Self Assembling Robots in an Underwater Environment

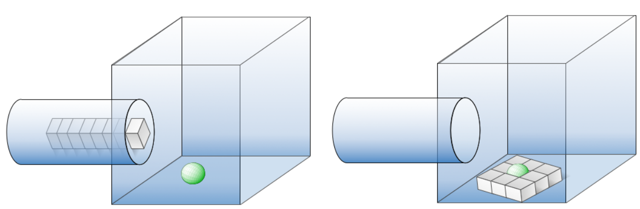
Varadarajan Ganesan and Mandar A. Chitre
Acoustic Research Laboratory, Tropical Marine Science Institute
National University of Singapore, Singapore 119223
e-mail: {varadarajan, mandar}@arl.nus.edu.sg

Abstract— **We present a novel rule based approach to carry out self-assembly in underwater environments. These rules define the manner in which small building blocks combine to form larger shapes. Unlike traditional AUVs or ROVs which are only capable of performing certain tasks, self-assembled systems have the potential to be used for different purposes in underwater environments based on the shape they assume. Stochastic forces are used to drive the assembly process. We demonstrate the feasibility of our approach in simulation by assembling four different shapes - 'L' and 'T' alphabets, a straight line and a square. We then examine the factors that affect the time taken to reach the final configuration.**

I. Introduction

Self assembling robots have gained a lot of interest over the years owing to their ability to form complex shapes from relatively simple and modular structures [1], [2]. These systems have the potential to assemble themselves according to the environment they work in and the nature of the task at hand. For example, a group of robots can assemble in a particular manner to transport items or organize themselves in another fashion to move in confined spaces while performing search and rescue operations during the time of a disaster.

Likewise, in underwater environments, self-assembling systems can potentially be used to perform multiple tasks such as transportation, surveying remote areas or moving through confined spaces. Fig. 1 shows an illustration of how individual modules assemble themselves into a slender structure (Fig. 1(a)) to move through a confined space in an underwater environment after which they re-assemble themselves into a flat structure (Fig. 1(b)) to move an object in a water tank.



(a) First Configuration (b) Second Configuration

Fig. 1: Illustration of a group of robots (cubes) moving through a confined space and re-assembling themselves to move an object.

As a consequence of self-assembly, simple building blocks can configure themselves into different shapes to perform different tasks as and when required. This manner of self-assembly derives its motivation from the biological process called protein folding [3]. Protein strings derive their respective functionalities based on the structure they fold into. Likewise, a group of small individual robots provide different use cases based on which structure they configure themselves into.

While self-assembling systems do have advantages as stated above, issues such as design, control and co-ordination of individual robots need to be addressed. Various groups that have tried to tackle these issues and have adopted approaches that can be broadly classified into two categories: *active* and *passive* assembly. While active assembly refers to the use of actuators to drive the assembly process, passive assembly makes use of stochastic forces instead.

Active assembly: It can again be classified into two types. The first class has a fixed number of simple modules linked together to form structures like chains [4], [5], 3D lattices [6] or complex 3-D mechanical structures [7]. These systems use the actuators of the individual modules to reconfigure themselves into meaningful shapes.

In the second class, individual modules are initially unconnected [8] or remain unconnected throughout [9]. While in [8], there is a seed robot to which all the other robots attach themselves to using a gripper and communicate with each other locally to assemble themselves into the desired final configuration, in [9], there is no seed robot and the individual modules do not connect to each other during self-assembly.

Passive assembly: This type of assembly derives its motivation from biological processes like DNA replication and protein folding which are randomly driven [3]. Likewise, passively assembled systems are generated by allowing individual modules to move in a stochastic manner and controlling which modules connect when they come close to one another.

White *et al.* in [10] have demonstrated the self-assembly of simple triangular and square shaped modules moving randomly on an air table. Individual modules communicate and derive power only after the attach themselves directly or indirectly to a seed module (powered externally). The modules use electromagnets for the purpose of linking. While the authors in [11] have demonstrated the self-assembly of 2D structures like hexagon from simple triangular structures on an air table using a graph grammar based approach to program the individual modules, Griffith *et al.* in [12] designed a similar system to demonstrate self-replication of template strings. In both [11] and [12], all modules are individually powered and begin communication only upon establishing physical connection.

