BONUS CENTERFOLD: COG

ROBOT
science & technology

OUR FIFTH ISSUE • APRIL 1999

Is a RoboPe
in Your Near Future

Build & Program Your
Own Hexapod

Can You
Skin a Robot?

0 4>

0 74470 93568 1

Grand Prize 1997 and 1998 SME National Student Robot Competitions

RAVE USER REVIEWS!

"The most user-friendly equipment and learning info for students that I have ever used."
Dennis Skurulsky, President, Wisconsin Technology Education Association

"Our students have really enjoyed working with your kit. It has given our small-town, very rural students a chance to see technology from a new perspective."
Larry White, Technology Coordinator/Teacher, Sterling City High School, Sterling City, Texas

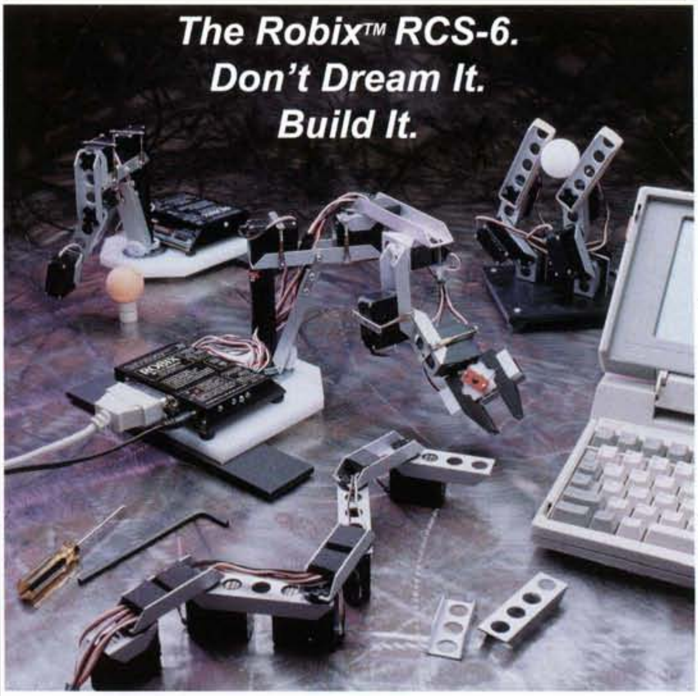
"I'm delighted with the product! Masterful!"
Scott Robbins, Technology Teacher, Winship Junior High

"Faced with a tight deadline we decided to try the Robix™ RCS-6 Controller to operate our custom-built trade show animatronic. We were running almost as soon as we opened the box. Great Product!"
Steve Hyman, Hyman Enterprises, Las Vegas, Nevada

"You sent us one of those educational robots and we've had so much fun with it and we've gotten so good that now we've gone out and bought the real thing. I don't think we'd have been able to do it without first seeing how easy it was to do. We really appreciate it."
James O'Halloran, Chairman, Radius Toothbrush, Kutztown, Pennsylvania

"Two weeks ago we ran the Commissioner of Education from Japan (!) through our program... showed him the Meccano curriculum we were developing, and our US First robot competition entry, trophy, and videos -- ho hum, yawn. But when he and his entourage saw our Robix™ RCS-6 demo, they were riveted."
Michael H. Bastoni, Technology Teacher
Plymouth North High School, Plymouth, Massachusetts

"No science class or technical college should be without one. Students at all levels would probably find the RCS-6 to be the best class activity of the year."
Popular Electronics Magazine, Cover Story



Capable. Inexpensive. Easy. Build and program all designs shown, and many, many more.

"We were up and running in n time, giddy with ideas!... An Erector Set for the Next Generation!"
Wired Magazine

"We've had five sets for over year and our students are jus fascinated with these robots."
Michael Janecek, Technolog Teacher, Palmer High Schoo Palmer, Alaska

"We purchased two RCS-6 systems last month and want t order another. My class of 12 Industrial Electronics students a so enthusiastic about it that they are working overtime and don't want to slow them down.
Jim Richerson, Associate Professor of Electronics, Blu Ridge Community College, Weyers Cave, Virginia

"I am so pleased with the Robix™ [RCS-6] I purchased from you two months ago that have decided to purchase a second as a gift."
John Ruttenberg, Principal Engineer, Silicon Graphics

"I teach Machine Design in the Industrial Tech Department at Illinois State University and I wanted you to know how muc we have enjoyed your Robix™ RCS-6 kit... Your kits have been getting quite a lot of pres as Illinois State, Southern Illino University and Illinois Wesley are all using them."
Harry McBurney, Illinois State

