Autonomous Robots 8, 309–324, 2000

CONRO: Towards Deployable Robots with Inter-Robot Metamorphic Capabilities

ANDRES CASTANO, WEI-MIN SHEN AND PETER WILL
Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292
andres@isi.edu

Abstract. Metamorphic robots are modular robots that can reconfigure their shape. Such capability is desirable in tasks such as earthquake search and rescue and battlefield surveillance and scouting, where robots must go through unexpected situations and obstacles and perform tasks that are difficult for fixed-shape robots. The capabilities of the robots are determined by the design specification of their modules. In this paper, we present the design specification of a CONRO module, a small, self-sufficient and relatively homogeneous module that can be connected to other modules to form complex robots. These robots have not only the capability of changing their shape (intra-robot metamorphing) but also can split into smaller robots or merge with other robots to create a single larger robot (inter-robot metamorphing), i.e., CONRO robots can alter their shape and their size. Thus, heterogeneous robot teams can be built with homogeneous components. Furthermore, the CONRO robots can separate the reconfiguration stage from the locomotion stage, allowing the selection of configuration-dependent gaits. The locomotion and automatic inter-module docking capabilities of such robots were tested using tethered prototypes that can be reconfigured manually. We conclude the paper discussing the future work needed to fully realize the construction of these robots.

Keywords: module, reconfigurable, autonomous, self-sufficient

1. Introduction

Metamorphic robots are robots that can reconfigure their shape, i.e., intra-robot metamorphing. These robots are formed and controlled by connecting modules together and synchronizing their actions. The capabilities of the robots are thus determined by the characteristics and functionality of their modules.

Metamorphic robots are useful in applications that benefit from or require the use of robots with different topologies. Such a robot may change its shape to adapt to different tasks or environments. It could become a snake to travel down a pipe, reconfigure into a hexapod to climb a slope or into a ball to roll down a hill, or may transform a leg into a gripper to perform a grasping operation.

Metamorphic robots have been proposed by a number of robotics researchers. Fukuda and Kawauchi (1990) proposed a cellular robotic system to coordinate a set of specialized modules. Yim (1993) studied how to achieve multiple modes of locomotion using robots composed of a few basic modules. Murata et al. (1994) and Yoshida et al. (1997), separately, designed and constructed systems that can achieve planar motion by arranging modules. Pamecha et al. (1997) described metamorphic robots that can aggregate as stationary 2-D structures with varying geometry and that implement planar locomotion. Kotay et al. (1998) proposed and implemented metamorphic robots based on a module called the "robotic molecule". Nilsson (1998) designed and implemented a torsion-free joint for modular snake-like robots. Fujita et al. (1998) built a biologically inspired reconfigurable robot. Paredis and Khosla (1995) proposed modular components for building fault tolerant multipurpose robots. Neville and Sanderson (1996) proposed a module for the dynamic construction of complex structures. Farritor et al. (1996) studied the application of genetic algorithms to the generation of task-oriented heterogeneous metamorphic robots.

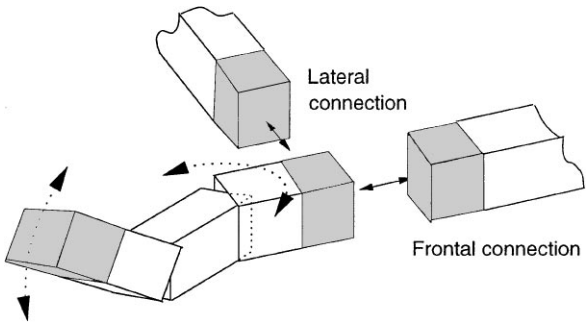


Figure 1. Basic shape of a CONRO module.

An objective of the CONRO project is to design metamorphic robots with inter-robot metamorphing capabilities, i.e., a change of shape that results in the creation or merging of individual robots. The basic shape of the CONRO modules is that of three segments connected in a chain, as shown in Fig. 1. The extremes of each module have connectors (one on each shaded face) that allow it to connect to other modules. A set of these modules can be connected together to make complex robots like the hexapod and snake shown in Fig. 2. A new robot is created when a large robot releases some of its modules for this purpose. Similarly, two independent robots merge when they link their modules to create a single larger robot.

Robots with inter-robot metamorphic capabilities are useful in applications that require robots of different sizes or tasks for which there is not a clear preference for a single large robot or a group of small robots. In these tasks, the creation of a robot requires the split of a large robot into two smaller robots. Thus, a single robot can function both as a large robot and later as a small robot by activating only one of the robots created after the split. For example, in a surveillance (or rescue) mission, a large robot is required to travel to a designated location in a reasonable amount of time. Then, this robot may create a small robot that can approach an objective without being noticed (or can step onto unsettled rubble). After the task is performed the small robot may re-attach itself to the large robot. This functionality is analogous to that of marsupial robots, in which large robots carry small robots to the locations where the small robots are needed.

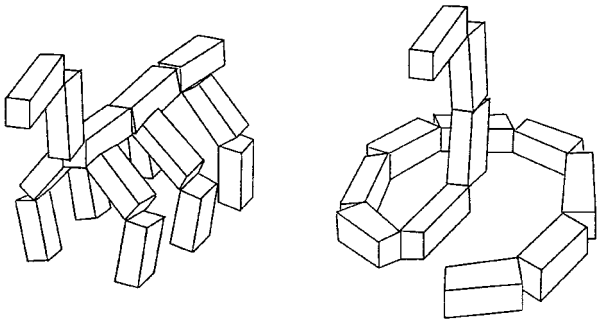


Figure 2. A hexapod and a snake CONRO robots.

Inter-robot metamorphing also allows a large robot to split into a number of small robots offering a tradeoff between the characteristics of a single large robot (e.g., high payload capacity) and those of a group of small robots (e.g., low payload capacity, able to perform simple tasks in parallel). For example, a group of small robots may explore, in parallel, a large area (e.g., the surface of a planet or a mine-field). When they finish, they might merge into a single large robot capable of carrying the tools needed to examine the discoveries made during the wide-area search.

Metamorphic robots offer a new vision where large scale results may be accomplished by the coordinated actions of a large number of small modules. Metamorphic robots with inter-robot metamorphing capabilities take this vision even further, adding variable-size and single/multiple robot capabilities to the robots, but now the system requires the coordinated action of individual robots at both the micro and macro scales. At a micro scale, the coordinated actions of the individual modules generate robot gaits and reconfigurations. At a macro scale, the coordinated actions of the individual robots generate actions of the robots as a group. The coordination of these actions, at both micro and macro level, lies in the domain of the control of distributed systems. In particular, the control of a group of CONRO robots can be seen as the control of a distributed system (the set of independent robots) where each robot is itself a distributed system (the set of independent modules). All the communication, control, actuation, and sensing capabilities that are required to make the overall system work are based on the design and capabilities of the modules.

The goal of this paper is to describe the process, experiments and results that guided the specific design of our version of a reconfigurable robot. This paper is organized as follows. Section 2 identifies and analyzes the capabilities required for a robot to be able to support both intra and inter-robot metamorphing. Section 3 describes our approach to design a module that can support these capabilities. Section 4 describes two CONRO robot prototypes used as test-beds for

actuator selection and development of gait and docking algorithms. Finally, we discuss the future work and present our conclusions in Sections 5 and 6, respectively.

2. Overview of the CONRO Robot

Depending on the situation, a CONRO robot needs to perform as a standard fixed-size fixed-shape mobile robot, as a robot with intra-robot metamorphic capabilities or as a robot with inter-robot metamorphic capabilities. At any given level, the robot must have the capabilities needed to perform at the lower levels.

