2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

# Efficient Enumeration of Modular Robot Configurations and Shapes

Kasper Stoy and David Brandt

***Abstract*— A modular robot consists of a set of mechatronic modules that can be connected in many different ways, which makes it possible to build robots of many different shapes from the same basic set of modules. The main contribution of this work is an algorithm that, given the parameters of a module and the number of modules, efficiently can calculate how many different configurations and shapes can be built. These numbers are important because the first is a measure of the self-reconfigurability and the second, given there is a relationship between form and function, the versatility of a modular robot. As an experimental contribution, we enumerate the configuration and shape spaces of square, two-dimensional modules with all possible connector configurations. We proceed to three dimensions and enumerate the spaces of the theoretically interesting sliding cube module, and the M-TRAN and SuperBot modules. Several observations are made, an important one is that the shape spaces of the two physical modules are large even for a small number of modules ($103$ different shapes using $3$ modules). This implies that it is not the lack of shape diversity that holds these modular robots back from being versatile. A result that suggests that if modules are designed right, versatility can potentially be reached even with few modules, which is contrary to the common belief in the community that more is better.**

## I. Introduction

There is a tight relationship between the shape of a robot and the tasks it can perform [13], in other words, between the form and function of a robot. Modular robots are a special kind of robots designed to take advantage of this relationship [17], [16], [12]. Modular robots are built from mechatronic modules that can be connected in different ways, which makes it possible to build robots of many different shapes from the same set of modules. Contributions in modular robotics are often specific module implementations or distributed control algorithms, in particular, self-reconfiguration algorithms, for these modules. However, the contribution of this work is at a more theoretical level. Our overarching goal is to move up from specific module implementations and understand how we from a theoretical point of view should design modular robots. To this end, we propose an algorithm for evaluating a potential module design by calculating the number of different configurations and shapes it can assume.

We calculate the number of shapes and configurations that can be built given the geometry and connector topology of a module and the number of modules. A shape is defined by the positions and orientations of its constituent modules. A configuration is defined as a shape with the addition that connectors and their gender types are also considered. The motivation for choosing these metrics is that the variety of shapes is an indication of the versatility of a modular robot and the number of configurations of its self-reconfigurability. These metrics are of course not the only factors in evaluating the versatility and self-reconfigurability of a module design, the distribution of sensors, actuators, materials, control, etc. are also important factors, but for this paper we limit ourselves to consider the potential indicated by the variety in shapes and configurations.

The main obstacle to enumerating the configuration and shape spaces is the computational and spatial complexity of enumeration given that we are faced with a combinatorial explosion. Our strategy for reducing the spacial complexity is to store only one representative for all configurations that are isomorphic. That is, configurations that with permutations, translations, rotations, or mirror operations can be made to overlap. An important question is which of a set of isomorphic configuration to use as a representative. The solution is to impose an order on configurations and only represent the smallest configuration. Asadpour et al. use a similar strategy and refer to the smallest configuration as the signature of a configuration [2], [1]. We will also call it a signature because the function is the same here even though the content and how we calculate it differ. Overall, our approach builds on the work by Golestan et al. [9]. However, where they focus on self-reconfiguration planning, we focus on enumeration. A concrete contribution is that we improve the computational complexity of calculating a configuration signature from $\mathcal{O}(n^2)$ to $\mathcal{O}(nlogn)$ where $n$ is the number of modules. A result that help us in enumeration, but also can be used to improve self-reconfiguration planning.

A complementary study to ours is the work by Brener et al. who use discrete displacement groups to categorize and compare modular robots [3]. Other groups have also used configuration space analysis to understand specific module designs either based on graphs [14], [6], [5] or matrices [7], [8]. However, we improve on this work by using configuration signatures to represent configurations, which lead to a significant reduction in time and space complexity of performing configuration enumeration. The aim of our work is also different in that we focus on shape and configuration enumeration as a general design tool and not only for analysis of existing designs.

