Programmable matter as a cyber-physical conjugation

Julien Bourgeois, Benoit Piranda, André Naz, Thadeu Tucci,
Hakim Mabed, Dominique Douthaut and Nicolas Boillot
Université de Franche-Comté,
Université Bourgogne Franche-Comté (UBFC),
FEMTO-ST Institute,
UMR CNRS 6174, France,
{FirstName.LastName}@femto-st.fr

Hicham Lakhlef
IRISA/INRIA Rennes Bretagne Atlantique,
Campus Universitaire de Beaulieu
35042 Rennes Cedex, FRANCE
hicham.lakhlef@irisa.fr

***Abstract*—Programmable matter i.e. matter that can change its physical properties, more likely its shape according to an internal or an external action is a good example of a cybermatics component. As it links a cyberized shape to real matter, it is a straight example of cyber-physical conjugation. But, this interaction between virtual and real worlds needs two elements. The first one is to find a way to represent the cyberized object using programmable matter and the second is to be able to adapt the matter to the cyberized changes.**

This article presents the progresses made in these two topics within the Claytronics project.

I. INTRODUCTION

The new era of a digital world started with the invention of the computer and has been popularized with the invention of the Internet and later with the development of the web. This digital world, coined as cyber world, comprises many entities named cyber entities [1]. A cyber entity can be a representation of an existing thing/entity but also a cyber entity without a physical presence. On the one hand, an intelligent thing like a body sensor will leave a trace in the cyber world and will therefore have a cyber and a physical representation. On the other hand, an object of virtual art will only have a virtual representation. The development of this kind of cyber entity without a physical representation is supported by the progresses of machine learning algorithm and more generally by content generated by computers, later referred to as cyberization. Cyberization constitutes a field of research in itself which has been named as cybermatics by Ma and al. [2].

Programmable matter is a good example of a cybermatics component because it is directly concerned by cyber-physical conjugation. A shape is first described virtually in a computer, cyberized, and then transferred to the matter. The matter can then evolve changing its shape either by moving by itself or following users needs and the cyberized version of the matter must reflect this evolution. The cyberized and the real version of an object are therefore in interaction, if one change the other must reflect this change. This interaction needs two elements. The first one is to find a way to represent the cyberized object using programmable matter and the second is to be able to adapt the matter to the cyberized changes.

This article presents the progresses made in these two topics. It is structured in three parts: the first one presents the programmable matter concept, the second describes the relation between a cyber representation and the programmable matter and the last one presents the ways to adapt matter to its cyber representation.

II. PROGRAMMABLE MATTER

A. Definition

There are many different ideas behind the concept of programmable matter. On a broad scope, programmable matter can change its physical properties more likely its shape according to an internal or an external action.

The term of programmable matter has been first coined by T. Toffoli in [3] but the meaning was quite different than a self-reconfiguring matter. In this article, Toffoli defines programmable matter as a "*[...] computer architecture that closely reflects those resources and constraints of physical matter insofar as they are relevant to the most efficient use of this matter as a computational medium.*" which means the matter is a computational resource. It is three years later, then programmable matter is really defined by S.C. Goldstein and al. In [4], programmable matter is defined as a cyber-physical conjugation: "[...] *programmable matter will allow us to take a (big) step beyond virtual reality, to synthetic reality, an environment in which all the objects in a users environment (including the ones inserted by the computer) are physically realized."*.

Many technologies or projects claim a programmable matter label: programmable matter using folding [5], 4D printing [6], modular self-reconfigurable robots [7] [4], quantum wellstone [8] or DNA structures [9] [10].

Although, all these projects are achieving programmable matter in some ways, they have very different output and their properties can greatly vary.

- Evolutivity: Can the shape of the matter vary over time, and if yes, to what degree?
- Programmability: Shape transformation is driven by a program or simply respond to an external stimulus (i.e. 4D printing example)

- Autonomy: A computer is driving the matter directly or the matter is autonomous and executes a program can change its shape
- Interactivity: Programmable matter can change its virtual representation back into a cyberized model

Only modular self-reconfigurable robots have all these properties as they embed computation. The most advanced research in this field is the Claytronics project which will be detailed in the next section.