FEATURES

The RCS-6 Hardware includes **everything needed** to work with your **DOS PC** - whether it's an old **PC-XT** or a **Pentium** or something in between. The RCS-6 comes with **6 servomotors**, **controller** including an 8 channel 8 bit **A/D**; **power supply**; 20 **custom extruded, machined and anodized aluminum "links"** (~5' total); **steel/nylon joints, clamps and guides**; "arm" and "breadboard" **construction bases**; **parallel-jaw gripper/wrist** assembly; all needed **tools**, safety **goggles** and safety marking **tape**; 40 minute **video** and 80+ page **manual** including 25+ pages of **illustrated projects**; laminated **reference card**; parallel port **cable**; **"props"** for robots to handle; and a rugged compartmented poly **tool case** that **stores** it all.

The RCS-6 Software is **layered**, starting with a **"no typing" teach mode**; adding **macros** for loops and nesting; and finally adding **C-Language** calls for logic and use of sensors. Software layers offers utility at **Grade School through University** level, allowing a single product to serve broad needs.

THE RCS-6 Video is ***Free to Educators*** (others pay just $3 for priority postage), so you can see the set ***in action*** and ***in detail*** before you buy. Just call or fax on your letterhead to request a copy.

THE RCS-6 Price is **$550*** complete +$8 s/h.

Surprised? Don't be --
We **design**, **manufacture**, and **sell direct**.
And we look forward to hearing from you soon.
YOUR ROBOT IS WAITING...

ORDERS

ROBIX™

Advanced Design, Inc.
6052 N. Oracle Road, Tucson, AZ 85704, USA
tel: 520 544 2390 fax 520 575 0703
email: desk@robix.com
www.robix.com

The World's Most Powerful Control System on a Chi

Basic Express™

BasicX™ On-Chip Feature

Multitasking Operating Syste
Embedded Basic Langua
Networking to 460Kb
IEEE Floating Poi
Clock Calend
And more

INTRODUCTORY PRI
$19.95
QTY. 10

WWW.BASICX.COM

tel. 520-544-4567 fax 520-544-0800
NetMedia Inc. 10940 N. Stallard Pl. Tucson, Arizona 85737

BasicX is Powerful, packing an Embedded Multitasking Operating System closely coupled with a Basic language all on a 40 pin chip (or 44 PLCC). BasicX has system calls specifically designed for control applications including, multitasking, networking, direct communications with I/O ports and internal chip functions.

BasicX is Fast, executing up to 100,000+ Basic lines of code per second.

BasicX busts Numbers. How many controller IC's have IEEE floating point? BasicX does! We also support boolean, bytes, integers and longs.

BasicX does Networking. An on-chip peer-to-peer network specifically built for control systems makes communication a snap.

BasicX has major I/O. Depending on your application you can make full use of the 32 I/O pins.

Complete development system $99.95

Compiler and Environment
For Windows 95/98 on CD
BasicX Chip
32K Bytes of Code EEPROM
In-Circuit Emulator Cable
RS485 Network Interface
COM port interface
Parallel Port Download Port
Expansion Connector
On-Board Power Regulator

LCD-X: 4X20 LCD w/BasicX $9

LCD-X is a fully integrated BasicX system and LCD display. Now display and control what is happening at the same time. Download your programs into the display using the BasicX development system. Communicate with other BasicX chips, modems, PC's or other serial devices using the on-board RS232 or RS485 network.

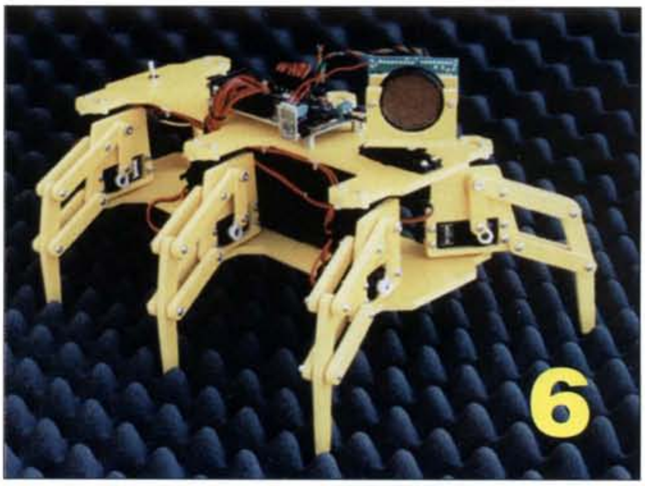
On-board BasicX System
BasicX Network or RS232
I/O's for Switches, Keypads, Data
Software Backlight & Contrast
EEPROM Backup
Display Source Code Included