White *et al.* have extended their work to three dimensions in [13] and generate stochastic forces by agitating the fluidic medium in a tank. In simulation and experiments, they have a base plate that serves as the power source and central node for communication, and the growing structure needs to be attached to the base plate to draw power and communicate with each other. The works in [14], [15] and [16] also use a similar approach where individual modules attach themselves to the base plate to grow and create well defined structures. But they direct the fluid flow in the tank by using valves in the base plate to guide the process of self-assembly.

Our work stems from the observation that if a completely random process is sufficient to replicate DNA strings or allow proteins to reach their functional form through protein folding, we should be able to exploit this stochasticity to create the desired 3D structure (passive assembly). Unlike the works of [14], [15] and [16], we adopt an entirely random approach without controlling any of the operational parameters. We also extend the works of [11] and [12] by implementing it in three dimensions to replicate the scenario in an underwater environment. However, instead of using a graph grammar approach which can get easily complicated in three dimensions, we formulate a novel rule based approach which is adopted by each module during self-assembly. By formulating appropriate rules as to how the individual robots combine, we control the self-assembly process to generate the desired structure.

II. Simulation Methodolgy

The process of self-assembly is simulated using Open Dynamics Engine [17], an open source physics engine that simulates rigid body dynamics, collision detection etc. We model a tank filled with a fluid and individual modules (cubes) experiencing random forces due to the agitation induced in the fluid. The velocity field is modelled as follows:

$$\begin{aligned}
v_x &= a_{0_t}\sin(2\pi f_0 x + \phi) + a_{1_t}\sin(2\pi f_1 t + \psi) && (1)\\
v_z &= a_{0_t}\cos(2\pi f_0 z + \phi) + a_{1_t}\cos(2\pi f_1 t + \psi) && (2)\\
v_y &= \mathcal{N}(0, \sigma^2) && (3)\\
a_{0_t} &= \mathcal{N}(\mu_0, \sigma_0^2) && (4)\\
a_{1_t} &= \mathcal{N}(\mu_1, \sigma_1^2) && (5)
\end{aligned}$$

While the first term in the v_x and v_z components provide a spatial variation to the velocity field in the xz plane, the second term provides a directional variation such that the velocity field rotates as time, t, elapses. The coefficients a_{0_t} and a_{1_t} are drawn from a Gaussian distribution, with mean μ_1 and μ_2, and variance σ_1^2 and σ_2^2 at every time step, t, to provide randomness to the velocity field. v_y is also drawn from a Gaussian distribution to model the disturbance in the y direction of velocity field. The values of $\mu_0, \mu_1, \sigma^2, \sigma_0^2, \sigma_1^2, f_0, f_1, \phi$ and ψ are chosen such that the cubes collide with each other and the walls of the tank.

This velocity field exerts random forces on the cube which is a manifestation of the drag force acting on the cube. Each cube is "divided" into 8 equal smaller cubes and the drag forces are calculated for each cube separately and applied at their respective centres. We adopt this approach primarily to account for rotational forces. The drag force is calculated in accordance with (6) where C_d is the drag coefficient (in this case, for a cube), A is the area of the cross section, ρ is the density of the fluid medium and $\mathbf{v}_{\text{rel}}$ is the velocity of the cube relative to the flow field.

$$\mathbf{f} = -\frac{1}{2}C_d\rho A|\mathbf{v}_{\text{rel}}|\mathbf{v}_{\text{rel}} \quad (6)$$

After applying the drag (random) forces, the cubes start to move in a random yet circular fashion overall. This type of motion can be reproduced in experiments by using directional pumps.

We consider the individual faces of each cube to be distinct and are capable of attracting the faces of other cubes at close proximity. We also assume that each face knows which other face it is colliding with. All these conditions can be accomplished in experiments by coloring the faces distinctly and having color sensors and electromagnets attached to each face. When two faces are close to each other and can attach themselves in accordance with the rules formulated, the faces turn on their electromagnets after a random back-off. The face to turn on its electromagnet first gets detected by the electromagnet in the other face since a current will be induced in its coil due to the presence of a magnetic field. As a result, the second face decides not to turn on its electromagnet and gets attached to the first face by automatically assuming the opposite polarity. This approach can be adopted in experiments because if both the faces were to turn on their respective electromagnets, then the polarity of the faces need to be taken into account which would constraint the self-assembly process. Finally, the objective of the paper is to show how different 3D structures are generated by formulating appropriate rules as to which face attracts which other face.