At the lowest level of complexity, a CONRO robot that has adopted a given configuration must be able to perform tasks that fixed-size fixed-shape robots of the same configuration can perform (e.g., locomotion, manipulation). For example, a snake CONRO robot must be able to perform actions that a similar non-reconfigurable snake robot performs like locomotion using a traveling wave gait (e.g., Paap et al., 1996; Poi et al., 1998) or grasping objects in a plane (e.g., Chirikjian and Burdick, 1991). Likewise, a hexapod CONRO robot must be able to perform actions that fixed-shape hexapods perform such as forward, backwards and sideways locomotion as well as turns. The difficulties of the design of a modular robot that performs the actions of robots of different topologies are a reflection of the difficulties of designing the software and hardware of its modules.

Most modular robots do not have inter-robot metamorphic capabilities. An operator decides on a shape for the robot and arranges the modules by hand; at each moment, the robot performs solely as a fixed-shape robot. Non-mobile modular robots (e.g., manipulators) require modules with sensors, actuators and the signal and power networks needed to control them. The control and power can be supplied by external sources (e.g., Schonlau, 1999; Will and Grossman, 1975). Untethered mobile modular robots like CONRO cannot be connected to an external source. They must either carry their CPU and power supply (e.g., Fujita et al., 1998) or be controlled and powered using wireless technologies.

At the next level of complexity, a CONRO robot with intra-robot metamorphic capabilities must be able to rearrange its modules into different configurations. This capability allows the robot to change its shape and provides a self-repairing mechanism by allowing it to dispose of a damaged module and either replace it with a spare or reconfigure to a new shape that requires fewer modules. Unlike most reconfigurable robots that change their shape by either sliding modules along their neighbors or rotating modules about their neighbors, a CONRO robot moves its modules to non-adjacent positions by extending a chain of modules and docking the module at the extreme of the chain onto a connector of another module. The robot must have software and hardware adequate to perform this version of the peg-in-a-hole problem. This type of reconfiguration allows the robots to separate the locomotion and reconfiguration actions. CONRO robots do not need to reconfigure in order to move. Instead, they may reconfigure and then select a gait suitable for the particular configuration.

The number of possible configurations is determined, in part, by the number of connectors of the modules, as shown in Fig. 1. The connectors must be identical and symmetric to allow any module to connect to any other module. Two connectors per module (i.e., one at each extreme of the module) allow the construction of linear structures such as snakes. The construction of structures with "branches" requires the use of multiple connectors per extreme. For example, the hexapod shown in Fig. 2 can be built using modules that have one connector at one extreme and three connectors on the other because no extreme of any module of the hexapod has more than three points of contact with other modules. The maximum number of connectors per extreme for a cubic structure such as that used by our modules is five (Chen and Burdick (1995) have also considered modules with this type of connector).

At the highest level of complexity, a CONRO robot with inter-robot metamorphic capabilities must be able to split itself to create two different robots or merge with a similar robot to create a single larger robot. Therefore, the design of the modules must guarantee that a split of the robot will not lead to a brain-less or power-less robot. The merging operation places two additional requirements on the robot. First, a robot must be able to communicate remotely with another robot to request or agree on a merging operation and therefore, the robot must have some type of wireless communication. Second, to initiate the merging, a robot must be able to generate a beaconing signal to guide the other robot toward it, from a distance.

3. Design of the CONRO Modules

In this section we present our approach to design a module that can support the capabilities of the CONRO

robot described in Section 2. This approach defines the general characteristics of a module in terms of its level of self-sufficiency, autonomy and homogeneity, its size and its remote sensing capabilities. The results of our analysis call for a totally self-sufficient module, with a basic level of autonomy, a basic level of homogeneity, miniature in size and with remote sensing and communication capabilities supported by infrared (IR) transceivers.

3.1. Self-Sufficiency and Autonomy

Our use of the terms self-sufficiency and autonomy makes a distinction between the hardware and software components of a mobile robot. We use the term *self-sufficient* robot to mean a robot with the necessary on-board hardware capabilities to allow it to operate untethered. For example, a mobile robot that has an on-board CPU, power supply, sensors and actuators is a self-sufficient robot. Other examples are untethered mobile robots that are controlled by a remote host using wireless communication or are powered using wireless transmission of energy. We use the term *autonomous* robot to mean a robot (not necessarily mobile) that is able to execute tasks without having a human in the loop, i.e., a robot with the software necessary to allow it to perform some tasks automatically. The complexity of these tasks determines the level of autonomy of the robot. For example, a robot able to avoid collisions while it is being guided manually to travel to a location A has a lower level of autonomy than a robot able to travel to the location A without being guided explicitly.

With respect to the hardware, a module that supports inter-robot metamorphing must guarantee that robots created from a split operation will be self-sufficient. In the simplest split operation, a robot may split a single module. Hence, a single module qualifies as a robot and thus, the module itself must be self-sufficient: it must have its own CPU, power supply, and have control over its sensors and actuators. Of these components, the most difficult to incorporate into a module is the power supply because of its size and weight. Unfortunately, designs of battery-less self-sufficient modules (e.g., using wireless transmission of energy or solar power) would have serious power efficiency problems.

With respect to the software, the on-board processor must give the module autonomy with respect to its local operations. Because a module has exclusive control over its actuators and sensors, other modules of the robot must be able to communicate with it to either request motions or query its sensors. The module must also be autonomous with respect to reactive behaviors because there is no time for deliberations between modules due to constraints of resources such as the low bandwidth and high traffic of the network. The global control of the robot needed for deliberative tasks can be achieved using centralized control or distributed control. A local centralized control uses the CPU of a module as a master computer and the CPUs of all the other modules of the robot as slaves. A remote centralized control uses a remote host with large computational capabilities to play the role of the master module. A distributed control is based on the collaboration of agents local to each module. The design of the module makes no assumptions about the type of global control used.

3.2. Level of Homogeneity

Reconfigurable robots are classified as homogeneous or heterogeneous depending on whether or not they use a single type of module. Heterogeneous modules are designed for a specific function that determines their final positions in the robot, e.g., a limb module, a head module, a power module (Farritor et al., 1996; Fujita et al., 1998). Homogeneous modules are generic and their position in the robot determines their function. Examples of homogeneous robots are snake robots (e.g., Paap et al., 1996; Poi et al., 1998), some planar robots (e.g., Pamecha et al., 1996), and lattice-based robots (e.g., Kotay et al., 1998; Murata et al., 1994, 1998). We now discuss the costs of the level of homogeneity of a module associated with the hardware and software of both the module and the robot. These costs are summarized in Table 1.

Cost of module hardware. A homogeneous module requires a single design, manufacturing and assembly process. A set of heterogeneous modules requires many designs and a manufacturing and assembly process for each design. Usually, modular robot designers have opted for a middle ground between design/manufacturing complexity and economic/time costs where the robot is heterogeneous but the number of different modules is restricted to only two or three. In the case of the CONRO modules, they must have, at least, a basic level of homogeneity that guarantees self-sufficiency: they must have a CPU, battery, actuators, communication capabilities and connectors.

Table 1. Tradeoffs between homogeneous and heterogeneous module design.

	Homogeneous (1 type of module)	Heterogeneous (N types of modules)
Module		
Hardware	1 design/manufacturing process	N designs/manufacturing processes
Software	1 large generic program	N small specific programs
Robot		
Hardware	Large size, weight, power	Small size, weight, power
Software	Easy reconfiguration/self-repair	Complex reconfiguration/self-repair

Cost of module software. In the CONRO case, where each module is autonomous and runs a local control program, the issue is whether to design a large program for a homogeneous module or to design several small programs for specific heterogeneous modules. Since some functions of all modules overlap (e.g., status, communication) and given the need to support the hardware associated with module self-sufficiency, the code for a heterogeneous module is not necessarily significantly smaller than the code for a homogeneous module. Therefore, the cost of code development, debugging and maintenance appears to favor a single generic program as used in a homogeneous module over a number of comparable-size specialized programs as required for heterogeneous modules.

