Robotic Enzyme-Based Autonomous Self-Replication

Michael Rubenstein, Maks Krivokon, Wei-Min Shen
Information Sciences Institute,
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292, USA
{mrubenst, krivokon}@usc.edu, shen@isi.edu

Abstract—**In this paper, we introduce and describe the notion of a robotic enzyme, and how it can use properties that are similar to biological enzymes to autonomously self-replicate. We test the idea of robotic enzymes using a virtual environment that simulates the currently existing modular robots in a physically accurate way. We describe the self-replicating features of robotic enzymes, and how they could be used to autonomously self-replicate for multiple generations, limited only by the amount of modules in the environment.**

***Keywords-** autonomous self-replication; robotic enzymes; self-reconfigurable robots.*

I. INTRODUCTION

Self-replication occurs when an entity produces an exact copy of itself. For a physical entity to self-replicate, it gathers materials from the environment, and manipulates them so that the end result is an identical copy of the original entity including both hardware and software. Sustainable self-replication occurs when multiple generations of entities reproduce until all the materials used for replication are exhausted.

Robotic self-replication allows a single robot to produce a copy of itself using material from the environment, without the assistance of humans. This ability to self-replicate will allow a group of these robots to exhibit desired qualities such as exponential growth, and fault tolerance. A major hurdle in self-replication is that the material found in the environment is in such a raw form, that there is no current robotic system that can use these raw materials to build copies of itself. Currently, robots that can self-replicate use a complex "raw" material to assemble themselves [6][5]. This allows the robots to exhibit some of the fundamental properties of self-replicating robots without solving the problem associated with the extremely raw material found in natural environments. The environmental robustness is an indicator of how many environments out of the set of all environments the robot can self-replicate. The use of complex "raw" materials decreases the environmental robustness by reducing the environments where the robot can self-replicate to a very specialized one containing the complex "raw" materials.

A direct self-replicating robot is a robot that can produce a copy of itself in one generation, and this robot can be classified into four groups. These groups, defined in [4], are divided based on certain requirements of the environment, and the number of robots that work together to replicate. Arguably, the most useful type of direct self-replicating robot is the single-robot-without fixture type. This type is capable of replicating without any external assistance from other robots or the environment

Currently, self-replicating systems compromise autonomy, which is a measure of how much outside help is required for the robot to self-replicate, and environmental robustness to achieve the goal of self-replication. A very basic self-replicating system designed by Penrose [5] consists of complex mechanical modules that are placed in a shaking container. If the modules in the container are separate from each other, then after shaking the container, they will remain separate. However, if a pair of connected modules is placed in the container with unconnected shapes, agitating the container will eventual cause the connected pair to form more connected pairs identical to the original pair. The Penrose self-replicating system sacrifices autonomy by requiring external agitation of the container, and depends on the random movement of the modules to connect them to a pair of shapes. It sacrifices environmental robustness by requiring complex modules in the environment in order to self-replicate. These sacrifices are made to allow for a completely passive self-replicating system.

A more recent self-replicating robot by Suthakorn et al. [6] attempts to self-replicate without compromising autonomy. This is accomplished using an environment that has the necessary modules, made of Legos, for replication placed at the end of drawn lines. A line following robot follows the drawn lines to the modules then follows the lines back to an assembly station. After all the modules are collected and assembled, the resulting robot is identical to the one that built it. This self-replicating system sacrifices environmental robustness and sustainability, because it requires a complicated layout of the environment and the complex "raw" materials needed.

II. ROBOTIC ENZYME

In our approach, we hope to have an autonomous self-replicating robot (similar to [6]), while increasing the environmental robustness of the replication. This is done by also using a complex "raw" material so that the difficulties associated with using naturally occurring raw materials can be avoided. A single module of a self-reconfigurable robot is considered to be our "raw" material.