Our experimental contribution is an enumeration of the shape spaces of all possible, nontrivial, two-dimensional square modules. We also contribute with an enumeration of the configuration space of the theoretically interesting three-dimensional sliding cube module for $n \leq 12$, and the SuperBot and M-TRAN robots for $n \leq 3$ which is low, but still allows us to compare shape spaces and assess

IT University of Copenhagen and Universal Robots, respectively. The work was done at University of Tarapacá, Chile and University of Southern Denmark `ksty@itu.dk, dbr@universal-robots.com`.

the differences between the configuration spaces of these three modules. The most important insight obtained from these experiments is that even for this low number of M-TRAN and SuperBot modules 103 different shapes can be made. This leads us to conclude that it is not the number of potential shapes that reduce the versatility of modular robots, but rather the number of *useful* shapes. Another insight is that the configuration space of the two physical modules quickly become four orders of magnitude larger than the configuration space of the sliding cube module hence making it questionable to use the sliding cube as a representative modular robot in theoretical work.

Overall, we conclude that the use of configuration enumeration is a useful tool for analyzing module designs and hence is a much needed contribution to the theoretical foundation of modular robotics.

## II. Configuration and Shape Enumeration Algorithm

Our enumeration algorithm is essentially a breath-first search algorithm where the level in the search tree corresponds to the number of modules $n$ in a configuration and the width of the tree is the number of non-isomorphic configurations. The root of the tree is the configuration signature of the module under investigation. We find the children of a parent by finding all the possible, legal ways a module can be attached to the parent configuration. A legal attachment is one where modules do not overlap and at least one connector pair match in terms of gender, position and orientation. If a child is legal, we calculate its configuration signature and check if it already known. If not, we insert it into a sorted array, one for each $n$, of configuration signatures. This process is repeated until a specified $n$ is reached. At the end of the search the lengths of the configuration lists are equal to the sizes of the configuration spaces.

In order to find the sizes of the shape spaces, we iterate through the configuration lists and remove all connectors, calculate a new signature of the shape alone, and check if this shape has been discovered before. If not, it is inserted into another sorted array, one for each $n$. At the end the lengths of these arrays are equal to the the sizes of the shape spaces.

In practice, in order to improve memory efficiency, we extract the number of configurations and shapes and delete the lists belonging to $n-1$ before moving on to $n$.

An extension to this basic algorithm is necessary in case a module has internal actuation. In this case, the starting point for the search is a list containing all the unique configuration signatures that can be generated through different actuator positions constrained to right angles (e.g. for M-TRAN the number of unique configuration signatures is 5, for SuperBot 8. See Table III).

## III. The Configuration Signature

A configuration signature is a unique representative of a set of isomorphic configurations. That is, a signature configuration is a unique representative for all configurations that can be obtained by the following operations or any combination thereof: permutation translation, rotation, and mirror. Before describing how to calculate the configuration signature we will present how to represent modules and configurations to make this calculation efficient.

### *A. Module Model*

The class of modular robots that we consider in this work is modular robots organized in a cubic lattice. A module may have a connector on each of its six sides of type male, female, unisex or none. The choice of the cubic module is based on the fact that it is a simple generalization of the square module that Liu et al. use in related work [10]. The cube is also often used for theoretical work. Finally, since we can also handle compound modules of several cubic modules we can model SuperBot and M-TRAN and other similar modules since they essentially consist of two connected cubic modules.

Modules are assumed to be able to connect if they have faces that touch each other, the gender of the connectors match, and they are rotated 0, 90, 180, and 270 degrees around the axis of connection. For our comparison with existing work we also use two-dimensional square modules in some experiments. The algorithm can be generalized to use other module geometries as long as the rotation, translation, and mirroring operations are well-defined.