ROBOT
science & technology

Departments

Features

Fluffy, The Convertible Robot



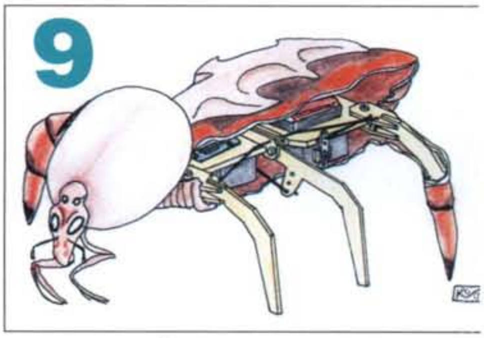
by Chris Harriman

A cool upgrade fo
your Hexapod
using the BasicX
Microprocessor.

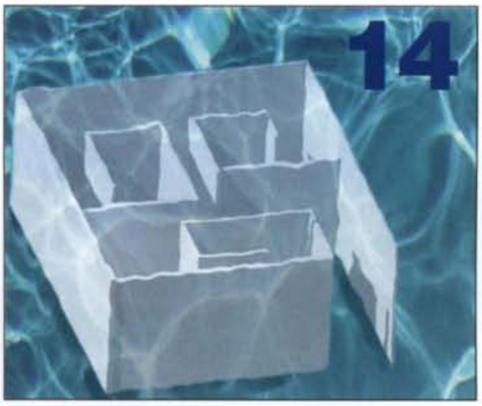
Build Your Own Hexapod Walker Robot – Part I

by George York and Shelley Christopher

How to assemble the chassis and legs to give the builder a base for conceptualizing the skins described in *The Hexapod Walker Comes to LIFE.*



The A* Algorithm - Part IV



by Tak Auyeung, PhD

A planning method for finding the fastest path in a micromouse maze. Fourth in the series by our friend at the University of California at Davis.

contents

Intelligent Evolving Soccer Robots – Part II

by Mohammed Jamshidi, PhD, Denise Padilla, and Marco de Oliveira

Soft computing – enhancing robots' ability to play soccer at NASA's ACE, University of New Mexico.

FIRST Robotics- A Robot Is Born

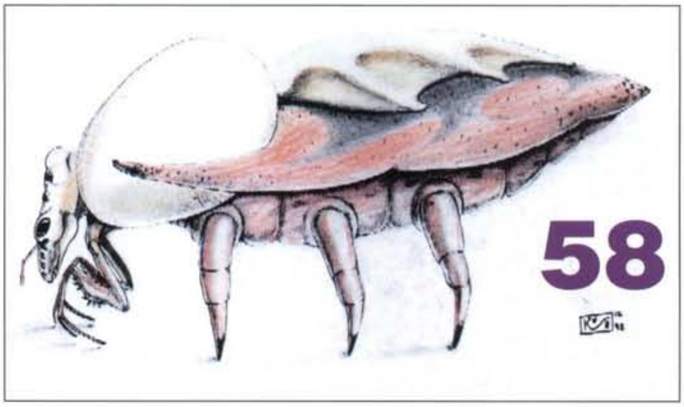
by Floyd Painter

High school students engineer and build their competition robot - the story of *G-Force*.

The Hexapod Walker Comes to LIFE - Part I

by George York and Shelley Christopher

Spice up your bot in skins that make yours *CRAWL*.



Constructing a Combat Robot – Part II

by Ronni Katz

Second Step: Cutting and fitting, plus where to get help and parts.

Cover Design: Julie A. Knudsen. *Fluffy* is a convertible walking robot built on a *Lynxmotion H2-KT Hexapod II Walker* kit chassis. It can function autonomously without a joystick, or be operated manually with it. Photo by Jack Schoof of *NetMedia, Inc.*

OUR MODULES....YOUR ROBOTS

The **RPC module** is an intelligent transceiver which enables a radio network link to be simply implemented between a number of digital devices. The module combines an RF circuit with processor-intensive low-level packet formatting and recovery functionality, requiring only a simple antenna and 5V supply to operate with a microcontroller or a PC.