III. Rules

We adopt a mathematical approach to formulate the rules required for self-assembly. Let the cubes be denoted by V_i, where $i \in \{1, \ldots, n\}$ and n is the total number of cubes. We denote each face of a particular cube by $f_j^{V_i}$ where $j \in \{1, \ldots, 6\}$. However, for the shapes being assembled, it is sufficient that we represent the six faces with just two values of j. Table I shows the label j and the corresponding faces it is used to represent. The number of faces that are attached, by virtue of the electromagnet being turned on, are also kept track of and is denoted by $C(V_i)$. We assume that the cubes can relay their respective counters, $C(V_i)$. This can be accomplished in experiments by having different colored LEDs and their respective color sensors on the faces of the cubes. The rules for self-assembly are represented by Φ.

TABLE I

j	face
1	Top/Bottom
2	Right/Left/Front/Back

An example of a rule which allows the side faces of a cube to attach itself with the top or bottom face of another cube is written as follows:

$$\Phi : f_1^{V_1} - f_2^{V_2} \tag{7}$$

where the '$-$' operator indicates that the relevant faces can attract each other at close proximities by turning on their magnets.

We now formulate the rules required to self-assemble the following shapes:

A. Straight Line

A straight line is assembled by formulating the rule that only allows the top or bottom face of one cube to attract the top or bottom face of another cube. This can be formally written as follows:

$$\Phi_1 : f_1^{V_i} - f_1^{V_k} \ ; i,k \in \{1,\ldots,n\} \ ; i \neq k \tag{8}$$

B. 'L' Shape

An 'L' shaped structure is generated from four cubes by formulating two rules that are executed sequentially. Additionally, individual cubes make use of the counter $C(V_i)$, which is the the number of faces activated. Initially, when the counter is zero for all the cubes, the rule only allows the top or bottom face of one cube to attract the top or bottom face of another cube. We ensure cubes with similar counters only attract each other. As a result we get two pairs of two cubes each and all their counters are updated to one. When the counter is one, the rule allows either the right, left, back or front face to attract either the top or bottom face which results in an 'L shape. The assembly process can be written as follows:

$$\Phi_1 \quad : \quad f_1^{V_i} - f_1^{V_k} \ ; C(V_i) = C(V_k) = 0 \tag{9}$$
$$\Phi_2 \quad : \quad f_1^{V_i} - f_2^{V_k} \ ; C(V_i) = C(V_k) = 1 \tag{10}$$
$$i,k \quad \in \quad \{1,\ldots,4\} \ ; i \neq k \tag{11}$$

C. 'T' Shape

A 'T' shaped structured can be generated by following the same rules to generate an 'L' shaped structure followed by adding another cube at the junction of the 'L' shape. The rules are as follows:

$$\Phi_1 \quad : \quad f_1^{V_i} - f_1^{V_k} \ ; C(V_i) = C(V_k) = 0 \tag{12}$$
$$\Phi_2 \quad : \quad f_1^{V_1} - f_2^{V_k} \ ; C(V_i) = C(V_k) = 1 \tag{13}$$
$$\Phi_3 \quad : \quad f_1^{V_i} - f_1^{V_k} \ ; C(V_i) = 2 \ ; C(V_k) = 0 \tag{14}$$
$$i,k \quad \in \quad \{1,\ldots,5\} \ ; i \neq k \tag{15}$$

Note that after generating the 'L' shape first, there will be two cubes with $C(V_j) = 2$. But only the cube at the junction will have its top/bottom face free. In order to be consistent with Φ_1, we add the fifth cube there accordingly. Furthermore, as the complexity of the shape increases, it can be seen that the number of rules also increase.

D. Square

A square shape can also be assembled similar to an 'L' shape by modifying Φ_2 alone. By allowing either the right, left, back or front face to attract either the right, left, back or front face we can generate a square. The assembly process can be formally written as follows:

$$\Phi_1 \quad : \quad f_1^{V_i} - f_1^{V_k} \ ; C(V_i) = C(V_k) = 0 \tag{16}$$
$$\Phi_2 \quad : \quad f_2^{V_i} - f_2^{V_k} \ ; C(V_i) = C(V_k) = 1 \tag{17}$$
$$i,k \quad \in \quad \{1,\ldots,4\} \ ; i \neq k \tag{18}$$

Fig. 2 shows an illustration of the different rules that guide the self-assembly process. The top and bottom faces are black in color and the remaining faces are white. For the self-assembly of a square, an ideal scenario is being illustrated in Fig. 2(d). However, the cubes rely on collisions with the walls of the tank to align themselves into the correct structure while executing Φ_2. This can be seen in Fig. 4.