### *B. Order of Modules*

We define a trivial order on modules based on their position and orientation: a module is less than another module if its x coordinate is less than the x coordinate of the other module, or if they are equal we compare the y coordinates, and if they are equal the z coordinates, and if they are all equal finally the orientation. The orientation is ordered as follows $\vec{x}, \vec{y}, \vec{z}, -\vec{x}, -\vec{y}, -\vec{z}$. In practice, the orientation is represented as an integer since there are only six possible orientations of a module in a cubic lattice.

### *C. Order of Configurations*

A configurations is an array of modules sorted using the order of modules. We also define an order of configurations based on the order of its constituent modules.

$$C_1 < C_2 \iff \forall i \in n | C_1^i < C_2^i \qquad (1)$$

Here $C_i^j$ is module $j$ in configuration $i$ and $n$ is the number of modules. Notice, that to determine if one configuration is smaller than another configuration is a linear operation on the arrays where elements are compared pairwise using the order of modules.

### *D. Calculating the Signature Configuration*

We now aim to calculate the configuration signature of a configuration represented as described above. The representation of a configuration as a sorted array of modules immediately gives us a unique representation of a set of configurations that are just permutations of each other. We then translate this configuration so that the first module coincide with the origo of the coordinate system. This removes configurations that are just translated with respect to each other.

In order to find a unique representative of a set of isomorphic configurations we eliminate configurations that are just rotations or mirrors of each other. This is done by iterating over all possible orientations and mirrors of a configuration and find the minimum configuration based on the order defined above. For the two-dimensional case there are eight isomorphic configurations for each configurations ( four ninety degree rotations times two mirror operations). For the three-dimensional case this is increased by a factor of three to twenty-four ( four ninety-degree rotations times two mirrors times three dimensions).

We now consider the computational complexity of the signature calculation since the lowering of this complexity is one of our contributions. It is clear that there is a constant maximum number of rotational isomorphic configurations given rotations are restricted to ninety degree increments. For each of these configurations we perform a rotation and/or a mirror operation which is linear, we then sort the modules in an $\mathcal{O}(nlogn)$ operation, translate the configuration so the first module is at the origo of the coordinate system, again a linear operation. Finally, compare the smallest candidate for the signature configuration with the current configuration which has worst-case linear time complexity. It can now be realized that the dominating term is the $\mathcal{O}(nlogn)$ of sorting and hence this is the bound for our signature calculation algorithm.

## IV. Configuration Enumerations

We may be interested in measuring the configuration space in several different ways to better highlight advantages and disadvantages of different module designs. The following are the metrics we calculate:

**All configurations**
: Not counting translations and permutations.

**Non-isomorphic configurations (c)**
: As above but also not counting rotations or mirrors

**Non-isomorphic shapes (s)**
: As above but also not counting configurations that have the same shape but different positions of connectors or connector genders.

The measurement *all configurations* is of practical importance in self-reconfiguration planning, because sometimes there is a need to represent configurations that only differ in terms of orientation. It may be that a chair made from a self-reconfigurable robot has fallen over and a plan needs to be made that can transform it into an upright chair. Hence both the initial configuration and the final configuration are identical except for their orientation.

The measurement *non-isomorphic configurations* is also important in self-reconfiguration planning, because in those cases where rotations of configurations are not significant the configuration space can be significantly reduced and facilitate planning as exploited by Golestan et al. [9]. These two measurements were proposed by Liu et al. [10].

| n | Non-isomorphic | Symmetric | Available | Time ($ms$) | |
|---|---|---|---|---|---|
| | conf. | conf. | conf. | $\bar{t}$ | $\sigma$ |
| 1 | 1 | 1 | 1 | 0.00 | 0.00 |
| 2 | 1 | 1 | 4 | 0.00 | 0.00 |
| 3 | 3 | 2 | 6 | 0.00 | 0.00 |
| 4 | 7 | 3 | 24 | 0.00 | 0.00 |
| 5 | 21 | 7 | 68 | 0.02 | 0.14 |
| 6 | 60 | 10 | 248 | 1.07 | 0.26 |
| 7 | 208 | 24 | 814 | 4.94 | 0.40 |
| 8 | 704 | 36 | 3216 | 19.89 | 1.05 |
| 9 | 2542 | 86 | 12164 | 81.17 | 2.35 |