B. The Claytronics project

Claytronics, which stands for *Clay-Electronics* is the name of a robotics project initiated by Carnegie Mellon University and Intel Corporation, and then joined by FEMTO-ST Institute. Mm-scale robots called Catoms, for Claytronics Atoms, are assembled to form larger objects. The idea is that each micro-robot have very restricted or let us say only strictly mandatory functionalities and as each catom is simple, hundreds of thousands can be assembled all together to create new solid objects of any shape or size. A catom is a mass-producible, sub-mm, MEMS using computationally controlled forces for adhesion and locomotion. Each catom therefore embeds a chip for computation and for driving its actuators and communication capabilities. Two communication hardware are studied: using electrostatic electrodes to transmit a signal, i.e. by contact communication or using nanowireless communications [11][12]. A first prototype has been realized which embeds actuation and a chip for managing the movement of the micro-robots [13]. There are many challenges to solve before Claytronics could bring transform matter into programmable matter. These challenges and perspectives have been enumerated in [14][15].

C. Applications

Several ideas and examples of programmable matter usage have already been described in [16], [17], [18], [19]. One of them is collaborative work around a or several programmable matter objects. In [16], a collaborative diagnosis for doctors is described. A meeting is started between several doctors which are in different locations. They are using programmable matter as a way to ”see” each other. They can work around a programmable matter representation of the patient taking full advantage of the programmability of the object. They can remove unnecessary elements of model for example skin or muscles to display directly the injured part. They can also zoom in or zoom out in parts of interest.

Figure 1 presents a potential application of an interactive computer aided-design tool. The shape designed on a computer will be formed in a liquid environment to cancel gravity. The shape will be sent to the catoms using either optical or electromagnetic power feeding waves. The catoms will attach/detach according to the algorithm. Once the construction phase will be finished, the user will be able to take the object. The object will know it is complete or the user will send the order that construction is finished. The object will therefore change its behavior: it will allow catoms to leave and to be attached anywhere. As the power will be remotely provided, the user will be able to reconfigure the object by adding or removing catoms at his convenience. Once the shape is formed, the catoms will send their position back to the computer using the optical or the electromagnetic power feeding waves.

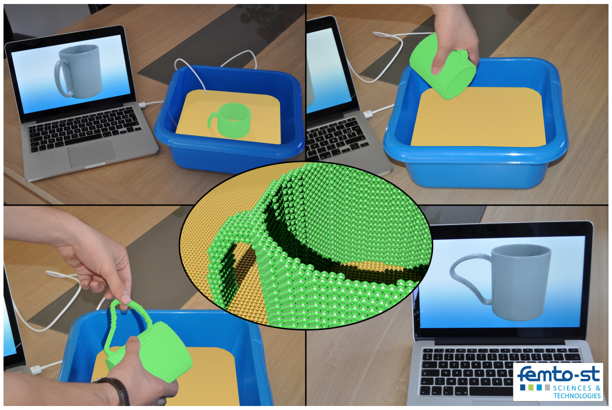
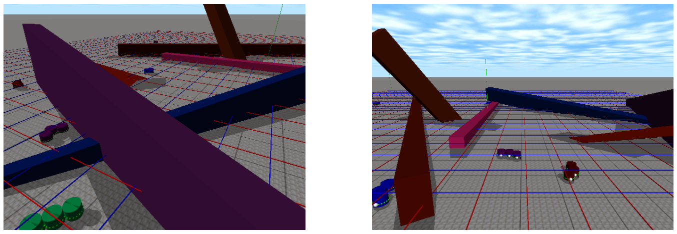


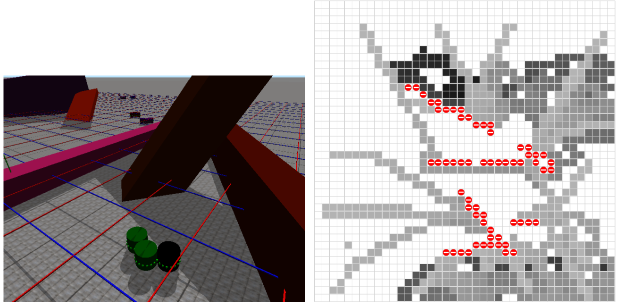
Figure 1. Computer aided-design tool using programmable matter

In these two examples, the fundamental cause of the gap between what people want and what technology delivers is that computation is limited by its media types (for example, text, audio, video), and therefore is confined to cyberspace. This either limits what technology can deliver or forces humans to adapt to technology in awkward ways.