Cost of robot hardware. A small increase in module size to accommodate in a homogeneous module all the functions that a set of heterogeneous modules have, translates into a large increase in robot size, weight and power requirements. The effect of this growth can be analyzed by considering the number of modules that a given module can lift as a parameter of design. As the module grows larger, the number of modules that its actuator can lift is reduced and so is the possibility of using the module to form complex three-dimensional robots. Thus, to maximize the number of modules that a single module can lift, the module must be as small as possible. A middle ground between having a large homogeneous robot and a small heterogeneous robot is to divide the components that we would like the module to have according to whether or not they are needed to preserve its self-sufficiency. Any components not required to preserve it (e.g., GPS, antennas, etc.) will not be part of the module and will be either piggy-backed on a particular module or carried by the robot as a load.

Cost of robot software. The reconfiguration and self-repair operations need to take into account the type of modules of a robot and their positions in the current configuration. These operations are simpler to perform in a homogeneous robot, where any module can replace any other module, than in a heterogeneous robot, where specialized modules can only be replaced by identical specialized modules. The reconfiguration costs of a heterogeneous robot are also high in terms of algorithmic complexity and reconfiguration time: the reconfiguration of a heterogeneous robot is bound to require many more intermediate reconfigurations than that of a homogeneous robot.

3.3. Module Size

The size of the module is selected to increase as much as possible the ratio of actuator torque to module size. We recall the physics fundamentals involved in this ratio and relate them to a simplified version of a module, as shown in Fig. 3. We consider two modules: module A has dimensions $L \times W \times H$ and module B, a scaled version of module A, has dimensions $Lk \times Wk \times Hk$ for $k < 1$. Their volumes are

$$V_{\mathrm{A}} = LWH, \quad V_{\mathrm{B}} = V_{\mathrm{A}}k^3.$$

Let the modules have a homogeneous density δ. Then, the masses of the modules are

$$M_{\mathrm{A}} = V_{\mathrm{A}}\delta, \quad M_{\mathrm{B}} = M_{\mathrm{A}}k^3.$$

Let the modules have only one actuator located at the center of the module and let one segment of the module be fixed to the horizontal plane. We consider motions of the free segment in the horizontal and vertical planes. In the first case (Fig. 3(a)) the actuator must overcome the friction of the horizontal plane. The magnitudes of the torques needed by modules A and B to move the

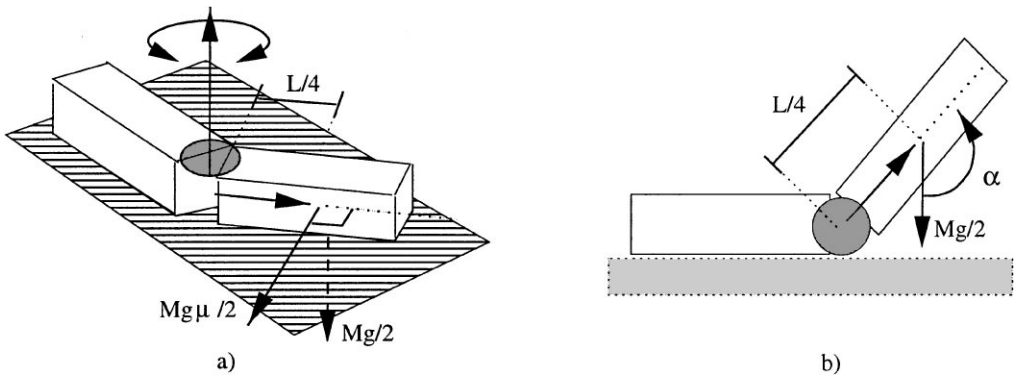


Figure 3. Actuator rotating segment of module under the presence of (a) friction, (b) gravity.

free segment are

$$\tau_A = \frac{M_A}{2} g \mu_s \frac{L}{4}, \quad \tau_B = \frac{M_A k^3}{2} g \mu_s \frac{Lk}{4} \qquad (1)$$

where μ_s is the static coefficient of friction of the horizontal plane and g is the acceleration due to gravity. In the second case (Fig. 3(b)) the actuator must overcome the force of gravity. The magnitudes of the torques needed by modules A and B to lift the free segment are

$$\tau_A = \frac{M_A}{2} g \frac{L}{4} \sin(\alpha), \quad \tau_B = \frac{M_A k^3}{2} g \frac{Lk}{4} \sin(\alpha) \qquad (2)$$

where α is the angle of pitch of the segment with respect to the vertical plane. Therefore, in both cases, the scaling of the module reduces the magnitude of the torque required to move the free segment by

$$\frac{\tau_B}{\tau_A} = k^4.$$

The scaling of the module also reduces the rotational inertia of the segment about the axis of the actuator. Approximate the segment by a thin rod. Then, the rotational inertias about the axes of the actuators of modules A and B are

$$I_A = \frac{M_A}{2} \frac{(L/2)^2}{3}, \quad I_B = \frac{M_A k^3}{2} \frac{(L/2)^2 k^2}{3}.$$

Therefore, the scaling reduces the rotational inertia by

$$\frac{I_B}{I_A} = k^5.$$

The scaling of the module is limited by the power consumed by the electronics, the time that we want the module to work, the weights and sizes of the actuators and batteries available and the number of modules that the actuator must be able to manipulate. For the following discussion we define some design parameters that will help us define a lower bound on the scale factor k. Let the average power consumed by the module be

$$\bar{P}_m = \bar{P}_e + \bar{P}_a \qquad (3)$$

where $\bar{P}_e$ and $\bar{P}_a$ are the average powers consumed by the electronics and the actuator (and its associated circuitry), respectively. Let the total weight of the module be

$$W_m = W_e + W_b + W_a \qquad (4)$$

where W_e, W_b and W_a are the weights of the electronics, the batteries and the actuator, respectively. Let t be the period of time that we want the module to be able to operate before replacing or recharging the battery. Let the electronics and actuator be rated at a voltage V. Let n be the number of modules that we want the actuator to manipulate (Fig. 4 shows the case for $n = 1$).

Two inequalities limit the scale factor k. First, the battery must have a current delivery capacity C_b greater or equal to that needed to supply the required average power $\bar{P}_m$ at the rated voltage V for a given period of time t, i.e., the capacity of the battery must satisfy

$$C_b \geq C_m = \frac{\bar{P}_m \cdot t}{V}. \qquad (5)$$

Second, the torque of the actuator τ_a must be greater than or equal to that needed to handle n modules of

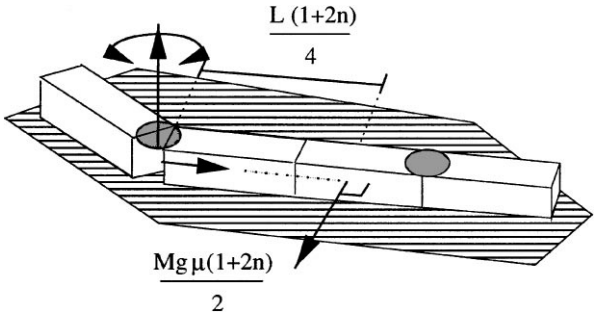


Figure 4. Actuator rotating segment of module and n additional modules ($n = 1$).

weight W_m and length L_m, i.e., the torque of the actuator must satisfy

$$\tau_\mathrm{a} \geq \tau_\mathrm{m} = \frac{M_\mathrm{m}(1+2n)}{2} g \frac{L_\mathrm{m}(1+2n)}{4} = W_\mathrm{m} L_\mathrm{m} \frac{(1+2n)^2}{8}. \quad (6)$$

This torque is the maximum torque that the actuator might need to deliver continuously and can be obtained from either Eq. (1) or (2) by setting $\mu_\mathrm{s} = 1$ or $\alpha = \pi/2$, respectively. These two inequalities restrict the possible modules to the set that can physically contain these batteries and actuators and bound the scale factor k because the module cannot be smaller that the smallest member of this set.