TABLE I: For the Liu Module, this table shows as a function of the number of modules, how many non-isomorphic configurations exist, how many of those are symmetric, and the total available number of configurations. In addition, the mean and standard deviation of the computation time in $ms$ of one hundred experiments are given.

The measurement *non-isomorphic shape* is the measurement that is unique to our study here and is a measure of the number of shapes of a modular robot. The practical assumption is that the position of external connectors does not influence the function of the configuration. This may often be true, but sometimes connectors do provide robots with functionality (in particularly for attaching tools or augmenting the robot in different ways) and in this situation the previous measure is better.

## V. Experiments

### A. Liu Modules: Confirmation, Validation, and Performance

Liu et al. [10] describe an alternative approach to configuration enumeration based on a matrix representation. Liu et al. use a simple two-dimensional square module with female connectors on opposite faces and male connectors on the other two opposite faces. We will refer to this module as the Liu Module. Running the proposed enumeration algorithm using the Liu Module produces the result shown in Table I. Columns one to four of this table are identical to the ones published by Liu et al. ([10], Table 1, page 63). confirming their results and validating our own algorithm in the two-dimensional case.

When we consider performance Liu et al. report that it takes $56943.825s$ to enumerate the configurations of a nine-module robot, it takes our algorithm on average $0.081s$. Part of the difference can be attributed to difference in hardware since Liu et al. report using an Intel 2.00 GHz Pentium 4 CPU with 256MB RAM and we use an Intel Core i5, 1.7Ghz CPU with 4GB RAM. However, it is clear that this cannot entirely explain this difference and hence the performance also rests on our more efficient algorithm. In our implementation we do not use a multi-threaded implementation which could be used to reduce computation time even further.

### B. From Analysis to Design of Square Modules

It may appear that the number of non-isomorphic configurations is a fair representation of the number of shapes that can be created. However, in practice this may not be the case. In Figure 1, the possible non-isomorphic configurations using

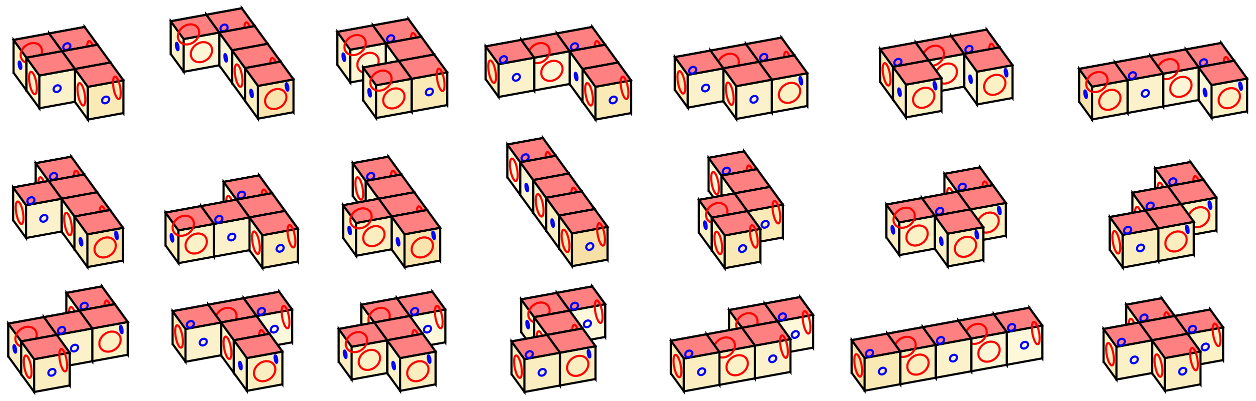

Fig. 1: The 21 possible non-isomorphic configurations using five Liu Modules. A module is represented by a box and female and male connectors are represented by a large red and a small blue circle, respectively. Note, that several configurations are isomorphic and only differ by gender of connectors (e.g. 1st and 5th configurations in the first row). If we disregard connectors there are only 12 non-isomorphic shapes.

