Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Multimode Locomotion via SuperBot Robots

Wei-Min Shen, Maks Krivokon, Harris Chiu, Jacob Everist, Michael Rubenstein, Jagadesh Venkatesh

Abstract – **This paper presents a modular and reconfigurable robot for multiple locomotion modes based on reconfigurable modules. Each mode consists of characteristics for the environment type, speed, turning-ability, energy-efficiency, and recoverability from failures. The paper demonstrates this solution by the Superbot robot that combines advantages from MTRAN, CONRO and others. Experimental results, both in real robots and in simulation, have shown the validity of the approach and demonstrated the movements of forward, backward, turn, sidewinder, maneuver, and travel on batteries up to 500 meters on a flat terrain. In physics-based simulation, Superbot can perform as snake, caterpillar, insect, spider, rolling track, H-walker, etc., and move 1.0 meter/second on flat terrain with less than 6W/module, and climb slopes of no less 40 degrees.**

Index Terms – space robots, modular robots, self-reconfigurable and multifunctional robots.

I. Introduction

Multimode locomotion is essential for any self-sustaining robotic system. Many tasks, such as transportation, assembly, and exploration, require a robot to travel through terrains that may not be fully characterized ahead of time. In such cases, a robot must use different moving modes in different environments. For example, a robot must "climb" if it is to go up a slope, must "run" if it is to cover more distance with less energy, must "balance" if the terrain is rugged and uneven, and must "get up on feet" if it fell down by mistake. We call such an ability multimode locomotion.

To support multimode locomotion, a robot must have at least four capabilities. First, it must be able to perform different locomotion mode. Second, it must be able to recover from unexpected locomotion failures. Third, it must be able to shift from one mode to another. Finally, it must be able to choose the correct mode for the correct environment. This paper will focus on the first and the second topic, and describe how SuperBot performs many locomotion modes (run, climb, fly, jump, reach, crawl, etc.).

It is a great challenge for a single robot to achieve multimode locomotion because the robot must simultaneously satisfy two competing and even conflicting criteria: the robot must be as *general* as possible so that it can deal with many types of environments and difficulty tasks; the robot must be as *special* as possible so that it can achieve goals (such as speed and distance) with greater efficiency.

Over the past decades, it has been proven that it is very hard for a conventional robotic system to achieve both criteria, since in such systems optimization of one criterion inevitably leads to deterioration of the other. For example, improving efficiency of locomotion on particular terrain type leads to decreased efficiency on other terrain types and therefore lowers the generality of the robot.

Reconfigurable and modular robotic systems provide a new approach to this challenge and allow simultaneous optimisation on both criteria. Generality is inherent in design of such systems because modules can form different configurations. Thus, improvements in individual modules contribute to efficiency of all configurations and behaviours, while improvements in a particular configuration/behaviour do not affect negatively on other configurations/behaviours.

This paper demonstrates a case study of multimode locomotion with a reconfigurable modular robot called Superbot. The Superbot robot demonstrates a diverse set of locomotion modes, which will enable the robot to traverse in different types of environment with non-deteriorating efficiency. The advantages of Superbot modules include the integrated capability of chain-based and lattice-based robots, the accomplishment of locomotion primitives (moving and turning) from any initial configuration, and the recoverability from unexpected failures in unknown environments. It also uses a totally distributed control method for locomotion and reconfiguration that is capable of supporting arbitrary reconfiguration of modules from one mode to another.

II. Related Work

Multimode locomotion for reconfigurable robots have been studied and implemented. For example, a PolyBot rolling track can run for 500 meters on batteries and can climb stairs and fences [1]. The MTRAN robots have demonstrated various locomotion modes, including rolling track, H-walker, snake, and caterpillar [2]. The CONRO robots have also demonstrated the snake, caterpillar, insect, and spider movements [3]. For lattice-based reconfigurable robots, there also exists a semi-universal locomotion mode that can "flow" in simulation over obstacles that are comparable to the module's shape and size [4].