(a) Snapshot of the environment (rear view)

(b) Snapshot of the environment (front view)



(c) Example of a walker escaping a cavity trap with backtrack

(d) Map overview at tick 14940

Figure 2. Walkers scanning a complex environment

In [12], a scalable application is described, where catoms scattered in small semi-autonomous groups, called ”walkers” collaboratively explore an unknown physical environment and inform a ”macro” user through a data sink. This allows a fast and detailed exploration of an unknown environment as each group of moving catoms are able to share the map of the environment while transmitting it to the sink (see Figure 2 for an example). This kind of application could be used in many areas. In structural health monitoring (SHM), walkers would be able to detect very small damages in structures by comparing the difference between the original and the current states of the structure. In a human body, they would allow detailed monitoring of an organ shape. For example, in the case of a cancer that would need 24/7 monitoring, it would help understanding the dynamics of the remission or development of the cancer.

III. TRANSFERRING CYBER REPRESENTATION INTO THE MATTER

A. Introduction

A major problem to tackle while transferring cyber representation into the matter is to consider catom storage limitation. Many algorithms that adapt cyber representation to real matter (e.g., [20] [21]) assume that each catom stores an exact specification of the cyber representation. Storing a list describing all the points of the cyber representation requires a linear amount of memory. While this approach allows a fine description of the cyberized object, this does not scale. In practice, a catom can for example not store a million of positions. We propose several approaches to address catom storage limitation, namely to use compact representation, shared representation and mapless representation.

B. Compact representation

As memory is a scarce resource, there is a need to study the most compact representation that can be used. Several ideas have been proposed to represent a cyberized object. Stoy [22] et al. followed by Fitch et al. [23] transform a CAD model, that is a largely used 3D format, into a set of overlapping bricks. Overlapping bricks are easy to work with, each brick can be represented by two coordinates but to increase precision there is a need to work with smaller bricks which can increase the representation size. In [24], Butler et al. published related work on mapping a configuration of modular robots. In their work they use for the representation of the configuration a binary matrix, with 0 corresponding to empty spaces and 1 corresponding to occupied spaces. It is a precise method to represent the goal configuration, easy to implement, without loss of details and in addition a simple operation could tell when the module is inside the model but the size of the map will grow linearly with the size of the robots giving some restrictions on scalability.

In the field of computer graphics, many solutions have been proposed to describe a 3D object. Two different models are analyzed to be part of description scene for modular robots, Triangle mesh and Constructive Solid Geometry (CSG). Triangle mesh is a very common representation of a 3D object, in literature we find it under the name of b-rep model (boundary representation) [25]. It consists in approximating the shape of an object by a set of small surfaces that defines the border of the object, and then interior and exterior spaces. The advantage of this description method is that we just have to describe a 2D surface in order to construct a 3D object, then it needs less memory.

An other advantage is given by the orientation of the surface: if every surface elements are defined counterclockwise, the normal vector is usable to determine if a point is inside or outside the volume relatively to the surface.

Although a wide number of 3D image software works based on this format, this solution does not guarantee the final object to be a solid where there is a place to check for a closed border of the object.

Constructive Solid Geometry [26] is a classical method for describing scenes in image synthesis. It consists in defining a tree of objects that can be combined in order to model the final scene. Leaves of the tree contain geometric models and internal nodes are associated to a geometrical transformation or a sub tree combination operator. Geometrical transformations are useful to apply displacements, rotations or scales in a sub scene, they are placed in unary internal node of the tree. Three combination operators are available: union, intersection and difference. Coding a scene using CSG Tree is very compact, because it consists in defining the volume occupied by the matter of the scene. Each object may be a simple geometric entity that can be described by some parameters. For example, a sphere or a cylinder need no intrinsic parameters their position and size are given by geometrical transformations, a torus is setup by the ratio of its radii. Describing a complex scene using CSG Tree becomes harder because it contains small details. In our case, the smallest size of a detail is the size of a catom.