five Liu Modules are shown, which is identical to the result reported by Liu et al. ([10], Figure 5, page 63). This figure helps us realize that many configurations are only different due to the gender and position of connectors and not module positions. Assuming that functionality is derived from the shape of the configuration and not the position of connectors we think the number of non-isomorphic shapes is a better representation of the potential functionality of a modular robot than non-isomorphic configurations. Based on this we get for the Liu Module the results of Table II column two. We can see that for the Liu Module the number of shapes is approximately half that of the number of configurations.

The motivation for our work on enumeration is to use it as a module design tool given the assumption that more available shapes lead to more functionality. Hence, a natural question is how many shapes can be made using Liu Modules compared to other similar modules? What impact does it have on the shape space to remove a connector and thus reduce the complexity of the module? What impact does it have to change the gender of a connector in case female connectors are simpler than male connectors? These relatively important questions regarding the mechanical design of an improved Liu Module can relatively quickly be answered by enumerating all combination of connectors of a square module.

However, first we would like to understand the maximum number of shapes a square module potentially can generate. In order to understand this we run the enumeration algorithm using a square module with unisex connectors on all four faces. This provides the upper limit for how many shapes can be generated using a square module. The result of this is shown in Table II, column two. A basic observation is that the number of non-isomorphic configurations is equal to the number of non-isomorphic shapes. This is due to the fact that all connectors are identical, unisex, and hence permutations of connectors do not change the configuration. The second observation and the answer we are after is that the Liu Module in fact can generate the entire shape space of a square module as can be seen by comparing columns two and three. This is an encouraging result; if only shape space is important there is no reason to use unisex connectors which are likely to be more complex than gendered connectors.

In order to find the optimal connector configuration of a square module let us find all possible non-isomorphic connector configurations except for trivial ones or symmetric ones and enumerate the corresponding shapes. The result is shown in the remaining columns of Table II. The first result is, looking at columns 2, 3, 4, and 8 that as long as there are connectors on all sides and at least one of each gender all shapes can be generated. The only change is in the number of ways these shapes can be generated. This result can be used to inform design if a male connector is simpler than a female connector. The second result is, looking at columns 6, 7, and 8, that if one connector is removed then if two connectors of identical gender are adjacent to each other the number of shapes is only reduced from 1285 to 1214 corresponding to a 5.5% reduction, hence making this an attractive design if strength of the structure is not important (all modules are only connected to one other module). The final result is trivial: using less than three connectors causes a complete break down and very few shapes can be built.

## C. Enumeration in Three Dimensions

In three dimensions we generalize the square to a cube with up to six connectors one on each side. This is an important module model because it has been used in numerous studies (e.g. [15], [4]) and hence an enumeration can shed light on the properties of this configuration space compared to that of physically implemented modules which we will consider afterwards. A cubic module with unisex connectors on all sides also serves as an upper bound for the size of the configuration and shape spaces of all cubic modules. The enumeration of the configuration and shape spaces of the cubic module is shown in column 2 of Table III. An