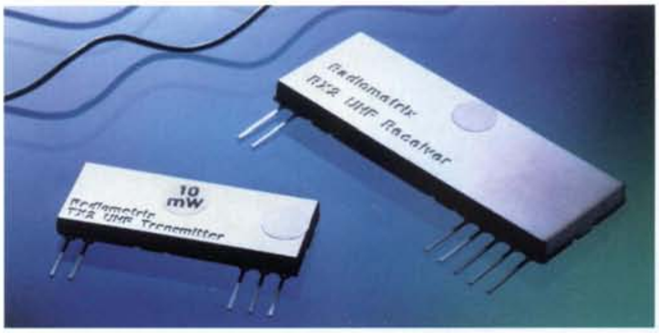
SAW controlled FM transmitter and superhet receiver
Reliable 30m in-building range, 120m open ground
Built-in self-test/diagnostics/status LEDs
Complies with ETS 300-220 regulations
40kbit/s half duplex
Free format packets of 1-27 bytes
Packet framing and error checking are user transparent
Collision avoidance (listen before transmit)
Direct interface to 5V CMOS logic
Single 5V supply @ < 20mA
Power save mode
Available in 418 and 433 Mhz

The **BiM module** integrates a low power UHF FM transmitter and matching superhet receiver with data recovery and TX/RX change over circuits to provide a low cost solution to implementing a bi-directional short range radio data link.

ETS 300-220 tested for European use
SAW-controlled FM transmission at -6dBm ERP
Double conversion superhet receiver
-107dBm receive sensitivity
Single 4.5 to 5.5V supply < 15mA (tx or rx)
Reliable 30m in-building range
Half duplex data at up to 40kbit/s
Direct interface to 5V CMOS logic
Fast 1ms power up enable for duty cycle power saving
On-Board data slicer, supply switched and antenna change over
Available in 418 and 433 Mhz

NEW! The **TX2 and RX2** radio transmitter and receiver pair enable the simple implementation of a data link at up to 40bit/s at distances up to 75m in-building and 300m open ground. Both modules combine full screening with extensive internal filtering to ensure EMC compliance by minimizing spurious radiations and susceptibilities. The TX2 and RX2 modules will suit one-to-one and multi-node wireless links in applications including car and building security, EPOS and inventory tracking, remote industrial process monitoring and computer networking. Because of their small size and low power requirements, both modules are ideal for portable battery-powered applications such as hand-held terminals.

Transmitter - TX2
EMC conformant to ETS 300-683
Type Approved to ETX 300-220
Two-stage, SAW controlled, FM modulated at up to 40kbit/s
Operation from 2.2 to 6V
10mW on 433.92MHz, 1mW on 418MHz
Improved frequency and deviation accuracy
Available in 418 and 433 Mhz



Receiver - RX2
Double conversion FM superhet
SAW front end filter, image rejection 50dB
Supply 3.0 to 6.0V @ 13mA
40kbit/s, -F version, -100dBm sensitivity @ 1 ppm BER
14kbit/s, -A version, -107dBm sensitivity @ 1 ppm BER
LO leakage < -60d Bm
Available in 418 and 433 Mhz

TYPICAL APPLICATIONS

robotics • environmental monitoring • remote process monitoring • wireless PC printer links • energy management
in-vehicle sensing high integrity security • EPOS & stock control • group response systems • data networks
data transfer through hazardous environments • commercial & domestic security • fire alarms • medical alert • mobile panic attack

Radiometrix specializes in design and manufacture of low power radio products to help you to rapidly implement high reliability cable-free data links

900 MHz versions available

For North American sales, contact:
Lemos International Co. Inc.
tel: (508) 798-5004 fax: (508) 798-4782
lemosint@ma.ultranet.com • www.lemosint.com

ROBOT
science & technology

April 1999
Volume 1, Issue 5

Publisher
Michael A. Greene

Editor-in-Chief
Floyd C. Painter

Technical Editor
Eugene P. Ronan

Customer Service
Elizabeth A. Walker

Art & Design
Julie A. Knudsen

Contributing Editors
John Piccirillo, PhD
Kristine Wills, MD

Contributors
Tak Auyeung, PhD
Shelley Christopher
Chris Harriman
The Iconoclast
Mohammed Jamshidi, PhD
Ronni Katz
Marco de Oliveira
Denise Padilla
George York

Advertisers, see page 50
916.632.1000
publisher@robotmag.com

Writers, see page 54
Send manuscripts to
Robot Science & Technology
2351 Sunset Boulevard #170-253
Rocklin, CA 95765
editor@robotmag.com

Subscribers and Libraries
See page 49

Educators Call
916.660.0480 for reprints

Our Mission: to immerse readers in 21st century robotics technology with in-depth reports on real robots, and through hands-on adventures with home, classroom, and sport robotics.

from the publisher

The World's Best Organized Competition hosted thousands of students in seven cities during the last thirty days. If you haven't yet become involved with the FIRST Foundation, do it now! They are revitalizing whole communities, re-energizing educators, and putting excitement back into learning. (See RS&T Jan, page 6 and this issue, page 25.)