However, to the best of our knowledge, no systematic study of locomotion modes has been proposed. Furthermore, recovery from unexpected failures is a new research topic that has not attracted much attention. For example, Polybot movements are mostly centralized controlled and the robot cannot switch from modes without adjusting software. MTRAN is the first to combine the advantages from both lattice-based and chain-based reconfigurable robots and is very flexible for locomotion. However, its modules have fixed moving direction, and a robot cannot turn if all its modules are initially configured in the same direction. CONRO can turn and support arbitrary module reshuffling in operation, but its modules cannot bend into 180 degrees and cannot dock to their immediate neighbours. In addition, most existing robots do not have recovery mechanism. If a robot (say a rolling track) falls down, it will not be able to stand up and run again.

For lattice-based reconfigurable robots, the flow-like locomotion mode is indeed robust but yet infeasible in practice. At the present, such a mode works only when the obstacles' distribution matches the robot's lattice configuration (i.e., all obstacles must be represented as a set of "occupied" lattice grids), and it is slow and inefficient because

locomotion is accomplished as a "side effect" of reconfiguration of modules in a grid space. For example, it would require at least 2N docking/undocking operations to move forward 1-module length for a robot of N-module in a chain configuration.

III. The Design of SUPERBOT Modules

The Superbot system is designed to overcome the above difficulties. Figure 1 shows a SuperBot module that has three degree-of-freedom (pitch, yaw and roll) and that combines features from MTRAN and CONRO with a rotational DoF added to the middle shaft that joins the two cubic segments. Each SuperBot module essentially has three parts, the two ends, and a central part that rotates. The module has 3 DOF, two pitch/yaw connected by a roll. This design will allow the module to pitch and yaw for up to 180° and roll for 90° in each direction, and provide the flexibility for a single module to move and change direction.

Each SuperBot module has six connectors for the six directions in 3D (front, back, left, right, up, and down). The required features for this new connector include genderless, strong mechanical endurance, power sharing, communication, guidance, and reliability in rough environments. These connectors are homogeneous so that any connector of a module can connect to any other connector of other modules. The connectors provide the strong and accurate mechanical linkages between modules as well as the linkage for communication and power sharing. For recoverability, the connector can disconnect even if the module on one side is damaged. The connectors are strong and can endure large torques. The details of these connectors are beyond the scope of this paper and will be described elsewhere. The mechanical package will enclose on-board batteries, computers, sensors, and other electronics and will have a sealed enclosure to protect the internal from dust, sand, and moisture.

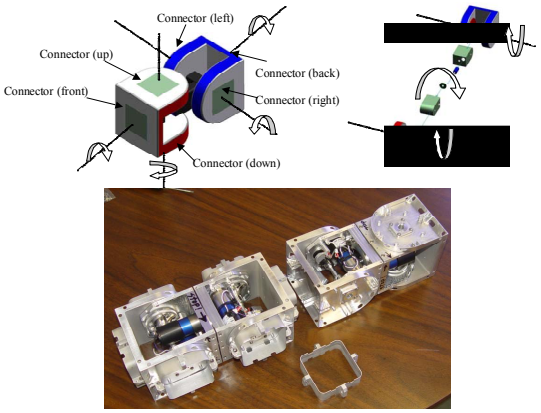

Figure 1: The design of Superbot module, and its ability to change between CONRO and MTRAN shape.

IV. The Physics-Based Simulation for Superbot

Physically accurate simulation of robotic systems provides a very efficient way of prototyping and verification of control algorithms, hardware design and exploring system deployment scenarios. It can also be used to verify feasibility of system behaviours using realistic morphology, body mass and torque specifications for servos. For the Superbot robot, we have developed a physics-based simulator called *Galina* to create modules and test environments as realistic as possible. It is built upon Open Dynamics Engine (ODE) and we have tailored it for the Superbot robot and used collision detection and rigid body dynamics algorithms. Galina also simulates the various internal and external sensors and the docking mechanism used in Superbot. Each module is simulated as a separate entity to ensure the distributed nature of the modular robots, and all the control algorithms are implemented in a distributed fashion that forces the modules to communicate and coordinate without any centralized entities.