These modules were designed to be the “raw” material because of their ability to easily attach to each other, and that a group of these modules can change the way they are connected. We call our approach robotic enzymes. This method is inspired by the biological world. In biology, an enzyme is often pictured as a molecule with two chemically active sites. These sites attract specific atoms or molecules, depending on the shape of the enzyme. When both sites have attracted their target atom or molecule, the enzyme “squeezes” together in such a way that the attracted atoms chemically bond to each other. The enzyme then releases the newly formed molecule, and is ready to repeat the process. Sometimes, the newly formed molecule is a copy of the enzyme itself, or a completely new enzyme. A robotic enzyme acts in a similar fashion. The robot enzyme has active sites, which it tries to attach to specific parts. These parts are strewn about the environment, and are capable of being located and attached to by the robot enzyme. After the robotic enzyme has parts attached to all of its active sites, it squeezes the parts together, assembling the parts to form a single structure. This idea of robotic enzymes can be applied to self-replication when the structure assembled by the enzyme is a copy of itself. This is demonstrated in this paper.

Our robotic enzyme is composed of two modules attached end to end to form a small snake. This dual-module design was chosen for two reasons: maneuverability, and to provide two active sites for docking module resources (one on each end). This enzyme shape is also easily formed with our CONRO self-reconfigurable robots. CONRO consists of connectable, autonomous, and self-sufficient modules. Illustrated in Figure 1, each module has one microprocessor, two motors, four docking connectors for connecting with other modules, and four pairs of infrared emitter/receivers for communicating and sensing other modules. Some modules are also equipped with other miscellaneous sensors such as tilt sensors and miniature cameras. More information about CONRO [1,2,3] and movies showing CONRO in action can be found at http://www.isi.edu/robots. The advantage of using CONRO or another type of homogenous self-reconfiguring robots to build robotic enzymes is that this allows single-robot-without-fixture [4] self-replication. This means that the robot is capable of replicating by itself without help from external structures in the environment. This increases the environmental robustness of the system. This level of autonomy is made possible by the presence on the module of both the sensors and the physical docks necessary to implement robotic enzyme based self-replication.

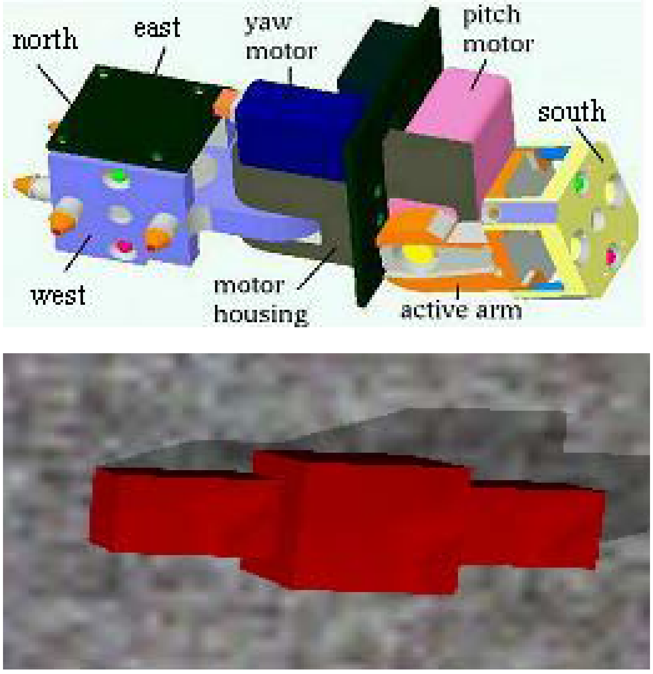


Figure 1. diagram of individual CONRO module (side view) and a simulated module (top view)

III. EXPERIMENTAL SETUP

The decision to run the replication experiments in a software simulation was made because the limited number of robotic modules in our possession would unacceptably limit the size of the replicating population. In a simulator, there was no real limiting factor in the number of modules that could be in the environment, and this allows for arbitrarily large population sizes. Our simulation framework was designed to create a reasonably accurate model of a CONRO robot in an environment that approximates real world physics. After considering several simulation environments, both commercial and open source, we decided to use Open Dynamics Engine (ODE) – an open source software library for simulation of rigid body dynamics developed by Russell Smith. Using ODE's API, we have developed a hierarchy of classes that represent the static and dynamic properties of a virtual CONRO module.