Given a specification of the module in terms of time of operation t and number of modules to handle n, we use Eqs. (5) and (6) to search for a combination of actuator and battery that satisfies the inequalities. The result provides the starting point for a detailed design of the module where the generalizations and simplifications can be removed.

Example. Suppose that we want to build a CONRO module with the following parameters: $V = 6$ volts, $\bar{P}_\mathrm{e} = 60$ mW, $W_\mathrm{e} = 0.01$ oz, $t = 2$ hours and $n = 4$ modules. Using Eqs. (3)–(6) we find that the capacity and torque equations that we need to satisfy are

$$C_\mathrm{b} \geq C_\mathrm{m} = \frac{(0.06 + \bar{P}_\mathrm{a})}{3} \text{ Ah}. \quad (7)$$

and

$$\tau_\mathrm{a} \geq \tau_\mathrm{m} = 10.1(0.01 + W_\mathrm{b} + W_\mathrm{a}) L_\mathrm{m} \text{ oz-in}. \quad (8)$$

Let us assume that the only batteries available are size-10 1.4-volt zinc-air hearing-aid batteries. These batteries have a capacity of 70 mAh, weight 0.012 oz and their diameter and height are 0.23 and 0.14 in, respectively. Let us assume that the only 6 volt actuators available are the MicroMo coreless DC motors described in Table 2. The columns of this table are the name of the motor and its gearhead, the maximum continuous torque of the motor, its torque constant K_τ, the reduction ratio N of the gearhead, its efficiency η for this particular ratio, the maximum continuous torque that the gearhead can provide for this particular ratio and the length L_a and weight W_a of the actuator. The torques are expressed in oz-in and the motor constant in oz-in/A. The values in the last three columns are tailored for the example and will be explained shortly.

Since we are using a combination of motor and gearhead as the actuator, we need an additional equation to verify that the motor can provide the required torque. The motor torque has to be smaller that the maximum torque that the motor can provide:

$$\max \tau_i \geq \tau_i = \frac{\tau_\mathrm{a}}{\eta N} \text{ oz-in} \quad (9)$$

where τ_a is the actuator torque, i.e., the torque at the shaft of the gearhead. Also, since the average power

Table 2. Characteristics of some MicroMo coreless DC motors and gearheads.

Actuator	Max τ_i	K_τ	N:1	η (%)	Max τ_a	L_a (in)	W_a (oz)	τ_m	τ_i	C_m (mA-h)
0816-08/1	0.02	0.467	4	90	8.5	1.0	0.22	2.9	0.8	3450
0816-08/1	0.02	0.467	1024	55	8.5	1.4	0.34	5.8	0.01	62
1016-10/1	0.07	0.426	4	90	0.7	1.0	0.44	5.15	1.43	6730
1016-10/1	0.07	0.426	16	80	2.1	1.1	0.48	6.1	0.47	2220
1016-10/1	0.07	0.426	64	70	7.6	1.2	0.51	7.0	0.15	720
1016-10/1	0.07	0.426	256	60	14.2	1.4	0.58	9.2	0.06	300
1016-10/1	0.07	0.426	1024	55	14.2	1.5	0.62	10.5	0.02	110
1016-10/1	0.07	0.426	4096	48	14.2	1.6	0.69	12.3	0.006	50

consumed by the actuator is

$$\bar{P}_{\mathrm{a}} = \frac{V\tau_i}{K_\tau\,\mathrm{W}}$$

then we can rewrite Eq. (7) as:

$$C_{\mathrm{b}} \geq C_{\mathrm{m}} = 0.02 + \frac{2\tau_i}{K_\tau}\ \mathrm{Ah}. \qquad (10)$$

Any module that satisfies Eqs. (8), (9) and (10) can be used as a starting point for the design of the module.

As an example of how this analysis may be used, consider the smallest actuator available and the smallest battery capable of providing 6 volts. We use a stack of five size-10 batteries for a total voltage of 7.0 volts, a current delivery capacity of $C_{\mathrm{b}} = 350$ mA-h, a weight of $W_{\mathrm{b}} = 0.06$ oz and a battery height of $L_{\mathrm{b}} = 0.7$ in. We assume that the voltage of the stack has a flat discharge curve around 6.0 volts. Selecting the 0816-08/1 actuator with a $4:1$ gearhead reduction ratio we find that $L_{\mathrm{a}} = 1.0$ in, $W_{\mathrm{a}} = 0.22$ oz and the maximum torques of the motor and the gearhead are 0.02 and 8.5 oz-in, respectively. Let the components of the module be organized as shown in Fig. 5. The actuator torque is transfered to the center of the module using bevel gears as shown in Fig. 5(a). Figure 5(b) shows the actuator lifting its free segment (the gears are not shown). According to our arrangement of the module, $L_{\mathrm{m}} \approx \max(L_{\mathrm{a}}, L_{\mathrm{b}}) = 1.0$ in.

The evaluation of Eqs. (8), (9) and (10) indicates whether this combination of battery and actuator satisfies the requirements of the module. The first inequality,

$$\tau_{\mathrm{a}} \geq \tau_{\mathrm{m}} = 10.1(0.01 + 0.06 + 0.22)1.0 = 2.9\ \text{oz-in}$$

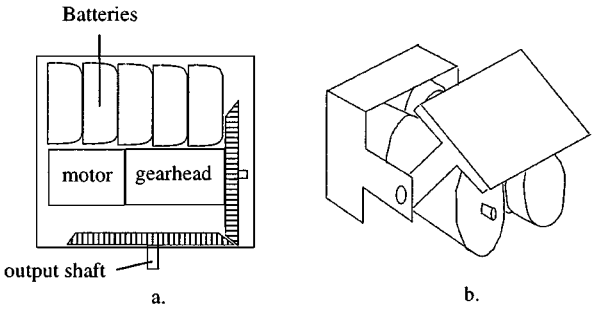


Figure 5. Top and orthographic views of a possible module (gears removed).

is satisfied because the gearhead can provide up to 8.5 oz-in of torque. The second inequality,

$$\max \tau_i \geq \tau_i = \frac{2.9}{0.9 \cdot 4} = 0.8\ \text{oz-in}$$

is not satisfied because the motor can provide only a maximum torque of 0.02 oz-in. The third inequality,

$$C_{\mathrm{b}} \geq C_{\mathrm{m}} = 0.02 + \frac{2 \cdot 0.8}{0.467} = 3450\ \text{mAh}$$

is not satisfied because the batteries have a capacity of only 350 mAh. Thus, this particular configuration would not be appropriate. These values for several other actuators are tabulated in the last three columns of Table 2. All the actuators of the table that satisfy $\max \tau_{\mathrm{a}} \geq \tau_{\mathrm{m}}$, $\max \tau_i \geq \tau_i$ and $C_{\mathrm{m}} \leq 350$ mAh have their names in boldface and could be used as a starting point for a detailed design of the module.

The data in Table 2 leads to two conclusions. First, we cannot reduce the length of the module beyond the value of L_{m} for the smallest combination that satisfies the inequalities, i.e., 1.4 in. This does not necessarily mean that we can make a real 1.4 in long module that satisfies the requirements; it means that we cannot make it smaller than 1.4 in. Second, the set of legal combinations can be reduced using secondary parameters, e.g., a constraint on the desired speed of the actuator.