V. Classification of Locomotion Modes

Table 1 shows a list of locomotion modes to be described in this paper. In general, these locomotion modes were hand crafted, however, other modes could be generated using another method, such as a genetic algorithm. Each mode is defined in terms of 8 parameters. The "mode" and "config" show the name and configuration of

TABLE I: Classifications of Lomotion Modes

Mode	Config	Slope	Obstacle	Speed	Turn	Energy(W/mdl)	Recover
6M	Loop	[-40,10]	0	1.0m/s	non	5.8	non
10C	Loop	[-40,10]	1/3 high	0.3m/s	yes	~5.0	yes
1M8C	H-Walker	[-10, 10]	1/1 high	0.36m/s	yes	4.35	yes
6M4C	T-Wheel	[-40,10]	0	0.6m/s	yes	~6.3	yes
2M4C	Loop	[-40,10]	0	0.70m/s	yes	~6.0	yes
8M	Climber	[-60,40]	0	0.1m/s	yes	~6.5	yes
Other	…	[…, …]	…	…	…	…	…

the mode. The "slope" and "obstacle" parameters show the type of environment that the mode can cope with. The value for slope is a range of degree of the slope that the robot can handle. For example, [-60,40] means that the mode can go down on slopes of 60 degree and climb up slopes of 40 degree. The "speed" tells how fast the mode can move, the "turn" parameter indicates if the mode can make turns or not, the "energy" parameter specifies the efficiency of the energy consumption in terms of W per module (W/mdl), and finally the "recover" parameter indicates if the mode can recover from failures or not.

For each locomotion mode, it is a challenge to optimise all the parameters simultaneously. Take for example, efficiency and adaptability two separate parameters. When optimising the efficiency parameters such as speed and energy consumption, it is hard to keep the adaptability parameters such as the range of slope and obstacle high. Using the advantage of reconfiguration, one can change the mode and select the most effective and efficient mode for the current task and environment. This way, we can avoid the impossible tradeoffs between efficiency and adaptability for a conventional robotic system and can simultaneously optimise efficiency of traversal of two different terrain types. Optimising and enhancing functionality of a Superbot module will lead to improvement in each locomotion mode – given properly designed control. Optimising control of a specific mode does not affect efficiency of other modes so that the robot can guarantee good performance on speed and energy consumption for all locomotion modes. We now describe the locomotion modes in Table 1 in detail.

A. The 6M-LOOP Mode

The first locomotion mode we consider is the 6M-Loop mode, which consists of six M-modules in a loop configuration. This mode can cope with relative flat terrain (with –10 to 10 slopes) with minimal or no obstacles. This type of environment is traditionally dominated by wheeled systems in terms of locomotion efficiency. Wheeled mode of locomotion is very energy efficient and allows achieving high speeds. However the tolerance to environment obstacles is limited by the size of the wheel and increase in wheel size leads to higher energy consumption. Thus, a wheel-actuated system usually uses obstacle avoidance instead of increased wheel size.

In order to achieve comparable locomotion performance results on this type of terrain, Superbot system has to mimic wheeled locomotion using its hyper-redundant configuration. New behaviour must be introduced for this rolling-track to accomplish a wheel-like movement. The idea is to configure the modules in a ring configuration of hexagon shape and put vertically as a rolling traveller. It rolls along vertical plane propelling by changing its shape. It keeps the rolling movement by contracting and relaxing its configuration based on gravity sensor in the modules. The six module size is chosen since more modules would make the robot more susceptible to fall over, and a configuration with less modules results in locomotion with lower speed

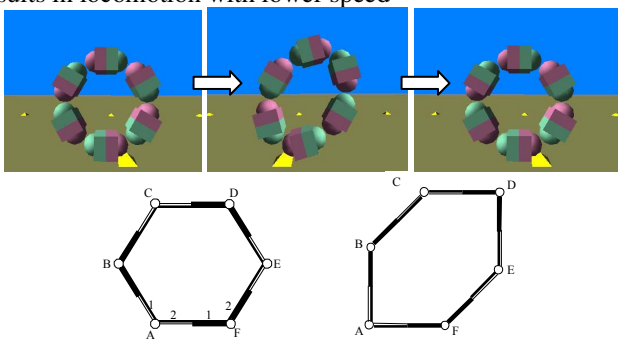

Figure 3: 6M-Loop mode for fast move with dynamic control.