The model of the CONRO module, as shown in figure 1, was intentionally simplified in the interests of simulation efficiency. It is composed of three bodies joined by two actuators, with pitch and yaw DOF. Values for dimensions and masses of parts, maximum available force, and angular speed of servos for a module were obtained from measurements and documentation on a real module. To simplify relative distances, the simulator’s internal coordinate system was set to be in metric. Since the simulation engine has many free variables that determine the stability and accuracy of the simulation, several prototype experiments were created to determine the most realistic settings. Simulation of friction was set to a pyramid model, with contact slip in both directions. The default setting for the friction direction produced inconsistent behavior dependent on the initial orientation of the model. To fix this, the direction is dynamically set to the same orientation in the model's frame of reference.

Prior to docking with an enzyme, all the single modules only have two key properties prior to becoming a new robotic enzyme. The first property is that they be available for docking. This ability requires passive docks, which are present in both the simulated and real modules. The second important property in this self-replication process is that the enzyme must have the ability to program a module it has docked to. This is realizable in both the simulation and in the actual robots. In the simulation, this is as simple as creating a new instance of a robotic enzyme class, and then placing the new enzyme's modules under its control. In the real life robots, the programming of modules can be done using the infrared communication path that is available to all linked modules.

The simulated robotic enzymes have full knowledge of the location and orientation of all single modules in the simulated environment. The original shape of the enzyme is a snake consisting of two simulated modules shown as at the middle of Stage 1 in Figure 2. The other two single modules on the side are the "raw" material to be used for replication. When a robotic enzyme first starts to replicate itself (Stage 1 in Figure 2), it locates the closest "raw" single module available, and targets it for docking.

It then approaches the target module using a series of X, Y, and θ movements shown in Figure 3 in order to place itself in the proper orientation for docking. The X movement is for forward/backward movement along the long body axis; the Y movement is for the lateral/side movement, and the θ movement is for bending the body. Using these movements, an enzyme robot can change its location and orientation to align with a target module, and then dock to the target using a series of X movements. After docking to the target module, the result structure (one enzyme plus one raw module) is shown in Stage 2 in Figure 2.

The enzyme then searches for another raw module, locates the nearest one available, and targets it for docking. Again using a series of X, Y, and θ movements very similar to those in Figure 3, but modified for a structure consisting of 3 modules. The enzyme places itself in the proper orientation and position for docking. The enzyme then docks its free end to the target module, again using a series of X movements. After docking, the result structure is an enzyme that has captured two raw single modules at its two ends. The configuration is shown in Stage 3 in Figure 2.

At this stage, the enzyme consists of the original two modules, with two new single "raw" modules docked at both ends. The enzyme then uses the yaw degrees of freedom of all four modules to bend the enzyme into a square, as shown in Stage 4 in Figure 2. This action was chosen because it allows the original enzyme to remain intact, in contrary to what would happen if the snake of four split in the middle. This bending allows the two new added raw modules to dock on their un-occupied connectors. The two original modules then pass on their program to the two new modules. At this point, a new enzyme with hardware and software identical to the original enzyme is created. The undocking and separation between the old enzyme and the new enzyme is shown in Stage 5 and 6 in Figure 2, respectively. When the two enzymes are separated, they are both in a state identical to the initial state that the original enzyme was in when it started the replication process. Both the new and the old enzyme then repeat the replication cycle independent of each other.