The three parameters used for representing a cyberized shape are the precision of representation, the memory used and the complexity for a catom to identify its position in the shape. Regarding these three parameters, it seems CSG is the best tradeoff as it has the smallest memory footprint while it is quite easy for a catom to calculate its position in the shape. The only trouble could be in the case of a very complex object which could need more memory, more calculation to obtain finally less precision.

C. Shared representation

In the shared representation approach, the cyber representation is disseminated into the matter and is shared between the catoms taking into account their memory capability. Every catom does not store the complete cyber representation but instead only a part of it and can transparently access to locally and remotely stored parts of the cyber representation. This allows to store a fine-grained representation of possibly complex cyberized objects. This approach comes with the challenges of data dissemination, distributed query processing and routing in resource-constrained, infrastructureless, highly mobile and faulty systems.

Traditional peer-to-peer approaches (e.g., Chord [27]) fail to efficiently adapt to our situation because they were proposed to share data between devices connected over the Internet, assuming proper underlying routing services and placement of data independently of their position of usage. Moreover, most of these approaches consider node mobility as a fault implying high maintenance overhead in highly dynamic systems.

Some approaches that do not require any underlying routing service (e.g., pathDCS [28]) or that use a stateless routing protocol based on local-topology information (e.g., GHT [29]) have been proposed for Wireless Sensor Networks. However, in most of these approaches, data are also placed independently of their position of usage and support for topology changes due to node mobility is not sufficient for highly dynamic systems.

These challenges of data placement and distributed query processing in our resource-constrained, infrastructureless and highly mobile systems can be overcome by combining geographic routing, landmark routing, virtual coordinate based routing (e.g., [30]) and biased random walk methods. Moreover, in order to achieve efficient data retrieval, we propose to disseminate the parts of the cyber representation as close as possible to the area they are related to. Indeed, during the matter transformation phase, catoms mainly use information about geographically neighboring areas.

D. *Mapless*

A self-reconfiguration with a map of the target shape is not memory-efficient, because describing a target shape will consist of a set of positions which will require millions of positions. Each node should have a memory capacity of millions or billions of positions, hence the importance of a reconfiguration protocol without map of the target shape. That is, in the literature works when a self-reconfiguration process aims to reach a given target shape composed of a set of P positions (like a pixels for a given picture), each micro-robot records all the P positions. This is neither efficient nor scalable since we address here configurations with millions of nodes have a low-memory capacity, and millions of final positions because the MEMS nodes have a very small size.

Efficient approaches for self-reconfiguration of MEMS micro-robots have been proposed in [31][32] where nodes do not record any position and where the target shape is built incrementally. Each node in the current increment acts as a reference point for other nodes to form the next increment. The proposed model makes the assumption that each node can obtain the state of its physical neighbors to achieve self-reconfiguration. Using these states, nodes collaborate and help each other without the need for a global information. Therefore, these algorithms do not need to know the map of the target shape (i.e. coordinates of the micro-robots), consequently memory usage is dramatically reduced. The presented approach has the advantage of not requiring the node knowledge of its own position either.

IV. ADAPTING CYBER REPRESENTATIONS TO REAL MATTER

A. *Introduction*

Self-reconfiguration is a hard problem for three reasons. First, the number of possible unique configurations for a Modular Self-reconfigurable Robot (MSR) is huge: $(c.w)^n$ where n is the number of modules, c the number of possible connections per module and w the ways of connecting the modules together [33]. Second, as modules can possibly move at the same time the branching factor of the tree describing the configurations is $O(m^n)$ with m being the number of possible movements and n the number of modules free to move [34]. Third, as a consequence of the previous reason, the exploration space of a reconfiguration between two situations is exponential in n which prevents from finding a complete optimal planning. It has been shown recently that finding the optimal self-reconfiguration planning for chain-type MSRs is a NP-complete problem [35].

B. *Reconfiguration using blocks*

In [21], a distributed self-reconfiguration algorithm for 2D lattice-based MSR composed of cubic-shape sliding modules has been proposed. This algorithm is fully decentralized and can self-reconfigure a 2D MSR from any position to any other position given that the intersection of the two configurations is non-void and border is thicker than one block. Furthermore, the convergence is proved and as the algorithm is distributed, it can easily scale up to thousands of modules. Finally with this algorithm, all the blocks that can move, will move at each time step which makes the convergence fast.