| | U<br>U □ U<br>U | | M<br>F □ F<br>M | | M<br>F □ F<br>F | | M<br>F □ F<br>N | | M<br>M □ F<br>N | | M<br>F □ N<br>N | | M<br>F □ M<br>F | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | Time ($ms$) | |
| $n$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $\bar{t}$ | $\sigma$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.0 | 0.0 |
| 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 4 | 1 | 4 | 1 | 2 | 1 | 0.0 | 0.0 |
| 3 | 2 | 2 | 3 | 2 | 7 | 2 | 3 | 2 | 20 | 2 | 8 | 2 | 14 | 2 | 0.0 | 0. 0 |
| 4 | 5 | 5 | 7 | 5 | 29 | 5 | 7 | 4 | 112 | 5 | 12 | 2 | 84 | 5 | 1.0 | 0.0 |
| 5 | 12 | 12 | 21 | 12 | 127 | 12 | 17 | 7 | 646 | 11 | 20 | 4 | 610 | 12 | 7.5 | 1.2 |
| 6 | 35 | 35 | 60 | 35 | 598 | 35 | 42 | 16 | 3878 | 34 | 32 | 5 | 4582 | 35 | 61.8 | 5.8 |
| 7 | 108 | 108 | 208 | 108 | 2869 | 108 | 110 | 35 | 23698 | 103 | 32 | 5 | 35828 | 108 | 647.5 | 35.4 |
| 8 | 369 | 369 | 704 | 369 | 14048 | 369 | 287 | 85 | 147155 | 353 | 52 | 9 | 285675 | 369 | 6151.6 | 243.7 |
| 9 | 1285 | 1285 | 2542 | 1285 | 69366 | 1285 | 756 | 203 | 923172 | 1214 | 84 | 12 | 2315618 | 1285 | 62390.8 | 1019.0 |

TABLE II: Enumeration of the configuration spaces for two-dimensional square modules. Specifically, non-isomorphic configurations $c$ and non-isomorphic shapes $s$ as a function of number of modules $n$ and connector morphology (connector abbrev.: F: female, M: male, U: unisex, and N: none). For the computationally hardest module the mean execution time and standard deviation of ten experiments are shown.

| | Cubic module<br>All unisex<br>connectors | M-TRAN<br>Original<br>M F<br>M □ □ F<br>M F | | | | M-TRAN<br>Unisex<br>connectors<br>U U<br>U □ □ U<br>U U | | SuperBot<br>Gendered<br>connectors<br>M F<br>M □ □ F<br>M F | | SuperBot<br>Original<br>U U<br>U □ □ U<br>U U | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Time (ms) | | | | | | | | Time (ms) | |
| $n$ | $c, s$ | $c$ | $s$ | $\bar{t}$ | $\sigma$ | $c$ | $s$ | $c$ | $s$ | $c$ | $s$ | $\bar{t}$ | $\sigma$ |
| 1 | 1 | | | | | | | | | | | | |
| 2 | 1 | 5 | 1 | 0.0 | 0.0 | 4 | 1 | 10 | 1 | 8 | 1 | 0.0 | 0.0 |
| 3 | 2 | | | | | | | | | | | | |
| 4 | 8 | 685 | 6 | 11.8 | 1.0 | 811 | 6 | 2799 | 6 | 3184 | 6 | 84.7 | 3.0 |
| 5 | 29 | | | | | | | | | | | | |
| 6 | 166 | 153929 | 103 | 3437.1 | 106.0 | 384698 | 103 | 1307878 | 103 | 3003064 | 103 | 90918.7 | 239.4 |
| 7 | 1023 | | | | | | | | | | | | |
| 8 | 6922 | 41627711 | 3482 | | | | | | | | | | |
| 9 | 48311 | | | | | | | | | | | | |
| 10 | 346543 | | | | | | | | | | | | |
| 11 | 2522522 | | | | | | | | | | | | |
| 12 | 18598427 | | | | | | | | | | | | |

TABLE III: This figure shows the number of non-isomorphic configurations $c$ and shapes $s$ as a function of type and number of modules $n$. For the original M-TRAN and SuperBot the mean execution time and std. dev. of ten experiments are shown. In the last four columns only every second row is filled because these modules consist of two cubic modules and hence an odd number of modules impossible. Also note, that for the cubic module with unisex connectors on all sides $c$ equals $s$.

astounding result is that from only 12 modules 18,598,427 different shapes can be built.