One issue that is not directly addressed is collisions between enzymes. This issue is not a significant problem because the target module is chosen as the closest "raw" module available. Due to the choice in target modules, the enzymes are more likely to move away from each other than towards, thus reducing the number of collisions. If a collision does occur, the robots will not detect this, and continue with their respective movements. The shape and movements of the enzymes will allow for most collisions to be resolved automatically, not permanently incapacitating the colliding enzymes. In the rare event that a collision does occur, and both enzymes are incapacitated, it is very unlikely that these enzymes are the only ones in the environment, and therefore this will only delay the consumption of all resources, without affecting the end result significantly.

Another issue that may adversely affect the desired end result of having all enzymes consist of two modules is the case when an enzyme has three modules and it can not find a fourth module because all the single modules are taken. This is a serious issue due to the exponential growth of the number of enzymes, and the fact that this problem only appears when all the materials have been consumed. This means that very roughly half of all enzymes present will consist of three modules when no more materials exist in the environment. To solve this problem, if the enzyme has three modules, and there are no more resources, it can release the third module, and then wait a random time until finding a new target and trying to dock to it. This behavior should progressively reduce the number of enzymes that contain three modules to at most one in the end case.

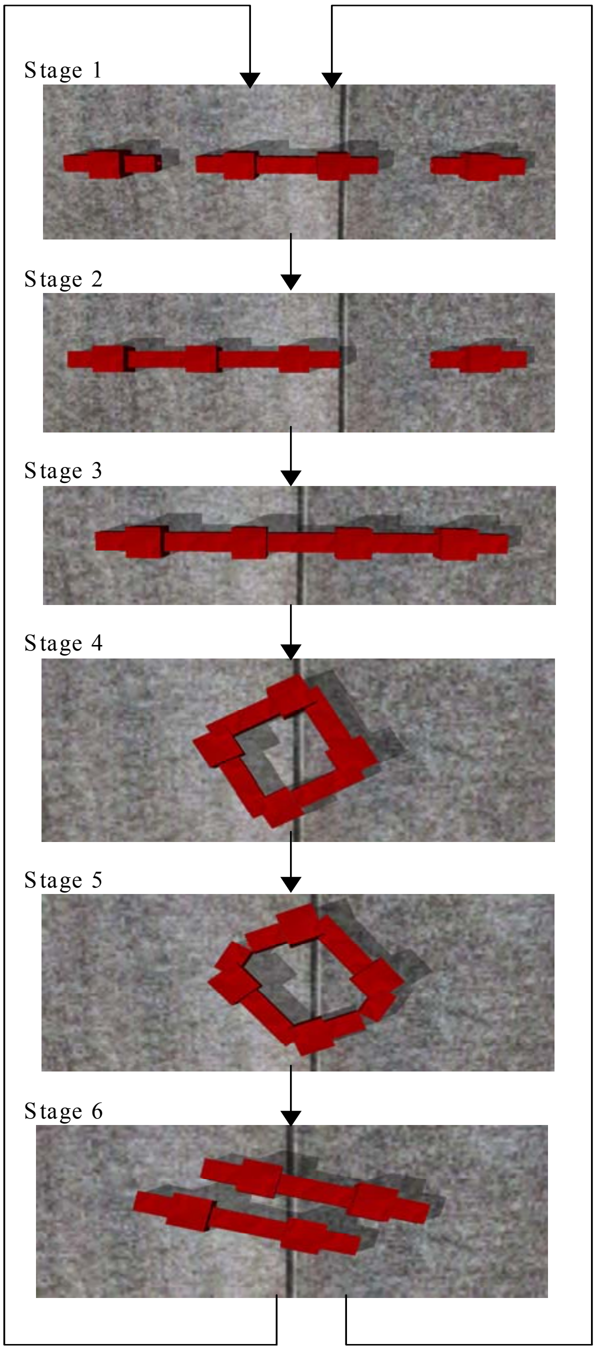


Figure 2. Robotic enzyme self-replication cycle.
Stage 1: one enzyme (two modules) and two “raw” single modules;
Stage 2: the enzyme has captured one raw module;
State 3: the enzyme has captured the second raw module;
Stage 4: the enzyme connects the two raw modules together and creates a new enzyme;
Stage 5: the old enzyme disconnects from the new enzyme;
Stage 6: both the new and the old enzyme begin the next cycle of recplication.