This algorithm uses three ideas to cope with the difficulties of self-reconfiguration:

- *Complex movements.* Due to the sliding motion type, a MSR built from blocks has many blocking constraints and it is clearly a non-holonomic system. Motions rules have been introduced to cope with the difficulty of complex collaborative movements. This makes the new system a holonomic one. These rules are really simplifying complex movements as they describe a list of movements to be applied in each complex sequence.
- *Combinatorial explosion and convergence speed.* If the number of blocks increases, then the number of possible positions increase exponentially with the number of modules free to move, making a convergence to the right one a difficult problem. One solution would be to authorize only one block to move at a time but this would slow down the reconfiguration. To cope with this difficulty, new kind of meta-modules are defined. They act like a train with a leader and followers. This meta-module, directed by only one leader, simplifies the planning and allows parallel movements as all the movable blocks will move in parallel.
- *Movement planning.* The meta-modules are only allowed to move on the perimeter of the shape and only in the counterclockwise direction. This greatly simplifies the

movements planning and it allows any number of meta-modules to move to their final position.

C. Reconfiguration using spherical catoms

Using the Parallel Algorithm with Safe Connectivity (PASC) [31], spherical catoms can self-reconfigure to a diamond-shape without needing a map. In PASC, each node can only turn around a physical neighbor. But as network connectivity as to be kept, a node can only move around a neighbor if it does not break network connectivity. For this purpose, a spanning tree is created to dynamically manage the leaf nodes that can move.

To form the matrix of our square with $\sqrt{N} \times \sqrt{N}$ nodes, we begin with an incremental process with a correct square (for example 1x1). After that, we add each time a new sub-layer containing $3T + 2$ nodes, with $T \times T$ being the last square. Then, we add another sub-layer with $T + 2$ nodes taking positions at W direction relative to nodes of the last shape. If N is even, at the last layer we add $2T + 1$ nodes, with $T \times T$ is the last square. Figures 3 and 4 show an example. The choice of the middle node depends on the optimality of the parallelism. Let N be the network size, and $n = \left\lfloor \sqrt{N} \right\rfloor \left\lfloor \sqrt{N} \right\rfloor$, if n is odd, the middle node will be $mi = \frac{n+1}{2}$, as reported in Figure 6. If n is even, the middle node will be $mi = \frac{n}{2} - (\frac{\sqrt{n}}{2} - 1)$, as reported in Figure 5. The node index is simply its rank into the initial chain which starts from the top.

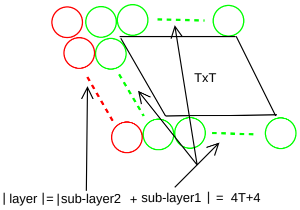


Figure 3. Number of nodes to be added to reach the next layer

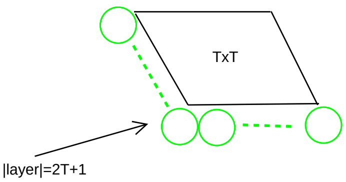


Figure 4. Number of nodes added in the last layer to reach the last shape when n is even

To find mi, the catoms need to know the size of the network. This is done by initiating a broadcast from a terminal node of the chain. Finally, the shape built itself whatever the number of catoms there is in the ensemble.

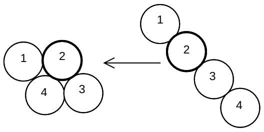


Figure 5. Example of initiator election when n is even, here the initiator is the node 2

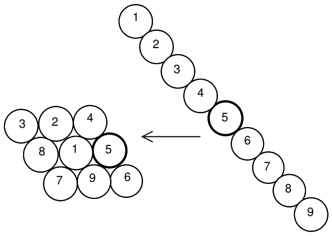


Figure 6. Example of initiator election when n odd, here the initiator is the node 5

V. CONCLUSION

Programmable matter is a promising case study for cybermatics as it is a good example of cyber-physical conjugation. In this article, we have presented different ways to transfer a cyber representation into the matter. Given, the physical constraints transferring a cyber representation is still an ongoing research work. Adapting cyber representations to real matter, needs self-reconfiguration which is a complex process deeply tied to physical capabilities of each catom. However, we have presented some ideas to simplify this process. Finally, inside the Claytronics project, many reflexion have been carried out on the technical point of view but a reflexion on the implications of cyber-physical conjugation is still to be made.