Figure 3 shows a sequence of the fast rolling movement, and the implementation of the dynamic control for each step. The mode alternates its shapes between a regular hexagon (shown in the left diagram) and a deformed hexagon that tends to fall forward (shown in the right diagram). Starting from the regular hexagon, the movement is controlled by the deformation of the shape to change the centre of gravity of the traveller. There are 2 commands governing the shape transformation. One is to retain the regular hexagon shape. The other one is to let the rolling traveller to "squeeze" itself to a deformed hexagon. The deformed shape shifts the centre of gravity forward creating torque about the joint of module (module F in the right diagram) and causes the robot to roll. On the other hand, the regular round shape helps to maintain the speed of rolling.

Fast rolling movement is achieved by co-operating these two commands with gravity sensor. Each module contains a 3 DoF Gravity sensor. Each segment of module knows the vector of gravity with respect to its own coordinate frame. Initially, the traveller is set in a regular hexagon shape. The lower left module (module A in the diagram) touching the ground will check the gravity sensor of its segment 2. If it indicates the module lies flat on the surface, a "squeezing" command is issued to deform the traveller and the robot starts to roll. Command to retain regular hexagon shape is issued when the segment 1 of the lower left module (module A) knows it is vertical. The duty of determining commands based on gravity sensor is transferred anti-clockwise to the module next to it. For example, after gravity sensor of segment 1 of module A indicates the segment is vertical and the traveler transforms back to a regular hexagon shape, the lower left module will be module F, which will start to examine its gravity sensor on its segment 2 and check if it is touching the ground. The duty of determining commands continually transfers in anti-clockwise direction. In this way, the rolling traveller keeps accelerating until the servo reaches its speed limit for responding to shape changing commands. Notice that this control is dynamic and sensor based.

In terms of energy-efficiency, this locomotion mode allows fast traversal of terrains with minor obstacles. With 6 modules in a loop configuration the robot can roll with average speed of (0.93-1.04) meter/second. This was measured in simulation for a distance of 1 kilometre travelled in 18 minutes. Maximum bound of energy usage was estimated by assuming each change of servo angle uses maximum torque of 1.8Nm. Sum of angle changes was accumulated then used to calculate energy consumption of 35W for 6 modules, thus 5.83W/module in average. To travel 1 kilometre, the total energy required by the entire robot for 18 minutes is 37,776J. These energy values are good approximations, however they do not take into account some energy losses such as losses due to holding torque, and joint back driving.

B. The 10C-LOOP Mode

Although the 6M-Loop can move fast, the robot cannot stand up again once it falls down. This is due to the fact that all modules in that mode are in MTRAN-shape, and they can only move in the same plane they were configured. To overcome this limitation, the 10C-Loop mode uses all CONRO-like modules so that each module can control its pitch and yaw movement. As a result, the robot is much more flexible and can run, turn, and recover from falling down. This mode can deal with environments where obstacles do not exceed in size the height of the robot configuration.

To cope with obstacles in the environment, rolling track is particularly effective and efficient. With a flexible track, it can roll over obstacles that are comparable to the size of the robot [1]. The 10C-Loop mode demonstrates the rolling track locomotion while retaining its ability to use more efficient configuration for flat terrains. Distributed action-based control is the key to support several different locomotion modes simultaneously.

When a single rolling track is fast moving, it is a challenge to keep it balanced when there are obstacles in the environment. The most common solution is to have two tracks in parallel, but it doubles the energy consumption. For a single track to survive in an obstacle-rich environment, it must be able to turn and avoid obstacles. To the best of our knowledge, the turn capability has been lacking in all previous single rolling track movement in modular robotics.