3.4. Remote Sensing and Communication

A CONRO robot needs communication capabilities to transfer messages between modules and remote sensing capabilities for docking, when a module needs to move toward another module. To support inter-robot metamorphing, we need to provide for a scenario where modules might belong to two different robots. As we now show, both remote sensing and communication can be provided using an IR transmitter and receiver pair located on the connector. The IR pair serves as a device for local and remote communication and as a beacon-sensor pair for docking operations.

There is a physical path between any two modules of a robot that can be used to extend a communication network. In the simplest case, each module could pass messages according to a routing table that describes the topology of the network. In a more complex case, the modules do not have a routing table and a message needs to be broadcast over the entire net to guarantee delivery. In either case, each module uses its adjacent

modules to communicate with non-adjacent modules. Since the points of contact between any two adjacent modules are their connectors, it is reasonable to give to the connector the function of the physical communication interface. We can easily establish the communication link between adjacent modules using an IR pair on each connector.

The communication between modules of two independent robots must be remote because there is no physical path between them. A radio link allows the robots to communicate over large areas. In contrast, an IR link requires both robots to be within a line of sight, within the range of the infrared and oriented appropriately with respect to each other. For these reasons, most untethered mobile robots with wireless communication use a radio link. However, for the case of the CONRO robot, where we must handle both the intra and the inter-robot communication cases, the use of infrared-based communication is ideal because it works in both cases. Two robots that need to communicate need only to aim the connector of one of their modules toward each other to establish a link. A single IR communication system fulfills two roles that otherwise would require different systems.

A module must be able to signal its position and to guide another module during both intra and inter-robot docking. The docking operation is similar to the "peg-in-a-hole" problem: a module A at the end of a chain of modules is the peg and the hole is on module B with which we want it to dock. The operation requires module B to act as a beacon and signal its position to module A and module A to be able to sense the position of module B in order to move toward it. These functions can be carried out by an IR pair where the emitter of module B serves as a beacon, signaling its position to module A, and the receiver of module A, serves as the sensor (see Furuta and Sampei (1988) for a similar use of lasers). Hence, the same IR pair used for communication can be used as a beacon-sensor pair needed during docking.

4. Experimental Prototypes

We have built two modular robots, a snake and a hexapod, to test the actuators and docking algorithms of a CONRO robot of similar size. Their bodies have the same structure that a CONRO robot configured as a snake or hexapod would have and thus, their gaits can be ported without change. For the docking procedure we used the same IR pair that the connectors of a CONRO robot would have.

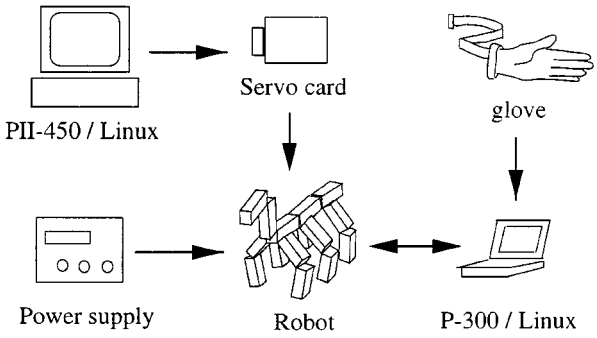


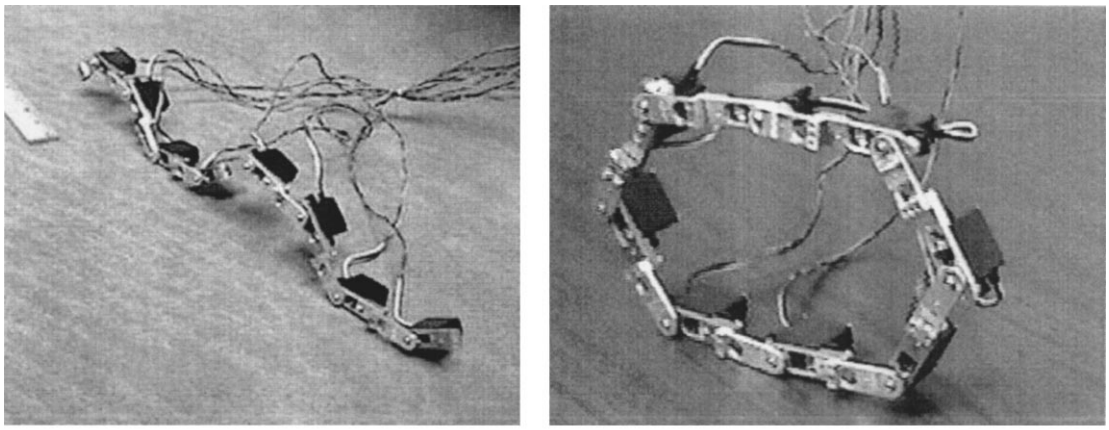
Figure 6. Prototype control setup.

The control setup of these robots is shown in Fig. 6. The robots are tethered, controlled from a host running Linux and powered by an external power supply. The actuators are RC servos driven by 8-axis servo controller cards. The control software is composed of two programs. A low-level program provides an interface to a library of functions, serial port and controller drivers, synchronization routines, a motion scheduler and a trajectory generator. A high-level program uses the interface to specify motions, their speeds or their duration. These two programs communicate through a UNIX socket so it is possible to run them on different computers to take advantage of the processor of each host or they can run on a single computer for purposes of portability. A laptop was used to display live video from the camera of the hexapod or when a user was controlling either robot directly, using a glove.

The first prototype, shown in Fig. 7 in two different configurations, is an eight degree of freedom (DOF) snake. Its weight is 250 g and its length is 515 mm. The second robot, shown in Fig. 8(a), is a hexapod with two-module chains for legs, as shown in Fig. 8(b). These chains can be reassembled manually to form a fully functional snake robot, a scaled-down version of the first prototype. The spine of the hexapod is a platform designed to carry an on-board digital camera. The weight of the hexapod is 500 g.

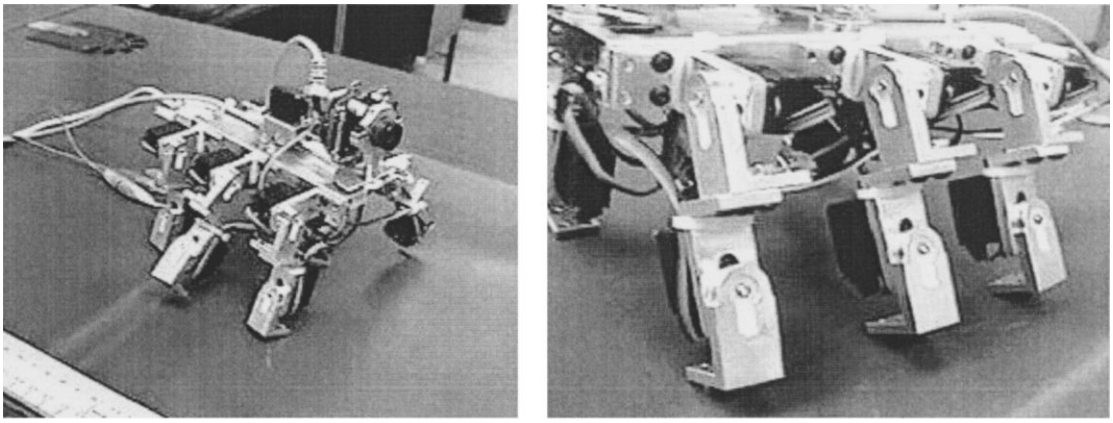
4.1. Selection of Actuator and Gaits

One of the reasons to build the robots was to verify that a given actuator was strong enough for a CONRO module. We wanted the CONRO module to be a two-actuator module, with pitch and yaw DOF, analogous to a leg of the hexapod. According to the methodology that we have outlined, we estimated that such a module would be about 10 cm long. Thus, the modules of the two prototypes were designed to be around this length.