Most dramatic result: Some young folks are getting off the streets, burning off their tatoos, and getting back into class. It's obvious why I'm so pleased to support this first-rate program, For Inspiration and Recognition of Science and Technology. See www.usfirst.org. I and the RS&T staff will see you at FIRST's national championship at EPCOT in Orlando, April 22-24.

Sony's Home Entertainment Robot is on everyone's mind. Tune in to robotmag.com for the latest breaking news.

It pains me greatly to say that RS&T is not yet a monthly magazine. So we are scheduling bi-monthly issues until our advertising income lets us print monthly. Expect your next issues to arrive in May, July, September, November and December. Related issue: we're adding a full-time advertising rep to service advertisers' needs. It's a synergistic thing.

What's the Status of your Subscription? What's in the next issue? When will it be shipped? Check out our new Frequently Asked Questions at www.RobotMag.com. No modem? No problem! Call us at 888-510-7728 (US/Canada) or 916-632-1000. Our customer service has pleasantly surprised folks who need help with their subscriptions. So whenever a magazine doesn't show up, call or write. (We aren't a big corporation with nameless subscribers, and we take great pride in making you happy, one reader at a time.)

The Robotics Mini-FAQ is now online to help beginners get immersed in robotics without being swamped by a full-sized FAQ. Contributing editor John Piccirillo conceived it, wrote it, and we're adding those pages to our bigger, better, content-filled website. Bonus: We also put samples of our back issues online. Check it out at www.RobotMag.com.

My personal thanks to Jennifer Duffek and Vince Wilczynski for all the hard work assisting the judges at the FIRST California regional. That was the highlight of our year!

Michael A Greene

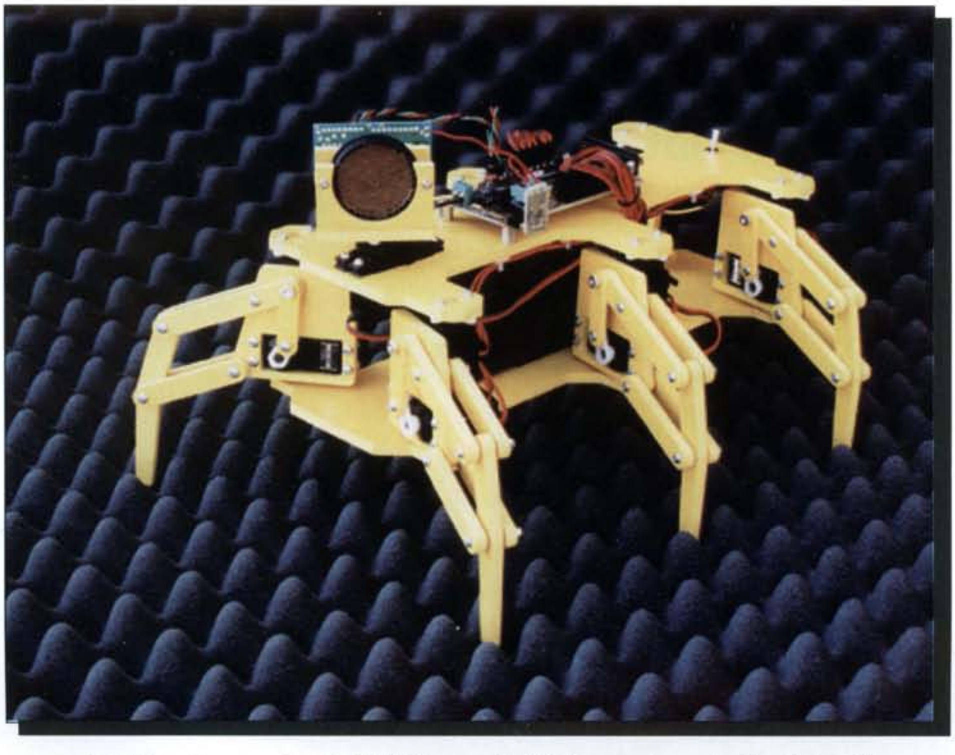
Robot Science & Technology (ISSN 1096-4754) is published monthly at 3875 Taylor Road, Suite 200, Loomis, CA 95650; 916.660.0480; Fax: 916.660.0730; info@RobotMag.com. Subscription inquiries: U.S. and Canada 888.510.7728; all other countries 916.660.0482. Issue price; $5.95 (U.S.) $6.95 (Canada). Subscription rates: 12 issues $39 U.S., $49 Canada (U.S. funds). Other countries see page 49. Periodicals postage paid at Rocklin, CA and additional mailing offices. **POSTMASTER**: Send change of address to: Robot Science & Technology, 2351 Sunset Blvd #170-253, Rocklin, CA 95765.