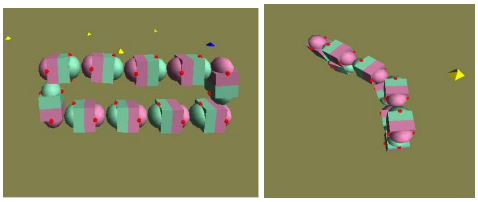

Figure 4: The implementation of 10C-Loop locomotion mode.

As shown in Figure 4, the 10C-Loop mode has 10 C-modules connected in a loop. The forward straight movement starts in the position shown in the left picture. At a fixed time interval, or when all modules have bended forward to the desired angle, each module begins to bend forward again to reach the angle that is equal to the current angle of the module that is in front of it. When this process repeats, the rolling track will move forward in a straight path. To make the robot turn, the top centre and bottom centre modules will bend sideways (shown on the right-hand side in Figure 4) while all modules maintaining the same process of described above. Because every module desires to reach the same angle positions as the current positions of the module in front of it, the entire rolling track will turn and move forward. Notice that in comparison with the dynamic control of the 6M-Loop mode, this movement is static and every module moves from one static state to another.

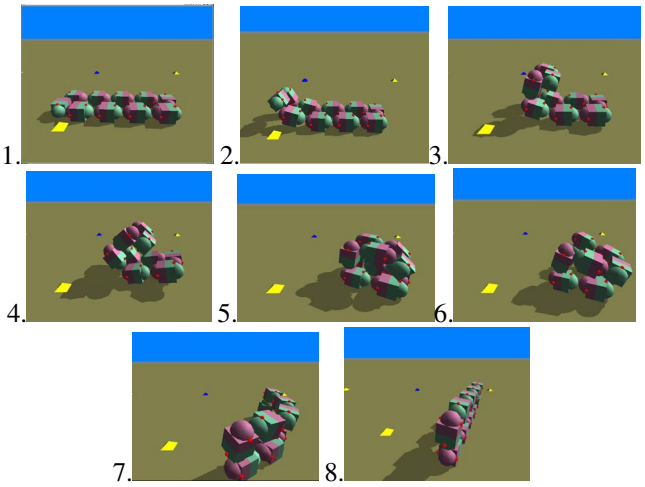

Figure 5: Recovery of 10C-Loop from a falling position.

Even with the ability to turn to avoid obstacles, there is no guarantee that the robot will never fall down in an unfamiliar and unstructured environment. Thus, the ability to recover from failure is essential. The most common locomotion fault is "tipping over" or more generally – change of robot's orientation and position relative to terrain such that interaction of locomotion gate and environment does not produce expected propelling effect. Conventional robotic systems usually use elaborate preventive strategies to avoid such faults since recovery from most of them is extremely difficult or impossible. This leads to very conservative results in terms of locomotion efficiency. For example, a Mars Rover's slow movement is partially due to this reason. However, modular reconfigurable robots have a great advantage in this respect because they can change configuration shape and recover from locomotion faults much easier than the conventional robots. This justifies the fact that a reconfigurable robot can employ more aggressive locomotion strategies for uneven and unknown terrain.

10C-Loop mode represents one fault recovery strategy for Superbot, and this strategy can be generalized for all modes that have the similar configuration. When the robot fall down, it can restore its original normal orientation by going through a sequence of motions shown in Figure 5. In step 1-4, the fallen-down loop first bends itself upward to form a folded loop in a vertical position (step 5), and then unfold this vertical loop horizontally (step 6-8) so that the loop stretched into a new rolling track. Then it can turn itself back to the original moving direction (not shown). The 10 modules are experimentally determined to be minimal for C-modules. If we use M-modules, then two M-modules must be used to achieve the effect of one C-module's pitch-yaw movement, so the total number of M-modules would be 20.

C. The 1M8C-Walker Mode

In many deployment scenarios terrain to be traversed is highly irregular and has many obstacles that are of size comparable to that of robot itself. In such environment even hybrid wheeled locomotion becomes impossible and limbed morphology has to be used to allow for "walking", "climbing" and similar gates. For specialized robotic systems with relatively monolithic design this drastic change in morphology makes impossible to use any other gates that are suitable for other types of environments described above. However modular systems, such as Superbot, allow using limbed modes of locomotion without giving up other more efficient gates, since they allow reuse of the same hardware through reconfiguration. Furthermore, distributed control allows reuse of the same software, which dynamically detects changes in configuration topology and adjusts control accordingly.