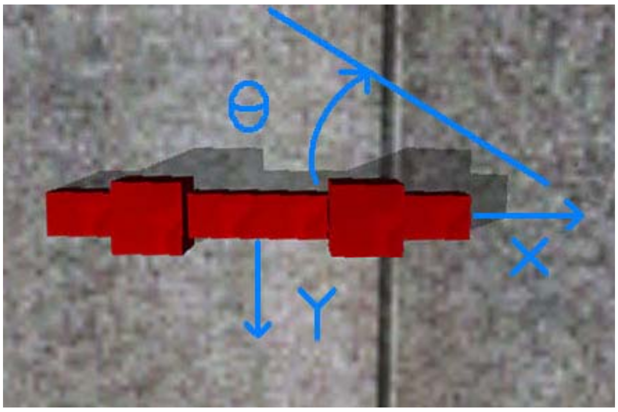
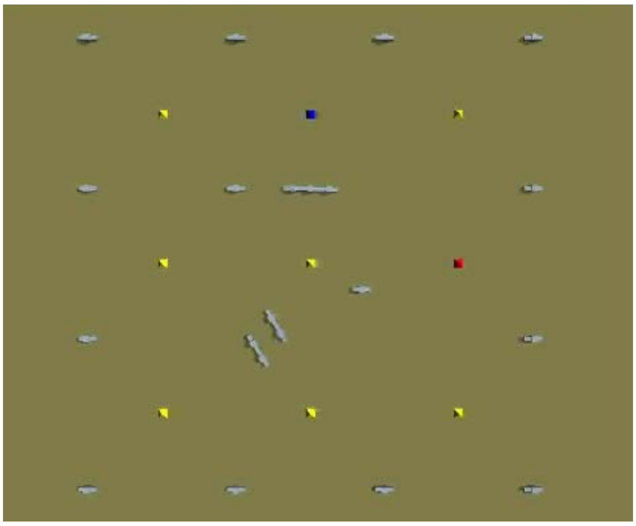


Figure 3. X, Y, and θ movmentst for robotic enzyme (shown in color)

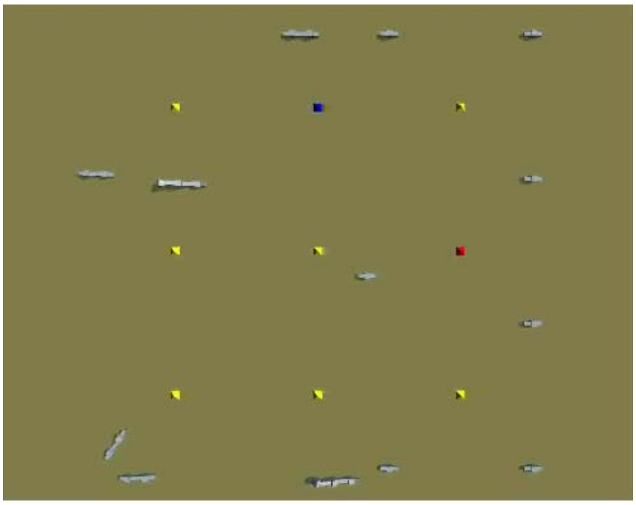
IV. RESULTS

The simulated robotic enzymes were capable of replication. Movies showing a single generation, two generations, and four generations can be viewed at http://www.isi.edu/robots. Figure 4 shows screen shots from one of the simulations. The screen shots show the initial setup, after two replication cycles, and then after three. The increase of the robotic enzyme population can be seen as the population starts at one in the first image, there are three present in the second image, and six present in the third image. The enzymes are seen in various stages of replication. The placement of the single modules in the environment can be done in an ordered grid like manner, or it can be done randomly. In either case, the robotic enzymes were able to self-replicate until all resources were depleted. Variations in initial module placement affected only the speed of replication. Initial placement in a grid pattern resulted in quicker average replication than random placement. This is because the enzymes had less alignment correction to do before docking. The simulation results show that robotic enzymes can be used for sustainable self-replication, and have a significant amount of environmental robustness.