FLUFFY, The Convertible Robot

Upgrade your Hexapod With The BasicX Microprocessor

by Chris Harriman

Fluffy has a split personality. The six-legged walker is a popular walking robot upgraded with an integrated microprocessor system and sonar. When a standard computer joystick is attached, *Fluffy* is manually controlled. Without the 'stick a panning *Polaroid* sonar gives it the capability to operate autonomously while exploring its environment. This dual personality is the result of an evolutionary build-and-test process you too can use to build a convertible.



Fluffy is a Lynxmotion H2-KT Hexapod II Walker configured with a Polaroid sonar sensor system. The Polaroid scanner is shown in its left-look position.

For years, a basic truth about walking robot kits stated that the programmable microprocessors that were available could not handle the servomotors and have capacity left for a sophisticated sensor suite. I had built walking robots, but had usually been disappointed in their performance. The 'dream walker' I always wanted to build would avoid obstacles and not run into walls. It was a project put off, because the required processing capability was not available. That changed with the introduction of the *BasicX* microprocessor system, which has raised the bar as far as speed, random access memory and code space are concerned. This new system's capability rekindled my interest in creating a cool walking robot, and I soon found myself building an *H2-KT Hexapod II Walker* kit from

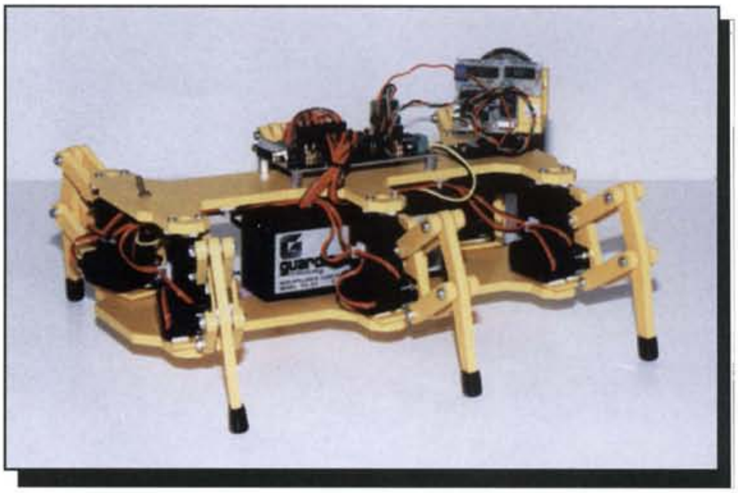
Lynxmotion, Inc. Integrating the *BasicX* system with the kit easily met the processing requirements of the walker servomotors, and provided the additional capacity needed to run obstacle avoidance sensors.

It took about two weeks to build a walking robot that didn't stagger, which in itself was a real accomplishment. The fact that I was able to control the whole thing with one *BasicX* processor on my own custom-built controller board was icing on the cake. In fact, the robot is a convertible because of the approach taken to perfect its walking ability. On the first custom controller board, a joystick port was used to aid in the programming of the walking algorithms. Using the joystick to test algorithms and make required changes was easy. Once the walking program was perfected, the final controller board with additional input/output capability for a sensor suite was constructed. At this point in its evolution, the robot was manually controlled and a very competent walker, with expandability.

Under The Hood

The *H2-KT Hexapod II* robot is a well-designed six-legged walking robot kit that has a lot to offer the roboticist. For locomotion, it uses two servomotors to control each of its legs, and walks with a tripod gait. The center leg from one side, and the front and back legs from the opposite side move as a set, while the other three legs remain on the ground for balance. The *Hexapod* is able to walk forward or backward, and can turn left or right within its own footprint. The robot is about 12 inches long, 11 inches wide, almost 6 inches tall and has about 3 1/2 inches of ground clearance.

