressed by: (i) slightly relaxing the conditions to localize for edge robots, and/or (ii) replacing the consensus-based transition algorithm with a quorum-based one. Both transition procedures lasted around 10 min.

The final shape formations in all three experiments shows good agreement with the formed coordinate system (see Fig. 1). In the starfish experiment, the final configuration contained 167 robots[1], whereas 164 robots had considered themselves to be part of the shape according to the coordinate system (error = $+1.82\%$). In the wrench experiment, the corresponding numbers were 363 and 356 (error = $+1.97\%$), and in the U shape experiment, they were 294 and 284 (error = $+3.52\%$). Additionally, in the wrench experiment there were four robots in the final configuration that were not connected to the shape, but still inside its convex hull. The reasons for the number of robots left in the final shape being larger than the count from the coordinate system count are twofold, and due to hardware issues. Firstly, some robots stopped working due to a low battery level, and robots close to the shape are more susceptible to this because they have to wait the longest before starting to move. Secondly, some robots suffered from miscalibration or malfunction of their light sensor and/or motors, and as such failed to perform successful phototaxis or antiphototaxis, occasionally becoming stuck to the shape. In the U shape experiment, three in-shape robots were knocked out of place by moving robots; no in-shape robots suffered this effect in the starfish experiment. These low errors and the high fidelity of the final shapes confirm that although the motion strategy of the non-shape robots is highly-parallelized and aggressive, the shape is not significantly distorted thanks to a successful collision avoidance behavior.

4 Comparison of Self-Disassembly to Additive Self-Assembly

The Kilobot platform not only allows us to validate complex collective behaviors at scale, but also allows for a direct comparison of different algorithmic approaches on the same hardware testbed. Here we compare two approaches to shape formation using the Kilobot platform: (i) an additive self-assembly approach, where the shape grows in layers starting from a seed [12] (similar to approaches such as [13]), and (ii) the subtractive self-disassembly approach presented here, which is similar to the Pebbles [6] and Miche [7] systems.

The self-disassembly algorithm has some disadvantages, most importantly the fact that it is more wasteful because not all robots are used in the shape. This effect is exacerbated in cases where the shapes bounding box (or convex hull) is only sparsely populated by the shape. Also, the shape classes formed by both approaches have different constraints. The self-assembly algorithm in [12] could form any simply-connected shape (i.e. no holes), and there exist self-assembly algorithms that can handle holes as well [4]. In contrast, the self-disassembly algorithm pre-

[1] The number of robots in the final shape is defined as the maximal connected subgraph, with two robots being considered adjacent if they are within 1 body length apart.

sented is currently limited to shapes classes 1 and 2, and by its nature fundamentally lacks the ability to handle holes in the shape. However, the self-disassembly algorithm can form disjointed shapes from a single seed, which is much harder to achieve with self-assembly algorithms.

At the same time, the self-disassembly algorithm has a significant advantage in terms of motion parallelism and time efficiency, both theoretically and experimentally. Theoretically, the additive self-assembly approach builds in layers that severely restricts parallelism, resulting in an $\mathcal{O}(n)$ scalability [12]. In contrast, self-disassembly has a high level of parallelism, and the time taken is bounded by the longest path that a non-shape robot takes in order to exit the convex hull, resulting in $\mathcal{O}(\sqrt{n})$ for a square or diameter of the shape in other cases. Experimentally however, this effect is even more more dramatic. The wrench experiment took around 6 hours to self-assemble, and only around 40 minutes to form through self-disassembly. The high efficiency comes from the fact that self-disassembly relies on fast imprecise motion; phototaxis and antiphototaxis are simple robot behaviors that are feasible at higher speeds and are collectively robust without collision avoidance. In contrast, self-assembly requires precise edge-following motion which is reliable only at lower speeds, and collectively creates traffic lanes that allow the slowest robot's speed to dominate. Finally, the wrench experiment also demonstrates that self-disassembly algorithms can potentially achieve higher accuracy; in the previous self-assembly algorithm the final wrench coordinate system had a significant bend whereas with self-disassembly the coordinate system was close to perfect. For shapes with high-aspect ratios, small errors and drift can easily accumulate in additive algorithms whereas in a subtractive algorithm, the initial layout creates a long-range consistent localization which the shape can take advantage of.