IV. Simulation Setup

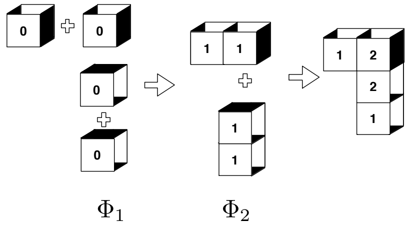
Four cubes were used for assembling all structures except the 'T' shaped structure, where five cubes were used. The dimensions of the tank were 2 units $\times$ 2 units $\times$ 3 units and the edge length of each cube was 0.25 units. The cubes were neutrally buoyant and placed at random locations in the tank and with random orientations at the start of the simulation. The faces of the cubes attract each other (in accordance with Φ) when they collide with each other and the angle between their surface normals is greater that $135°$. This type of attraction emulates the working of small magnets where the attractive forces become dominant only at close proximities.

V. Results & Discussion

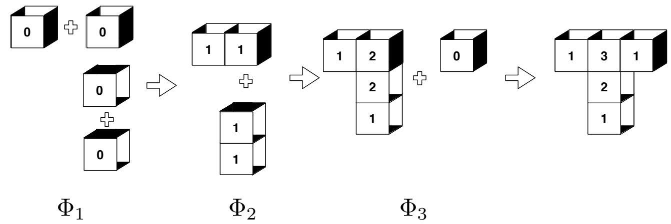
Each shape was assembled a hundred times and statistics of the assembly processes were collected. All the simulations were successful and the desired shapes were generated without any fault in the final structure. The generated structures are shown in Fig. 3. Only the top and bottom faces are colored red for ease of comprehension. The other four faces are kept white since the rules formulated treat all of them equally. But they provide the flexibility to be treated distinctly to assemble more complex structures.

As mentioned in Section III-D, the square structure relies on collisions to complete the self-assembly process through Φ_2 which can be seen in Fig. 4. Fig. 5 shows the histogram plots of the time taken to generate the four shapes. Additionally, the box plots in Fig. 6 shows the distribution of the time taken to complete the process of self-assembly for the four shapes.

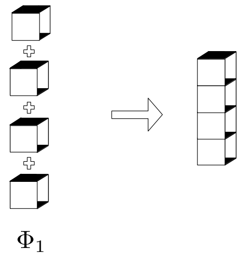
It can be seen from Fig. 6 that the 'L' shape takes the least amount of time to complete self-assembly. This can be attributed to the fact that for Φ_2, there are four faces ($j = 2$) that are available for bonding with two faces of



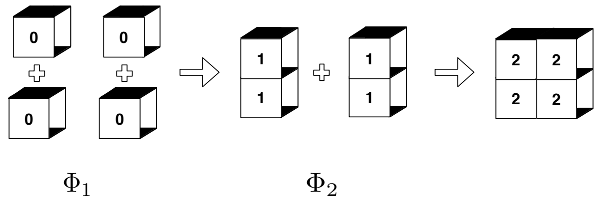
(a) L Shape



(b) T Shape

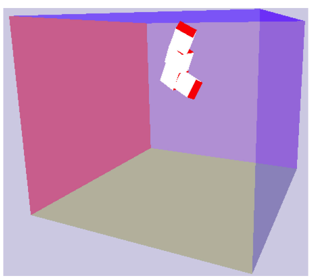


(c) Straight line

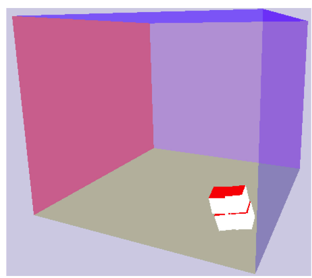


(d) Square Shape

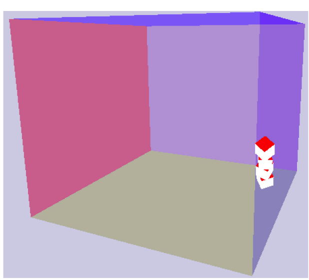
Fig. 2: Illustration of different rules to carry out the self-assembly process for different shapes. The counters, $C(V_i)$, of the cubes are also shown on their respective front faces for the assembly processes that require it.