## *D. M-TRAN and SuperBot*

M-TRAN and SuperBot are two of the recent self-reconfigurable robots. One of the differences between the two is that SuperBot has a rotational joint between the two halves of the module. While this intuitively seems to expand the capabilities of the module, in particular when it comes to movement in a fixed configuration e.g. locomotion, it is unclear how this extra rotational degree of freedom influences the configuration space. Also, SuperBot has six unisex connectors and M-TRAN has three female connectors on one half-module and three male connectors on the other.

M-TRAN and SuperBot are based on the cubic model and hence we can enumerate their configuration spaces. In practice, a single module is represented as a two-module configuration. The actuators' positions are indirectly represented by inputting all non-isomorphic connector configurations that a two-module configuration can generate by any combination of actuators positions limited to right angles. In other words, actuators are not modelled and configurations that only differ in terms of positions or orientations of actuators are not counted. The enumeration is shown in Table III. Enumerating the configuration spaces of these modules is computationally expensive and could not be extended beyond three modules because the workstation runs out of physical memory and hence relies on swap-space causing the computation to slow to a crawl. We did manage to enumerate for 4 M-TRAN modules using a powerful Core i7 with 8Gb of RAM. Note, we have to keep all configurations in memory to check if generated configurations have been encountered before.

Looking at the results, the first observation is that both M-TRAN and SuperBot are able to assume the same number of shapes. However, the number is slightly less than that of a cubic module with a corresponding number of modules. This is due to the limited flexibility of the two module halves being permanently attached. The second observation is that the configuration space of SuperBot is vastly larger than that of M-TRAN. This implies that shapes can be reached in many different ways making the physical self-reconfiguration capabilities of SuperBot superior to those of M-TRAN. In columns four and five we also consider the effect of improving M-TRAN with a rotational joint and unisex connectors in isolation. The results show that the rotational joint increases the configuration space significantly more than upgrading the M-TRAN with unisex connectors. However, both play a role in giving SuperBot its larger configuration space.

## VI. Discussion & Future Work

The main contribution of this work is an algorithm that efficiently calculate configuration signatures. This allows us to enumerate the configuration spaces of a number of practically interesting modules. A common observation from these enumerations is that a large number of configurations can be obtained using a limited number of modules. It was expected that there would be an exponential relationship between the number of modules and configurations, but the astoundingly large numbers we calculated make it more clear. For instance, any three-dimensional modular robot with just ten modules can assume more shapes than we could possibly use for any practical application. The consequence of this is that it is not the number of configurations generated that is holding modular robotics back from real applications, but to make robots that can generate useful shapes. This insight - although a bit trivial - may lead us to design new modular robots that contain few modules as opposed to the traditional view that more is better. Further evidence supporting this is that the Thor heterogeneous modular robot [11] only uses few modules to obtain useful robots.

Configuration enumeration as a design tool has received little attention and hence there are many open questions. We have only looked at square and cubic modules, but it is also interesting to study other shapes to understand the relationship between module shape and configuration space better. As long as the geometry of configurations are lattice structures our algorithm can be applied with only a few modifications. However, our approach does not work for soft modular robots. In this case one has to rely on the slightly less efficient approach using configuration signatures instead as described by Asadpour et al. [1].

A final remark is that in this work we consider the number of shapes that are possible assuming that all modules are connected to the configuration with at least one connector. However, not all these configurations are practically useful because the internal physical limits of the modules, e.g. the stress on connectors arising from gravitational pull on connected modules, are not considered. In future work we plan to consider the internal physical limitations and consider what shapes can meet a given set of external requirements.