5 Conclusion and Future Work

In this paper we present and experimentally demonstrate a self-disassembly algorithm on a large-scale robot collective (up to 725 Kilobots). Our theoretical and experimental results suggest that such a self-disassembly algorithm can achieve a wide class of shapes with high efficiency and accuracy, making it a good candidate for shape formation in modular robots and programmable materials. In the future, we intend to extend this algorithm by introducing a stochastic motion component, in order to deal with a class of shapes for which phototaxis and antiphototaxis alone are not sufficient for self-disassembly.

Acknowledgments

This research was funded by a DARPA DSO grant and the Wyss Institute for Biologically Inspired Engineering.

References

1. Anderson, C., Theraulaz, G., Deneubourg, J.L.: Self-assemblages in insect societies. Insectes sociaux **49**(2), 99–110 (2002)
2. Bishop, J., Burden, S., Klavins, E., Kreisberg, R., Malone, W., Napp, N., Nguyen, T.: Programmable parts: A demonstration of the grammatical approach to self-organization. In: Proc. 2005 Int. Conf. Intell. Robot & Sys. (IROS) (2005)
3. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeu, E.: Self-organization in biological systems. Princeton University Press, Princeton, NJ (2003)
4. De Rosa, M., Goldstein, S.C., Lee, P., Campbell, J., Pillai, P.: Scalable shape sculpting via hole motion: Motion planning in lattice-constrained modular robots. In: Proc. ICRA (2006)
5. Gauci, M., Nagpal, R., Rubenstein, M.: Programmable self-disassembly for shapeformation in large-scale robot collectives: Online supplementary material (2016). URL http://goo.gl/VKnMmk
6. Gilpin, K., Knaian, A., Rus, D.: Robot pebbles: One centimeter modules for programmable matter through self-disassembly. In: Proc. 2010 Int. Conf. Robot. Autom. (ICRA) (2010)
7. Gilpin, K., Kotay, K., Rus, D.: Miche: Modular shape formation by self-disassembly. Int. J. Rob. Res. **27**, 345–372 (2008)
8. Groß, R., Bonani, M., Mondada, F., Dorigo, M.: Autonomous self-assembly in swarm-bots. IEEE Trans. Robot. **22**, 1115–1130 (2006)
9. Haghighat, B., Platerrier, B., Waegeli, L., Martinoli, A.: Synthesizing rulesets for programmable robotic self-assembly: A case study using floating miniaturized robots. In: Proc. 10th Int. Conf. Swarm Intell. (ANTS) (2016)
10. Moore, D., Leonard, J., Rus, D., Teller, S.: Robust distributed network localization with noisy range measurements. In: Proc. 2nd Int. Conf. Embedded Networked Sensor Systems, pp. 50–61. ACM (2004)
11. O'Grady, R., Christensen, A.L., Dorigo, M.: Swarmorph: multirobot morphogenesis using directional self-assembly. IEEE Transactions on Robotics **25**(3), 738–743 (2009)
12. Rubenstein, M., Cornejo, A., Nagpal, R.: Programmable self-assembly in a thousand-robot swarm. Science **345**, 795–799 (2014)
13. Stoy, K., Nagpal, R.: Self-reconfiguration using directed growth. In: Proc. 7th Int. Conf. Distrib. Auton. Robot. Sys. (DARS) (2004)
14. White, P., Zykov, V., Bongard, J., Lipson, H.: Three dimensional stochastic reconfiguration of modular robots. In: Proc. Robot.: Sci. & Sys. I (2005)
15. Wolpert, L., Tickle, C., Lawrence, P., Meyerowitz, E., Robertson, E., Smith, J., Jessell, T.: Principles of Development, 4th edn. Oxford University Press, Oxford, UK (2011)
16. Yim, M., Shen, W.M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., Chirikjian, G.S.: Modular self-reconfigurable robot systems. IEEE Robot. Autom. Mag. pp. 43–52 (2007)
17. Zykov, V., Mytilinaios, E., Adams, B., Lipson, H.: Robotics: self-reproducing machines. Nature **435**, 163–164 (2005)