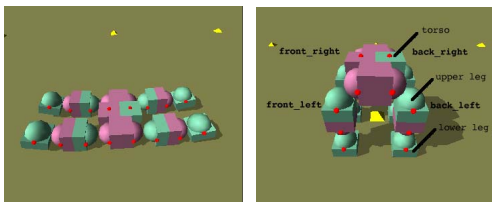

Figure 6: H-Walker configuration and naming convention.

Figure 6 We present a limbed locomotion mode – "H-Walker" that is implemented using Superbot simulated modules. The H-Walker is a 4-legged walker using two degrees of freedom on each module. It is composed of two C-modules for each leg and a single M-module representing the torso as seen in Figure 6. This locomotion design was inspired from the MTRAN experiments in automatic locomotion generation [2]. The naming convention for the modules and the control strategy are also illustrated in Figure 6. There were three possible local topologies for a Superbot module in this robot: torso, upper leg, and lower leg. Distributed locomotion control was achieved using the digital hormone method [3] based on these different topological types. Four hormones were used in this robot and each one is to control a different leg (front-left, front-right, back-left, and

back-right). The torso was responsible for sending hormone messages to each of the legs and synchronizing their coordinated actions. Each leg receives one of four hormones representing whether it is the right-front leg, left-front leg, right-back leg, or left-back leg. The torso sends a message to each leg once every locomotion period to reset and synchronize the local timers of all the leg modules.

Both the upper and lower legs based on the local topology and the hormone message they receive, reset their local timer to zero, and move the pan and yaw servos based on a sinusoid with a given a phase offset and amplitude. This way, the motions of all the legs are reasonably coordinated, so long as the hormone messages are received quickly to synchronize the clocks of all the modules.

Using this control method, the H-Walker can move forward and backward, and can turn to change its moving directions. It can reach speed of 0.6 meter/second, with average speed 0.36 meter/second. The total energy required is 39.07W for 9 modules, thus 4.35W per module in average.

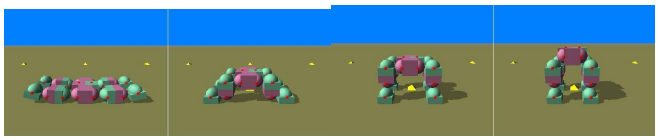

Figure 8: H-Walker Standing Sequence.

The H-Walker locomotion mode demonstrates conceptually different approach to locomotion fault recovery. Since in limbed configuration change of morphology can be costly the initial design itself should include features facilitating recovery. In H-Walker mode, it is achieved through symmetry of design and control. Because the H-Walker's topology is in the shape of an 'H', the Superbot can walk forwards and backwards using the same control strategy, as well as upside down with the legs re-positioned below the torso. In fact, this symmetry prevents the H-Walker from ever falling into any unrecoverable position because the robot does not need to flip itself over to resume walking.

Should the H-Walker fall, it is easy to achieve the relaxed position in which the legs are straightened out to the sides in a double-caterpillar shape. Then "H-Walker" proceeds to stand up using the steps as seen Figure 7. These steps perform well in simulation, and further tests are required on a physical system in various terrains.

D. The 6M4C-Training-Whell Mode

As we mentioned before, the fastest mode is the 6M-loop but it cannot turn or recover from falling. The 6M4C-training-wheel mode is a modified version of 6M so that it can run fast, and can turn and recover from falling. To do so, we add four extra legs as "training wheels" to the 6M-loop. Shown in Figure 8, one pair of "legs" is attached to the sides of the loop, perpendicular to the hexagon plane and at the opposite direction. The other pair of legs is attached in the same fashion but at the other end of the loop. To turn and change the travelling direction of the loop, the "leg" modules on one side, which is a C-module, can lower down its "feet" by rotating the pitch servo at the same time the 6M-Loop continue to roll to the other side. The 6M-Loop will be tilted and turn to the other direction. This sequence is shown in Figure 8.