ACKNOWLEDGMENT

This work was supported in part by ANR/RGC (contracts ANR-12-IS02-0004-01 and 3-ZG1F) and Labex ACTION (ANR-11-LABX-01-01).

REFERENCES

[1] H. Ning, H. Liu, J. Ma, L. T. Yang, and R. Huang, "Cybermatics: Cyberphysicalsocialthinking hyperspace based science and technology," *Future Generation Computer Systems*, vol. 56, pp. 504 – 522, 2016.

[2] J. Ma, H. Ning, R. Huang, H. Liu, L. T. Yang, J. Chen, and G. Min, "Cybermatics: A holistic field for systematic study of cyber-enabled new worlds," *IEEE Access*, vol. 3, pp. 2270–2280, 2015.

[3] T. Toffoli and N. Margolus, "Programmable matter: concepts and realization," *Physica D: Nonlinear Phenomena*, vol. 47, no. 1, pp. 263–272, 1991.

[4] S. C. Goldstein and T. C. Mowry, "Claytronics: An instance of programmable matter," in *Wild and Crazy Ideas Session of ASPLOS*, Boston, MA, October 2004. [Online]. Available: http://www.cs.cmu.edu/ claytronics/papers/goldstein-waci04.pdf

[5] E. Hawkes, B. An, N. M. Benbernou, H. Tanaka, S. Kim, E. D. Demaine, D. Rus, and R. J. Wood, "Programmable matter by folding," *Proceedings of the National Academy of Sciences*, vol. 107, no. 28, pp. 12 441–12 445, 2010.

[6] S. Tibbits, C. McKnelly, C. Olguin, D. Dikovsky, and S. Hirsch, "4d printing and universal transformation," pp. 539–548, 2014.

[7] K. Gilpin, A. Knaian, and D. Rus, "Robot pebbles: One centimeter modules for programmable matter through self-disassembly." in *IEEE International Conference on Robotics and Automation (ICRA)*, 3–7 May 2010, pp. 2485–2492.

[8] W. McCarthy, "Programmable matter," *Nature*, vol. 407, no. 6804, pp. 569–569, 2000.

[9] Y. Ke, L. L. Ong, W. M. Shih, and P. Yin, "Three-dimensional structures self-assembled from dna bricks," *science*, vol. 338, no. 6111, pp. 1177–1183, 2012.

[10] J.-W. Kim, J.-H. Kim, and R. Deaton, "Dna-linked nanoparticle building blocks for programmable matter," *Angewandte Chemie International Edition*, vol. 50, no. 39, pp. 9185–9190, 2011.

[11] N. Boillot, D. Dhoutaut, and J. Bourgeois, "New applications for MEMS modular robots using wireless communications," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–13, Jan. 2015, available online. Paper version to appear. [Online]. Available: http://dx.doi.org/10.1109/JSYST.2015.2427734

[12] ——, "Using nano-wireless communications in micro-robots applications," in *NANOCOM 2014, 1st ACM Int. Conf. on Nanoscale Computing and Communication*. Atlanta, Georgia, USA: ACM, May 2014, pp. 1–9.

[13] M. E. Karagozler, A. Thaker, S. C. Goldstein, and D. S. Ricketts, "Electrostatic actuation and control of micro robots using a post-processed high-voltage soi cmos chip," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2011.

[14] J. Bourgeois and S. C. Goldstein, "Distributed intelligent MEMS: Progresses and perspectives," *IEEE Systems Journal*, vol. 9, no. 3, pp. 1057–1068, Sep. 2015.

[15] J. Bourgeois and S. Goldstein, “Distributed intelligent mems: Progresses and perspectives,” in *ICT Innovations 2011*, ser. Advances in Intelligent and Soft Computing, L. Kocarev, Ed. Springer Berlin / Heidelberg, 2012, vol. 150, pp. 15–25.

[16] S. C. Goldstein, T. C. Mowry, J. D. Campbell, M. P. Ashley-Rollman, M. De Rosa, S. Funiak, J. F. Hoburg, M. E. Karagozler, B. Kirby, P. Lee, P. Pillai, J. R. Reid, D. D. Stancil, and M. P. Weller, “Beyond audio and video: Using claytronics to enable pario,” *AI Magazine*, vol. 30, no. 2, July 2009.