(a) (b)

Figure 7. Snake prototype using (a) traveling wave and (b) rolling loop gaits.



(a) (b)

Figure 8. Hexapod prototype: (a) moving forwards, (b) leg detail.

The first robot that we built was the snake. Using a conservative approach, the module of the snake has a single actuator that allows the snake to move in a plane. The length of the module is 7 cm which increases the torque-to-module-weight of the actuator when compared with a 10 cm long module. The actuator, also selected conservatively, is a Naro Pro GWS RC servo with a torque of 30.55 oz-in. This actuator is strong enough to support the traveling wave and the rolling loop gaits, as shown in Fig. 7(a) and (b) (the robot was reconfigured manually). These gaits have been implemented in many other hyper-redundant planar robots (e.g., see Poi et al. (1998), Yim (1993)), and can be used by a CONRO robot configured as a snake.

The second robot that we built was the hexapod. Its leg, which has the DOF of the CONRO module that we want, is 8.8 cm long. The actuators of the legs are Cirrus CS-21 RC servos with a torque of 23 oz-in. As shown in Fig. 8(a), these actuators are strong enough to support hexapod locomotion. The forward, backward and turns implemented in the prototype can be used by a CONRO robot configured as a hexapod.

The experience acquired with these prototypes has been decisive in the design of a CONRO module that uses RC servos for actuators.

4.2. Docking

A CONRO robot uses a docking operation to reconfigure at both the intra and inter-robot levels. During intra-robot reconfiguration, the robot docks its limbs

with other parts of its body. During inter-robot reconfiguration, a limb of one of the robots docks with the body of another robot. In either case, the basic step for reconfiguration is the process used by a module to search for a designated location and dock with it.

If module A is going to dock with module B, module B must be able to signal its position to module A and module A must be able to sense this signal. As we described earlier, the beaconing and sensing elements of the module are implemented using an infrared diode (the transmitter or TX) and an infrared photo-transistor (the receiver or RX) located on the faces of the connectors that are docking.

The docking algorithm has three steps: open loop phase, closed loop phase and entrapment. The open loop step moves the modules to a position and pose where the receiver can sense the signals of the transmitter. The closed loop step moves the receiver toward the transmitter by maximizing the perceived intensity of the IR pulses. The entrapment step finalizes the docking.

As a development platform for this docking algorithm, we modified the snake to allow us to reconfigure it into a ring by docking its extremes. At both extremes of the snake we installed IR pairs that duplicate the signal-sensor capabilities of a connector of a CONRO module and magnets to simplify the entrapment phase.

The first step of the docking procedure is to configure the snake, in open loop, to a position where the sensor at one extreme of the snake is within range of the emitter at the other extreme of the snake. We aim the two extremes toward each other to increase the probability that the sensor will pick up the pulses from the transmitter. The positions of the modules {4, 5}, {2, 3, 6, 7} and {1, 8} are set to 36, 45 and 55 degrees, respectively, as shown in Fig. 9(a). The distance between the IR RX and TX after the open loop motion is finished is 10 cm. For this particular set up, where the docking is finalized with the help of magnets, a pure open loop docking is possible. However, such docking is not very useful because it does not work in the case where the modules belong to two robots or when the entrapment is not executed with magnets, i.e., uncertainties due to the control, the environment and the model of the module cause an accumulation of error that prevent a pure open loop docking.

The second step of the docking procedure is to move the extremes of the snake toward each other in closed loop. We restrict the motions to rotations of the joints of modules 1, 2 and 3, which are treated as a three-link planar manipulator. We locate the frame of reference 0 at the end-effector (the extreme of module 1), as shown in Fig. 9(a). The connector of module 8 emits light pulses that can be sensed by the receiver at the end-effector. The forward kinematics of the three-link chain can be computed as follows. Let the joint angles associated with the joints 3, 2 and 1 be θ_a, θ_b and θ_c, respectively, as shown in Fig. 9(b). Then, the homogeneous transformation that maps a vector expressed in homogeneous coordinates with respect to the end-effector frame 0 to the reference frame of the manipulator 3 is

$$ {}^3A_0 = \begin{bmatrix} \cos abc & -\sin abc & 0 & (n+m)(\cos\theta_a + \cos(\theta_a+\theta_b)) + n\,\cos abc \\ \sin abc & \cos abc & 0 & (n+m)(\sin\theta_a + \sin(\theta_a+\theta_b)) + n\,\sin abc \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

where n and m are the lengths of the segments of a module, and $abc = \theta_a + \theta_b + \theta_c$ (Paul, 1981). To specify the position and pose of the end-effector we select as variables $\{X, Y\}$, the coordinates of the end-effector, and ϕ, the angle between module 1 and the $\hat{x}$ direction, as shown in Fig. 9(b). Thus, given joint angles $\{\theta_a, \theta_b, \theta_c\}$, the coordinates of the end-effector $\{X, Y\}$ as seen from the origin of the manipulator are

$$ \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = {}^3A_0 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} $$

and the angle ϕ is

$$ \phi = \theta_a + \theta_b + \theta_c. $$

The inverse kinematics of the manipulator are determined by writing $\{\theta_a, \theta_b, \theta_c\}$ in terms of X, Y and ϕ:

$$ \theta_a = \tan^{-1}\left(\frac{C_y}{C_x}\right) - \tan^{-1}\left(\frac{\sin\theta_b}{1+\cos\theta_b}\right), $$

$$ \theta_b = \tan^{-1}\left(\pm\frac{\sqrt{1-D^2}}{D}\right) $$

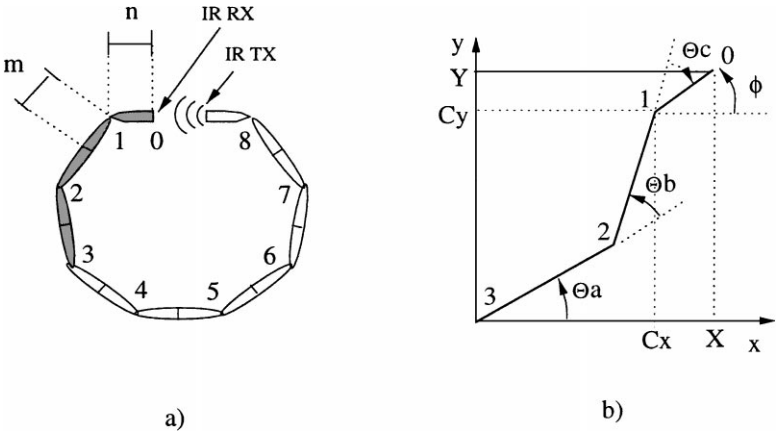


Figure 9. Three-link manipulator (a) position in snake after open loop, (b) geometric representation.

and

$$\theta_c = \phi - \theta_a - \theta_b$$

where

$$C_x = X \pm \sqrt{\frac{n^2}{1+\tan^2\phi}}, \quad C_y = Y - (X - C_x)\tan\phi,$$

$$D = \frac{C_x^2 + C_y^2}{2(n+m)^2} - 1.$$

C_x and C_y are the coordinates of joint 1.

The feedback signal for the closed loop is V_r, the voltage of the IR receiver, which is proportional to the detected irradiance. To estimate the parameters of the IR circuits, we placed the receiver and transmitter facing each other. Figure 10 shows V_r as a function of d, the distance between the emitter and receiver. The radiance of the pulses of the IR emitter are a function of the current of the emitter. This current is controlled with a resistor R_e; the lower R_e, the brighter the emitted pulse. The different curves of the graph show V_r for various values of R_e.