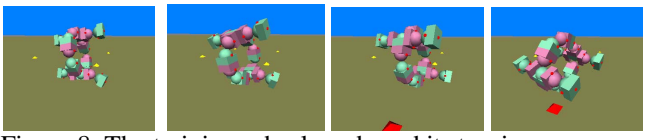

Figure 8: The training-wheel mode and its turning sequence.

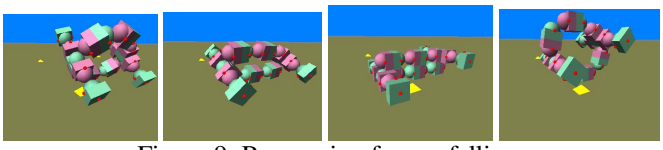

Figure 9: Recovering from a falling.

Figure 9 shows how this locomotion mode recovers from falling down. The robot first straightens all the "leg" modules and collapses the hexagon to a flat loop. The hexagon plane can then be made vertical and the flat loop will change back to its hexagon shape and continue to roll. Note that this recovery procedure is simpler than that of 10C-Loop. It is through changes of morphology original contacts between the robot and the ground. Efficiency-wise, adding the 4 extra leg modules reduces the average rolling speed down to 0.77 meter/second.

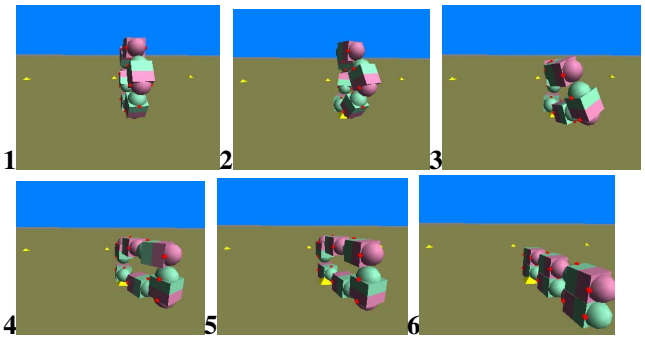

Figure 10: The 2M4C-Loop mode and its turning operation.

E. The 2M4C-LOOP Mode

The training-wheel mode uses extra modules to allow a 6M-loop to turn and recover. However, there exist other modes that can accomplish the same effect without extra modules. The 2M4C-Loop is one of such modes and it still uses 6 modules for the loop, but alternates the types of module to enable the loop to turn and recover from falling.

Figure 10 illustrates this new locomotion mode and its turning sequence. The 6 modules in the loop are arranged to be M-C-C-M-C-C. To run, this mode uses the same dynamic control as 6M-Loop but each module will only use one servo (the C-modules will use their pitch motor) for the rolling action. To turn, the hexagon of the 2M4C-Loop first bends into a rectangular shape where the C-modules are on the top and bottom of the rectangular and M-modules are at the vertical sides of the rectangular. Then, it uses the yaw servos of the C-modules to changes its direction. After that, the rectangular will go back to the original hexagon shape to run again. Note that when this turning is static and the robot is not rolling.

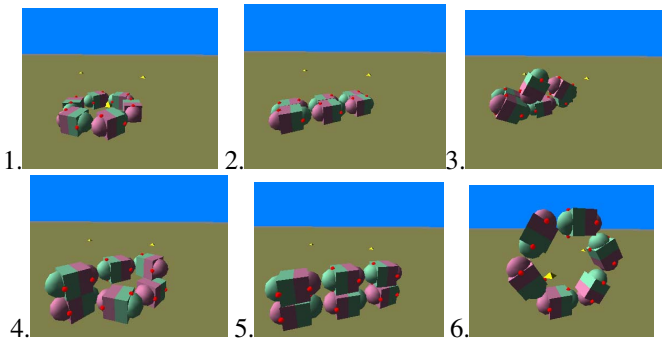

Figure 11: The recovery of 2M4C-Loop locomotion mode.