Initial position (1 enzyme)



After two cycles (3 enzymes)



After three cycles (six enzymes)

Figure 4. Screen shots of a simulation run.

V. A GENERAL ENZYME-INSPIRED FRAMEWORK

The approach described above can be generalized into an enzyme-inspired framework for self-replication. The basic control loop of each enzyme is as follows and many enzymes can work in parallel as long as there are raw materials for them to process.

```
1. While more raw material must be collected
      {
        Search for raw material with the correct format;
        Captured (dock and connect) the raw material;
      }
2. Create a new enzyme using the collected material;
3. Pass the software to the new enzyme;
4. Disconnect from the new enzyme;
5. Go to Step 1.
```

This framework can be implemented in either simulation or on physical robots. The basic idea can be applied to many different domains that have different raw material and desired final structures. This process can also be nested into hierarchy so that some enzymes may make intermediate products, and others may use these products to create more complex structures.

Enzymes may also communicate among themselves in order to collaborate and orchestra their replication work. This communication can be chemical or electronically, and can be very similar to the hormone-inspired distributed control approaches.

Another advantage of this approach is its robustness. When there are many enzymes, the process of self-replication will succeed even if individual enzymes may fail due to unexpected reasons.

VI. FUTURE WORK

The idea of robotic enzymes has not been fully explored, and holds promise in not only self-replication, but in self-assembly. There are no major hurdles blocking the implementation of this self-replicating robotic enzyme on actual hardware. Previous work [7] has shown that small groups of modular robots are capable of docking using only sensor feedback, without knowledge of starting locations. Most of the methods necessary for implementing self-replicating robotic enzymes on hardware are known. Given the proper amount of hardware, this method can be implemented in the physical world.

VII. ACKNOWLEDGMENT

We are grateful that this research is sponsored by AFOSR under award numbers F49620-01-1-0020 and F49620-01-1-0441. We also want to thank other members in the Polymorphic Robotics Laboratory, especially Peter Will, for their generous intellectual support and discussions.

REFERENCES

[1] A. Castano, W.-M. Shen and P. Will, CONRO: Towards Deployable Robots with Inter-Robot Metamorphic Capabilities. *Autonomous Robots*, 8(3), 309-324, 2000.

[2] Shen, W.-M., B. Salemi, and P. Will, Hormone-Inspired Adaptive Communication and Distributed Control for CONRO Self-Reconfigurable Robots, *IEEE Transactions on Robotics and Automation*, 18(5), October, 2002.

[3] Castano, A., A. Behar, P. Will, The CONRO modules for reconfigurable robots, in *IEEE/ASME Trans. On Mechatronics*, 7(4) 403-09, 2002. Edited by Shen, W.-M. and M. Yim.

[4] Chirikjian, G.S., Suthakorn, J., "Towards Self-Replicating Robots", Proc. 8th Int'l Symp. Experimental Robotics, Italy, July 2002.

[5] Penrose, L.S.,"Self-reproducing machines.", *Scientific American*, 200(6):105-113, 1959.

[6] Suthakorn, J., Cushing, A.B., Chirikjian, G.S., "An Atonomous Self-Replicating Robotic System", Proc. 2003 IEEE/ASME Int'l Confrence on Advanced Intelligent Mechatronics, 2003

[7] Rubenstein, M., Payne, K., Will, P., Shen, W.M. "Docking Among Independent and Autonomous CONRO Self-Reconfigurable Robots", ICRA 2004.