Thanks to a well-written instruction manual, most roboticists will not have any trouble with the build-up of the *Hexapod* chassis. Some light finishing work is required on the pre-cut plastic pieces, but it just involves removal of flash. Viewed as a collection of parts, the *Hexapod* kit looks complex, but thanks to good engineering, the assembly process is very straightforward. You will spend a lot of time bolting together and aligning assemblies because of the sheer number of joints that require bolts, washers, and nuts. The crucial step in the assembly process is the alignment. It is very important to be sure that all the joints move smoothly and the servomotors are properly installed and aligned. Without proper joint movement and alignment of the leg and shoulder joints in relation to the servomotors, the *Hexapod* will not walk straight. It will stagger or drift to one side, which is unacceptable if you plan to use a dead reckoning navigation system.



The Guardian DG-63 battery is mounted between Fluffy's main deck plates. This view shows the servomotors, controller board and the back of the Polaroid sonar sensor suite.

The problem is manageable, and the risk of poor alignment can be reduced during the construction process. This is accomplished by insuring that the movement (range of motion), and servomotor alignment of all six legs are identical. Do this by building the leg units for one side, and using one of them as an alignment and joint movement guide to build the three leg units for the opposite side. Finally, once all the leg units are equal in terms of movement and alignment, finish the construction by mounting them to the chassis.

Two major changes to the basic *Hexapod* chassis kit were made that were not in the instructions. First, to increase battery life, a *Guardian DG-63*, 6-volt three ampere-hour gel-cell battery was used. It is mounted over the robot's center of gravity between the upper and lower chassis plates. Second, a panning *Polaroid* sonar sensor from an old instant camera was added. An identical sensor, the *R14-Sonar 1* is available as a package (sensor, driver board, and interface cable) from *Acroname, Inc.* (see Resources).

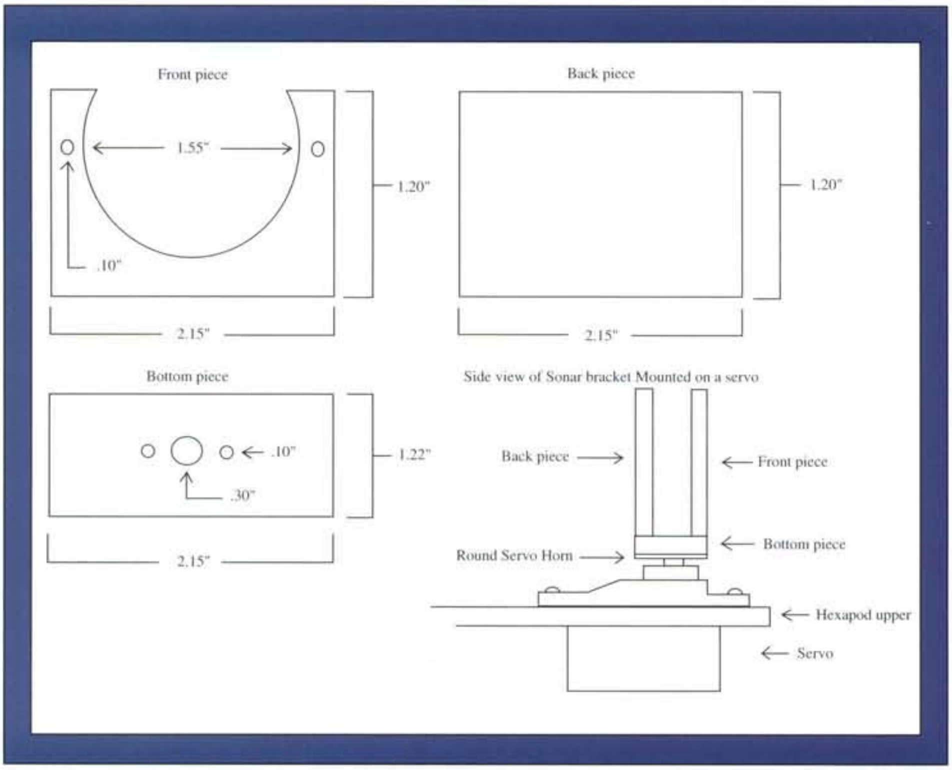


Figure 1. Polaroid Sonar Mounting Assembly.

The panning sonar suite was added by cutting a square hole in the upper plate and adding a 13th servomotor to the upper front of the *Hexapod* chassis. The three pieces that make up the mounting assembly for the sensor and its driver board were crafted from some of the leftover kit plastic. These were super-glued together (see Figure 1) and fastened to the top of the servomotor's control horn with two small screws.