We can vary V_r by either increasing the radiance of the emitter or by changing the relative position of the receiver with respect to the emitter (e.g., varying d by moving the receiver toward or away from the emitter). As Fig. 10 shows, with the given set of resistors the receiver cannot sense the emitter if d is greater than

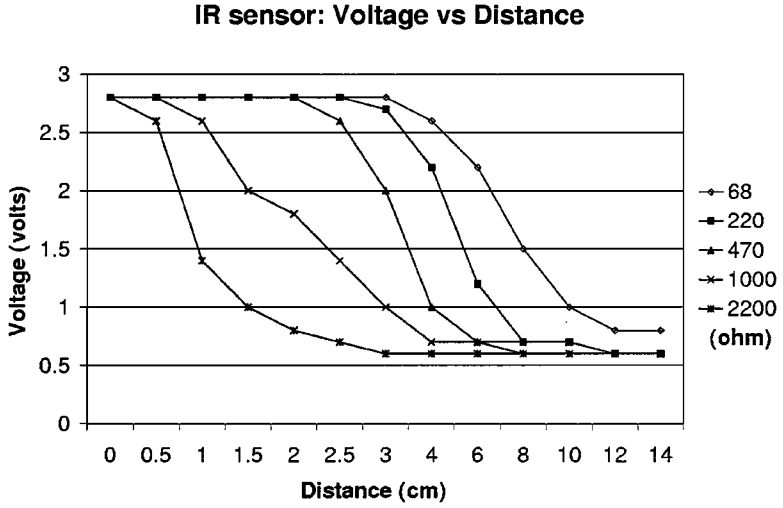


Figure 10. Infrared receiver voltage V_r versus distance between emitter and receiver d (horizontal scale is not linear).

10 cm. As we reduce d, V_r increases until the sensor saturates; further reduction of d is not reflected as an increase in V_r. For example, the voltage V_r that corresponds to $R_e = 1\ k\Omega$ allows us to observe whether we are moving toward or away from the emitter for $1.5 < d < 3$ cm. Since no single R_e allows us to observe this motion for a large range we are forced to use different resistors at different moments. If a motion saturates the sensor, we attenuate the signal of the emitter by increasing R_e. If a motion takes the sensor so far from the emitter that it cannot sense the light at all, we reduce the attenuation of the signal of the emitter by reducing R_e.

The following pseudo-code describes the closed loop algorithm:

```
CLOSED-LOOP-APPROACH ()
1  atten ← 0
2  Do forever
3    for joint ← 1 to 3
4      [Intensity, θ_joint] ← SWEEP (joint, atten)
5      if Intensity < min
6        atten ← REDUCE-ATTEN()
7        if atten < minatten
8          CARTESIAN-SWEEP()
9      else if Intensity > max
10       atten ← INCREASE-ATTEN()
11       if atten > max
12         break
```

In lines 3 and 4, we sweep the space about each joint and search for the angle θ_{joint} that maximizes V_r. In lines 5 to 8, if V_r is so low that the sensor can barely see the pulses we reduce the attenuation by reducing the value of R_e. If we cannot reduce R_e any further, we perform a vertical sweep, to recover the signal. In lines 9 to 12, if V_r is so large that the sensor has saturated and is no longer giving us any information, we increase the attenuation by increasing the value of R_e. If we cannot increase R_e any further then d is smaller than 0.5 cm and the closed loop approach is finished. Figure 11(a) shows the sweep of joints 1 and 3.

The routine CARTESIAN-SWEEP is called when the sensor has lost the location of the light. In this case, we perform a fine search using joint 1 to sweep along the vertical line. This routine is not called often and is used only after searches using the brightest pulses have failed. The pseudo-code of CARTESIAN-SWEEP is the following:

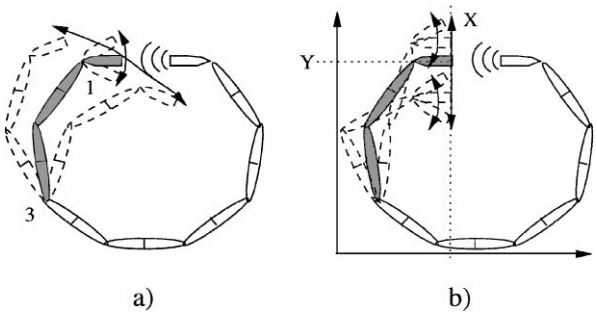


Figure 11. Closed loop sweeps (a) normal sweep of joints 1 and 3, (b) positions during Cartesian sweep.

```
CARTESIAN-SWEEP ()
1  Do forever
2    [Intensity, θ_c] ← SWEEP (1, atten)
3    if Intensity > min
4      break
5    [X, Y, φ] ← FWD-KINEMATICS (θ_a, θ_b, θ_c, 0⃗)
6    [θ_a, θ_b, θ_c] ←INV-KINEMATICS (X, Y − δY, φ)
7    MOVE-TO (θ_a, θ_b, θ_c)
```

Line 2 sweeps the space using joint 1 only. SWEEP returns the value of the maximum intensity found during the sweep and the angle at which this intensity was reached. Lines 3 and 4 determine if the light was found. In line 5, if the pulses were not found, we find the Cartesian position of the end-effector given the current angles of the manipulator. In lines 6 and 7, we move the end-effector to a position that is a distance δY away from its previous position. Figure 11(b) shows some positions of a Cartesian sweep, each of them starting at coordinates $\{X,\ Y + n\delta Y,\ \phi\}$ for n integer. The process is repeated until the light is found. A legal vertical range can be used as an exit condition to indicate that the light could not be found; in such case the docking procedure fails.

The third step of the docking procedure, entrapment, uses the fields of permanent magnets located on the connectors to finalize the docking. As shown in Fig. 10, the closed loop approach can be used to guide the receiver toward the transmitter up to a distance of 0.5 cm. At this distance, the sensor saturates and thus, the IR pair is useless; the saturation of the sensor is so fast that using other resistors does not help. However, a distance of 0.5 cm is small enough for the magnets to enter into each others' field and thus, they pull toward each other finalizing the docking. The magnets, although strong

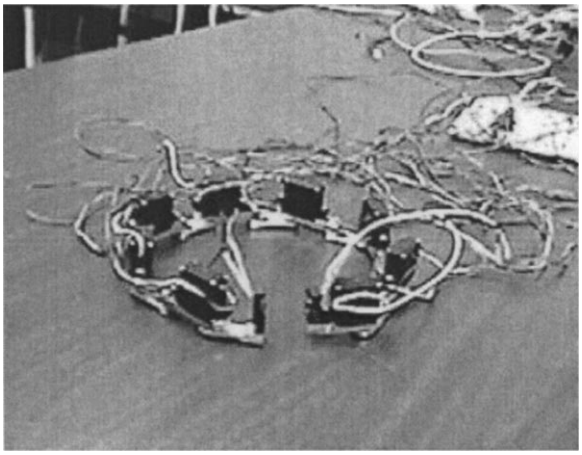


Figure 12. Snake reconfiguring into a ring.

enough to finalize the docking, cannot secure the connected modules in place. The CONRO modules will be secured instead using a locking mechanism based on a shape memory alloy actuator.

The average time to dock the extremes of the snake starting from a position where all the modules lie along a line is 45 secs. Figure 12 shows the prototype executing the docking.

The docking algorithm was developed for portability to a distributed system. Unless the communication bandwidth between the modules is large enough to allow a single module to coordinate the docking, each module involved in the docking process must be able to use its own information and cooperate on docking the chain with the target. We cannot use a conventional approach that would require the solution to the inverse kinematics of the chain every time a correction is estimated. Instead, the procedure described solves the docking problem while minimizing communication between the modules. The first step, the open loop phase, requires a master module to direct each module to a given position. Each module is informed of its goal position with a single message. The second step, the closed loop phase, decouples the motion of each joint. Each module could run the same CLOSED-LOOP-APPROACH algorithm (after removing line 3) independently of the other modules. The third step, entrapment, is independent of the software. The only moment when there is a need for message passing and coordination is when the sensor loses contact with the emitter and the Cartesian sweep routine is called. In this case, both forward and inverse kinematics need to be solved and the traffic on the network increases. However, as discussed before, the Cartesian sweep routine is seldom called.