Figure 11 illustrates how the 2M4C-Loop recovers from falling. First, the loop straightens itself by bending the 2 M-modules into 180 degrees (note that this can be done only by the M-modules) and reset the shape of all 4 C-modules (step 2). The C-modules then change their yaw servos so that the robot is rising up yet unbalanced (step 3). The unusual movements of the C-modules will cause the robot to fall sideways (step 4). The loop will then straighten up again (step 5) and goes back to its original hexagon shape (step 6).

F. The 8M-Climber Mode

Previous locomotion modes show how different combinations of modules and gaits to tackle the exploration issue on a relatively flat terrain. To tackle climbing problem, a rolling track gait has been simulated to climb up a slope of 40 degrees.

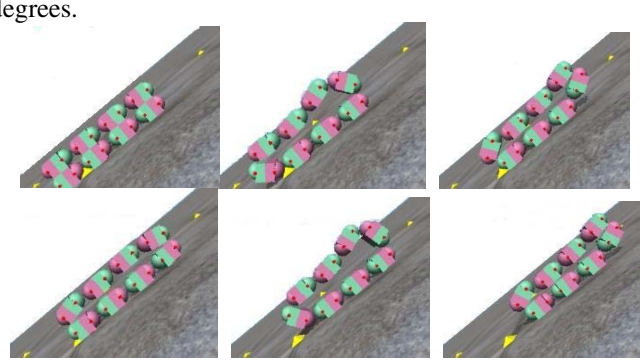

Figure 12: The 8M-Climbing Locomotion Mode.

Shown in Figure 12, the mode consists of 8 M-shape Superbot modules forming a rolling track that is only 1.5-module in height. The advantage of this configuration is to make use its low height property to stabilize it on the slope. In this simulation we assume the friction between the skin of the module and the slope is about 1.5 times higher than the rest of the modes in the paper. In reality, this friction ratio can be achieved by allowing modules to dock with some special material on their outside docking faces. The mode will climb up the slope slowly by moving module by module. The simulation shows the configuration can climb up a slope of 40 degrees straight with sufficient friction, but it also show a +/- 10 degrees of tolerance in the direction of travel before it falls sideway.

VI. Conclusions

This paper presents the concept of multimode locomotion for the Superbot robot and a list of locomotion modes. The effectiveness of these modes are demonstrated by the Superbot modules and configurations in simulation and the details of the moving, turning, recovering, and energy-efficiency of these locomotion modes are described. Our future work will be addressing the process of how to reconfigure the robot from one mode to another through self-reconfiguration and implement all these modes on Superbot. This research is supported in part by ARO W911NF-04-1-0317 and W911NF-05-1-0134, and in part by NASA NNA05CS38A. We thank Peter Will, Berok Khoshnevis, Yigal Arens, and other members in the Polymorphic Robotics Laboratory for their intellectual and moral support.

References

1. Yim, M., Roufas, K., Duff, D., Zhang, Y., Eldershaw, C., and Homans, S., "Modular Reconfigurable Robots in Space Applications", Autonomous Robots Special Issue on Space Applications Vol.14, No.2/3 March/May. 2003.

2. Kamimura, A., H. Kurokawa, E. Yoshida, K. Tomita, S. Murata, S. Kokaji, Automatic Locomotion Pattern Generation for Modular Robots, *Proceedings of IEEE International Conference on Robotics and Automation* (ICRA'03), 714-720, 2003.

3. W.-M. Shen, 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.

4. Butler, Z., K. Kotay, D. Rus and K. Tomita, Generic Decentralized Control for a Class of Self-Reconfigurable Robots, ICRA 2002.

5. Stoy, K, W.-M. Shen, P. Will, Using Role-Based Control to Produce Locomotion in Chain-type Self-Reconfigurable Robots, IEEE Transactions on Mechatronics, 7(4), 410-417, Dec. 2002.

6. Salemi B., Will P., and Shen W.-M. Distributed Task Negotiation in Modular Robots, *Robotics Society of Japan, Special Issue on "Modular Robots"*, 2003.