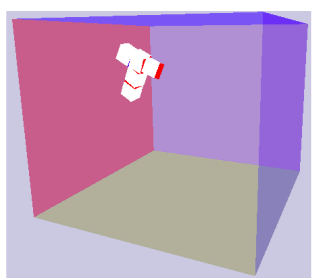
(a) L shape



(b) Square shape

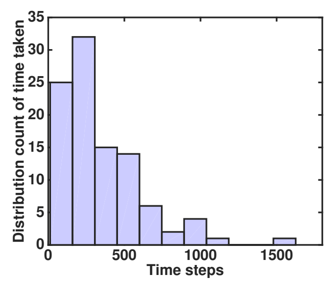


(c) Straight line

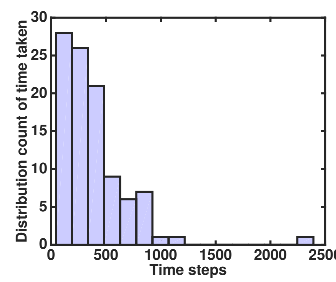


(d) T shape

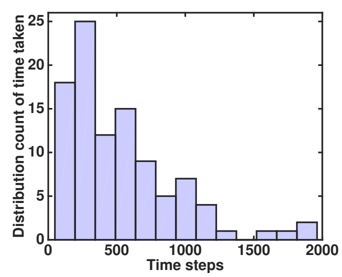
Fig. 3: Different shapes formed during self-assembly with appropriate rules.



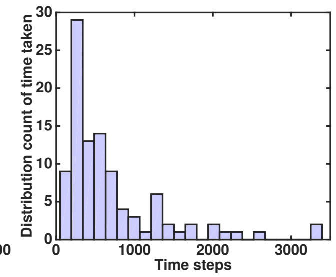
(a) L shape



(b) Square shape

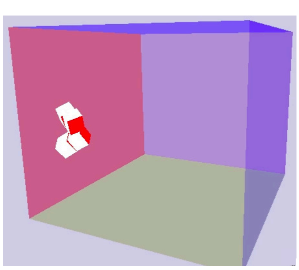


(c) T Shape

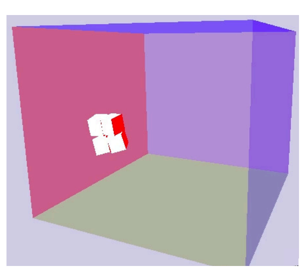


(d) Straight Line

Fig. 5: Histogram plots of time taken to complete self-assembly.



(a) Alignment of cubes immediately before a collision



(b) Collision with the front wall of the tank aiding in comepletion of self-assembly

Fig. 4: Self-assembly of a square.

another cube ($j = 1$). This results in a higher number of favourable collisions that lead to attraction between the faces and hence, less time to complete self-assembly. The square shape takes slightly more time than the 'L' shape as expected because apart from executing its Φ_2, it relies on collisions with the walls of the tank for the cubes to align themselves into the final structure. The 'T' shape takes even more time which can be explained by the fact that the 'L' shape needs to be formed first. Following this, there is only one face that is available for bonding with the fifth cube to generate the 'T' shape (Φ_3). Hence, the chances of that

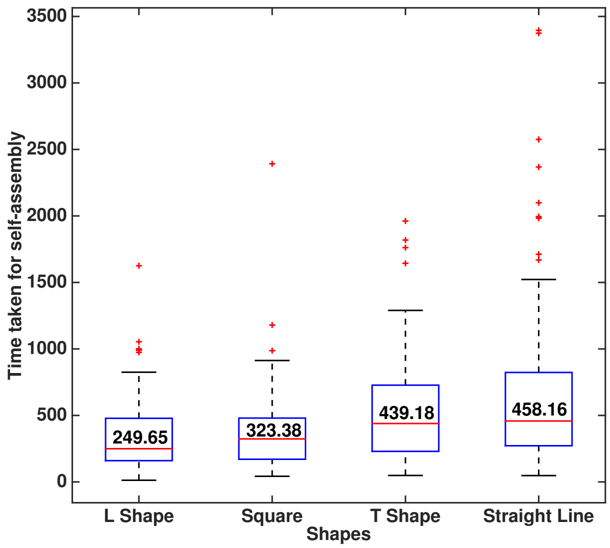


Fig. 6: Distribution of time taken for self-assembly of different shapes.

particular collision (leading to attraction) occurring is really low. Interestingly, the straight line takes the most amount of time to self-assemble even though the cubes require only one rule to be executed. Again, this can be explained by the number of faces that are available for bonding. There is only one face that is capable of attracting only two other faces, and the chances of both the faces coming close to each other is extremely low. Furthermore, this process needs to happen three times to create a straight line of four cubes resulting in a longer time taken for the cubes to assemble themselves.