## VII. Conclusion

The main contribution of this work is an efficient algorithm for calculating configuration signatures. A configuration signature is a unique configuration that represents all isomorphic configurations. This efficiency and the fact that we can reduce our exploration to non-isomorphic configurations allow us to enumerate the configuration spaces of modular robots. In this paper we have argued that configuration enumeration is a useful design tool for modular robots and have demonstrated this in experiments with the theoretical square and cubic modules and with the existing M-TRAN and SuperBot modules. The hope is that enumeration can form part of a theoretical basis for module design.

## References

[1] M. Asadpour, M. Ashtiani, A. Sproewitz, and A. Ijspeert. Graph signature for self-reconfiguration planning of modules with symmetry. In *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5295–5300, St. Louis, MO, USA, 2009.

[2] M. Asadpour, A. Sproewitz, A. Billard, P. Dillenbourg, and A. Ijspeert. Graph signature for self-reconfiguration planning. In *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 863–869, Nice, France, 2008.

[3] N. Brener, F. Ben Amar, and Ph. Bidaud. Characterization of lattice modular robots by discrete displacement groups. In *Proceedings, IEEE/RSJ Int. Conf. on Robots and Intelligents Systems*, pages 1133–1139, Tapei, 2010.

[4] Z. Butler, K. Kotay, D. Rus, and K. Tomita. Generic de-centralized control for lattice-based self-reconfigurable robots. *The Int. Journal of Robotics Research*, 23(9):919–937, 2004.

[5] A. Casal and M. Yim. Self-reconfiguration planning for a class of modular robots. In G.T. McKee and P.S. Schenker, editors, *Proc., Sensor Fusion and Decentralized Control in Robotic Systems II*, volume 3839, pages 246–257. SPIE, 1999.

[6] A. Castano and P. Will. Representing and discovering the configuration of conro robots. In *Proc., IEEE Int. Conf. on Robotics and Automation*, volume 4, pages 3503–3509, Seoul, Korea, 2001.

[7] I.-M. Chen and J.W. Burdick. Enumerating the non-isomorphic assembly configurations of modular robotic systems. *The International Journal of Robotics Research*, 17(7):702–719, 1998.

[8] C.-J. Chiang and G.S. Chirikjian. Modular robot motion planning using similarity metrics. *Autonomous Robots*, 10(1):91–106, 2001.

[9] K. Golestan, M. Asadpour, and H. Moradi. A new graph signature calculation method based on power centrality for modular robots. In *Proceedings, 10th Int. Symposium on Distributed Autonomous Robotic Systems*, pages 505–516, Lausanne, Switzerland, 2013.

[10] J. Liu, Y. Wang, S. Ma, and Y. Li. Enumeration of the non-isomorphic configurations for a reconfigurable modular robot with square-cubic-cell modules. *International Journal of Advanced Robotic Systems*, 7(4):58–68, 2010.

[11] A. Lyder, K. Stoy, R.F.M. Garciá, J.C. Larsen, and P. Hermansen. On sub-modularization and morphological heterogeneity in modular robotics. In *Proceedings, 12th International Conference on Intelligent Autonomous Systems*, pages 1–14, Jeju Island, Korea, 2012.

[12] S. Murata and H. Kurokawa. *Self-Organizing Robots*. Springer, 2012.

[13] R. Pfeifer and C. Scheier. *Understanding Intelligence*. MIT Press, 1999.

[14] D. Rus and M. Vona. Crystalline robots: Self-reconfiguration with compressible unit modules. *Autonomous Robots*, 10(1):107–124, 2001.

[15] K. Støy. How to construct dense objects with self-reconfigurable robots. In *Proc., European Robotics Symposium (EUROS06)*, pages 27–37, Palermo, Italy, 2006.

[16] K. Stoy, D. J. Christensen, and D. Brandt. *Self-Reconfigurable Robots: An Introduction*. MIT Press, 2010.

[17] M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G.S. Chirikjian. Modular self-reconfigurable robot systems. In *IEEE Robotics & Automation Magazine*, pages 43–52, March 2007.