5. Future Work

Most of the work performed on the project has been devoted to the design and construction of the modules. As the hardware matures, effort will be shifted to the software component, particularly to the areas of robot control and the dynamics of morphing. This control will be studied at the levels of centralized and distributed control.

Self-reconfigurable robots must know when a new configuration is needed, what the new shape should be, and how to transform into a new configuration. Solutions to such problems require research on integrating information from distributed sensors to form a global situation assessment, detecting and selecting a better configuration when needed, and determining a sequence of actions to perform the transformation.

In parallel with the previous work, we are developing custom-made components for the module: a miniature CMOS camera, a symmetric connector, and actuators based on shape memory alloy and solenoids.

6. Conclusion

We have introduced the design specification of CONRO, a deployable reconfigurable robot with inter-robot metamorphic capabilities. We have analyzed some necessary and desirable capabilities that a robotic module must have and introduced the CONRO modules as a possible solution to the technical challenges raised by the desired capabilities. We have pointed out that the design and implementation of feasible metamorphic robots in terms of today's technology must take into account some key concepts. Our design calls for self-sufficient modules with local autonomy, a basic level of homogeneity, miniature size and with remote sensing capabilities. The module is totally self-sufficient; it has an on-board CPU, battery, actuators and communication capabilities. The level of autonomy gives the module local and exclusive control over its resources. The basic level of homogeneity of the modules makes the design and manufacturing processes cost-effective, adds robustness to the robot in the form of redundancy and simplifies the reconfiguration process. The miniaturization of the module gives its actuator a high torque-to-module-weight ratio allowing it to manipulate a large number of identical modules. Finally, the

CONRO modules use an infrared pair for local and remote communication and for docking operations. The present design and experimental results suggests that our approach may provide a feasible solution for the realization of a new class of metamorphic robots.

Acknowledgments

The following people are involved in specific aspects of the CONRO project: Alberto Behar, Ramesh Chokkalingam, Robert Kovac, Behrokh Khoshnevis, Yi-Min Lu and Nestoras Tzartzanis. This research is being sponsored by DARPA/MTO under contract number DAAN02-98-C-4032.

References

Chen, I.-M. and Burdick, J. 1995. Determining task optimal modular robot assembly configurations. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 132–137.

Chirikjian, G. and Burdick, J. 1991. Kinematics of hyper-redundant robot locomotion with applications to grasping. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 720–725.

Farritor, S., Dubowsky, S., and Rutman, N. 1996. On the design of rapidly deployable field robotic systems. In *ASME Design Engineering Technical Conference*.

Fujita, M., Kitano, H., and Kageyama, K. 1998. Reconfigurable physical agents. In *Proc. Int. Conf. Autonomous Agents*, pp. 54–61.

Fukuda, T. and Kawauchi, Y. 1990. Cellular robotic system (CEBOT) as one of the realization of self-organizing intelligent universal manipulator. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 662–667.

Furuta, K. and Sampei, M. 1988. Path control of a three-dimensional linear motional mechanical system using laser. *IEEE Trans. Industrial Electronics*, 35(1):52–59.

Kotay, K., Rus, D., Vona, M., and McGray, C. 1998. The self-reconfiguring robotic molecule. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 424–431.

Murata, S., Kurokawa, H., and Kokaji, S. 1994. Self-assembling machine. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 441–448.

Murata, S., Kurokawa, H., Yoshida, E., Tomita, K., and Kokaji, S. 1998. A 3-D self-reconfigurable structure. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 432–439.

Neville, B. and Sanderson, A. 1996. Tetrabot family tree: Modular synthesis of kinematic structures for parallel robotics. In *Proc. IEEE Int. Symposium on Robotics Research*, pp. 382–390.

Nilsson, M. 1998. Why snake robots need torsion-free joints and how to design them. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 412–417.

Paap, K., Dehlwisch, M., and Klaassen, B. 1996. GMD-Snake: A semiautonomous snake-like robot. *Distributed Autonomous Robotic Systems 2*, Springer-Verlag: Tokio.

Pamecha, A., Chiang, C.-J., Stein, D., and Chirikjian, G. 1996. Design and implementation of metamorphic robots. In *Proc. 1996 ASME Design Engineering Tech. Conf. and Comput. in Engineering Conf.*, pp. 1–10.

Pamecha, A., Ebert-Uphoff, I., and Chirikjian, G. 1997. Useful metrics for modular robot motion planning. *IEEE Trans. Robotics Automat.*, 13(4):510–521.

Paredis, C. and Khosla, P. 1995. Design of modular fault tolerant manipulators. In *Proc. First Workshop Algorithmic Foundations of Robotics*, pp. 371–383.

Paul, R. 1981. *Robot Manipulators: Mathematics, Programming and Control*, MIT Press: Cambridge, MA.

Poi, G., Scarabeo, C., and Allota, B. 1998. Traveling wave locomotion hyper-redundant mobile robot. In *Proc. IEEE Int. Conf. Robotics Automat.*, pp. 418–423.

Schonlau, W.J. 1999. MMS: A modular robotic system and model based control architecture. In *Proc. 1999 SPIE Conf. on Sensor Fusion and Decentralized Control in Robotic Systems II*, pp. 289–296.

Will, P. and Grossman, D. 1975. An experimental system for computer controlled mechanical assembly. *IEEE Trans. Computers*, C-24(9):879–888.

Yim, M. 1993. A reconfigurable modular robot with multiple modes of locomotion. In *Proc. JSME Conference on Advanced Mechatronics*.

Yim, M. 1994. Locomotion with a unit-modular reconfigurable robot. PhD Thesis, Stanford University Dept. of Mechanical Engineering.

Yoshida, E., Murata, S., Tomita, K., Kurokawa, H., and Kokaji, S. 1997. Distributed formation control of a modular mechanical system. In *Proc. Int. Conf. Intelligent Robots and Systems*, pp. 1090–1097.

Andres Castano received the B.S. degree from the University of Los Andes, Bogota, Colombia, and the M.S. and Ph.D. degrees from the University of Illinois at Urbana-Champaign, all in Electrical Engineering. At Urbana-Champaign he worked at the Beckman Institute in visual servo-control of robotic manipulators and very large depth-of-field panoramic cameras. In 1998 he joined the Information Sciences Institute at the University of Southern California where he is currently the lead roboticist of the Conro project. His research interests are robotics, computer vision and graph theory.

Wei-Min Shen is a Project Leader at the Information Sciences Institute and a Research Assistant Professor of computer science at the University of Southern California. He received his Ph.D. from Carnegie-Mellon University in 1989 and is the author of

"Autonomous Learning from the Environment." His current research interests include self-reconfigurable robots, multi-agent organizational learning, and data mining. He is the recipient of a 1996 AAAI Robotics Competition Silver-Medal Award, and a 1997 RoboCup World Championship Award.

Peter M. Will is the Director of the Distributed Scalable Systems Division at USC/Information Sciences Institute, a Research Professor in Industrial and Systems Engineering at USC, and has over 35 years research experience in industry. Prior to joining USC, Will spent 16 years at IBM's Yorktown Research Lab, 7 years with Schlumberger, and 5 years at Hewlett-Packard Labs. Will has over 50 publications and 10 patents, and for six years he was a member of the ISAT group working with DARPA. In 1990, he was awarded the International Engelberger Prize in robotics. He received a B.Sc. degree in Electrical Engineering and a Ph.D. in Non-linear Control Systems from the University of Aberdeen.