[17] S. C. Goldstein and T. C. Mowry, “Claytronics: A scalable basis for future robots,” in *RoboSphere 2004*, Moffett Field, CA, November 2004.

[18] S. C. Goldstein, J. D. Campbell, and T. C. Mowry, “Programmable matter,” *IEEE Computer*, vol. 38, no. 6, pp. 99–101, June 2005.

[19] S. C. Goldstein, “Brain in a bottle,” in *Wild and Crazy Ideas Session of ASPLOS*, October 2006.

[20] D. Dewey, S. S. Srinivasa, M. P. Ashley-Rollman, M. De Rosa, P. Pillai, T. C. Mowry, J. D. Campbell, and S. C. Goldstein, “Generalizing metamodules to simplify planning in modular robotic systems,” in *Proceedings of IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems IROS ’08*, Nice, France, September 2008. [Online]. Available: http://www.cs.cmu.edu/ claytronics/papers/dewey-iros08.pdf

[21] B. Piranda and J. Bourgeois, “A distributed algorithm for reconfiguration of lattice-based modular self-reconfigurable robots,” in *PDP 2016, 24th Euromicro Int. Conf. on Parallel, Distributed, and Network-Based Processing*. Heraklion Crete, Greece: IEEE, Feb. 2016, pp. 1–9.

[22] K. Stoy and R. Nagpal, “Self-Reconfiguration Using Directed Growth,” in *Distributed Autonomous Robotic Systems 6*, R. Alami, R. Chatila, and H. Asama, Eds. Springer Japan, 2007, pp. 3–12, dOI: 10.1007/978-4-431-35873-2_1.

[23] R. Fitch and Z. Butler, “Million Module March: Scalable Locomotion for Large Self-Reconfiguring Robots,” *The International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 331–343, Mar. 2008. [Online]. Available: http://ijr.sagepub.com/content/27/3-4/331

[24] Z. Butler, R. Fitch, D. Rus, and Y. Wang, “Distributed goal recognition algorithms for modular robots,” in *Robotics and Automation, 2002. Proceedings. ICRA’02. IEEE International Conference on*, vol. 1. IEEE, 2002, pp. 110–116.

[25] J. D. Foley, A. v. Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics (2nd edn in C): Principles and Practice*, Jan. 1996.

[26] A. G. Requicha, “Representations for Rigid Solids: Theory, Methods, and Systems,” *ACM Comput. Surv.*, vol. 12, no. 4, pp. 437–464, Dec. 1980.

[27] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001.

[28] C. T. Ee, S. Ratnasamy, and S. Shenker, “Practical data-centric storage.” in *NSDI*, 2006.

[29] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, “Ght: a geographic hash table for data-centric storage,” in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 78–87.

[30] Y. Zhao, Y. Chen, B. Li, and Q. Zhang, “Hop id: A virtual coordinate based routing for sparse mobile ad hoc networks,” *Mobile Computing, IEEE Transactions on*, vol. 6, no. 9, pp. 1075–1089, 2007.

[31] H. Lakhlef, J. Bourgeois, H. Mabed, and S. C. Goldstein, “Energy-aware parallel self-reconfiguration for chains microrobot networks,” *Journal of Parallel and Distributed Computing*, vol. 75, pp. 67–80, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.jpdc.2014.10.003

[32] H. Lakhlef and J. Bourgeois, “Fast and robust self-organization for micro-electro-mechanical robotic systems,” *Computer Networks*, vol. 93, pp. 141–152, 2015.

[33] M. Park, S. Chitta, A. Teichman, and M. Yim, “Automatic configuration methods in modular robots,” *International Journal for Robotics Research*, vol. 27, no. 3-4, pp. 403–421, March/April 2008.

[34] J. Barraquand and J.-C. Latombe, “Robot motion planning: A distributed representation approach,” *The International Journal of Robotics Research*, vol. 10, no. 6, pp. 628–649, 1991.

[35] F. Hou and W.-M. Shen, “Graph-based optimal reconfiguration planning for self-reconfigurable robots,” *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 1047 – 1059, 2014.