Hence, it is clear that the number of rules, i.e., complexity of the shape, alone does not determine the time taken to complete self-assembly. The number of faces required for favourable collisions to occur, leading to attraction between the faces, also plays an important role in determining the time taken by the cubes to assemble themselves.

VI. Conclusion

We have developed a novel rule basis model for self-assembly of cubes into different shapes. A passive approach which makes use of stochastic forces to drive the assembly process was adopted. Simulations were carried out to show the feasibility of the proposed approach. Factors affecting the time taken to complete self-assembly was also studied. The proposed approach has the potential to be used in underwater environments where small building blocks can assemble themselves into different shapes for different tasks. Work is underway to build cubes with the necessary sensors and magnets in order to experimentally demonstrate the ideas presented in this paper.

However, the rules generated treat the top/bottom and all the side faces as similar to one another and hence, we can only assemble the structures mentioned earlier. We intend to explore the use of other faces in a distinct manner to generate more complex structures as a part of our future work. We also intend to develop statistical methods to predict the time taken for self-assembly of different shapes. This tool will be helpful in deciding which shapes to use for the different tasks at hand as some shapes would take longer time to assemble which may not be desirable for operational purposes.

Finally, instead of formulating rigid rules for self-assembly, an algorithm with generates the rules based on the desired final configuration (which is taken as input) provides more flexibility in terms of creating complex shapes. We hope to create such an algorithm which completely automates the process of self-assembly in underwater environments.

References

[1] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *IEEE International Conference on Robotics and Automation*, May 2012, pp. 3293–3298.

[2] R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous self-assembly in swarm-bots," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1115–1130, Dec 2006.

[3] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Molecular Biology of the Cell, 4th edition*. Garland Science, 2001.

[4] M. Yim, D. Duff, and K. Roufas, "Polybot: a modular reconfigurable robot," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2000, pp. 514–520.

[5] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji, "M-tran: self-reconfigurable modular robotic system," *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 4, pp. 431–441, Dec 2002.

[6] E. H. Østergaard, K. Kassow, R. Beck, and H. Lund, "Design of the atron lattice-based self-reconfigurable robot," *Autonomous Robots*, vol. 21, no. 2, pp. 165–183, 2006.

[7] S. Murata, H. Kurokawa, E. Yoshida, K. Tomita, and S. Kokaji, "A 3-d self-reconfigurable structure," in *IEEE International Conference on Robotics and Automation*, vol. 1, May 1998, pp. 432–439 vol.1.

[8] R. O'Grady, A. Christensen, and M. Dorigo, "Autonomous reconfiguration in a self-assembling multi-robot system," in *Ant Colony Optimization and Swarm Intelligence*, ser. Lecture Notes in Computer Science, M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stützle, and A. Winfield, Eds. Springer Berlin Heidelberg, 2008, vol. 5217, pp. 259–266.

[9] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014. [Online]. Available: http://www.sciencemag.org/content/345/6198/795.abstract

[10] P. White, K. Kopanski, and H. Lipson, "Stochastic self-reconfigurable cellular robotics," in *IEEE International Conference on Robotics and Automation*, vol. 3, April 2004, pp. 2888–2893.

[11] J. Bishop, S. Burden, E. Klavins, R. Kreisberg, W. Malone, N. Napp, and T. Nguyen, "Programmable parts: a demonstration of the grammatical approach to self-organization," in *IEEE International Conference on Intelligent Robots and Systems*, Aug 2005, pp. 3684–3691.

[12] S. Griffith, D. Goldwater, and J. M. Jacobson, "Robotics: Self-replication from random parts," *Nature*, vol. 437, no. 7059, pp. 636–636, 2005.

[13] P. White, V. Zykov, J. C. Bongard, and H. Lipson, "Three dimensional stochastic reconfiguration of modular robots." in *Robotics: Science and Systems*. Cambridge, 2005, pp. 161–168.

[14] M. T. Tolley and H. Lipson, "Programmable 3d stochastic fluidic assembly of cm-scale modules," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2011.

[15] M. T. Tolley, M. Kalontarov, J. Neubert, D. Erickson, and H. Lipson, "Stochastic modular robotic systems: A study of fluidic assembly strategies," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 518–530, June 2010.

[16] M. T. Tolley and H. Lipson, "Fluidic manipulation for scalable stochastic 3d assembly of modular robots," in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 2473–2478.

[17] Open dynamics engine. [Online]. Available: http://www.ode.org