The *BasicX* Chip

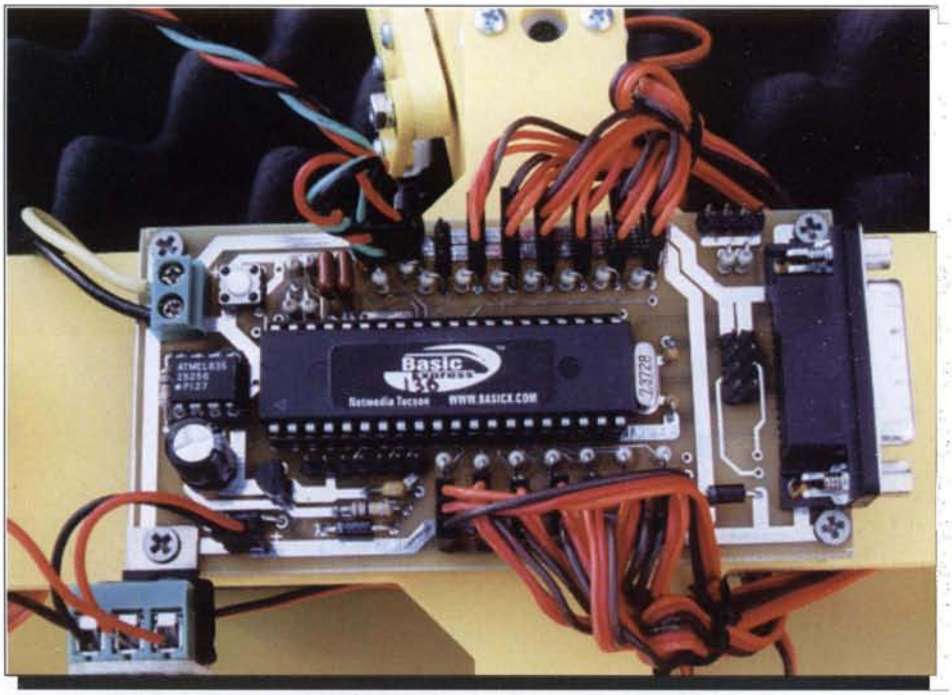
The *BasicX* is a single chip Basic programmable microcontroller that requires only an external crystal and a 3-6 volt power source to run. It was used for this project because it has: (1) tremendous speed, 65,000 lines of Basic code per second, (2) a large electronically erasable programmable read only memory (EEPROM) for user code storage, (3) random access memory (RAM), (4) multitasking capability, and (5) 32 input/output (I/O) pins. This chip incorporates new and upgraded technological features absent from many of its predecessors.

To build *Fluffy*, you will need the *BasicX Development System*. This is a kit with all the materials required to program the chip. It contains one *BasicX* chip and development/downloader board, as well as software for creating, editing, compiling and downloading your programs. The *BasicX* programming language closely mirrors that of *Visual Basic*. In fact, most programs that run on the *BasicX* chip would run equally well on your *Basic for Windows*.

Using *BasicX* to control a servomotor

A servomotor is comprised of a direct-current (DC) motor, a position sensor on the shaft, a gear-reduction set, and an integrated circuit. To operate a servomotor you need two things, power (between 4.5 and 6.5 volts) and a data signal. The important elements of the servomotor's data signal are:

- The position signal - the high signal (+5v). The duration of this high pulse can be anywhere from 1-2 ms. This is the command that tells the servomotor what position to be in, or move to, if it's not already there.
- The refresh rate - the total length of time it takes the data signal or pulse stream to go from start of high, through to low, and back to start of high again. It is typically between 10-20 ms in length.



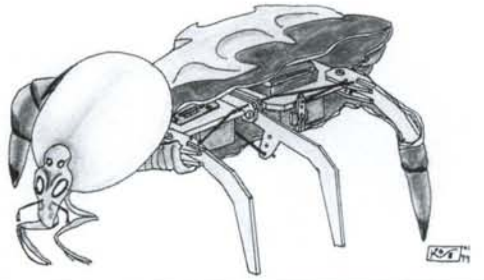
The controller board, showing the Polaroid sonar sensor connectors at the lower left and slightly left of center at the top, the BasicX Chip mounted in the center of the board, the thirteen servomotor connectors (six at the top and seven at the bottom), and the joystick connector at the right.

(continued on page 35)

Build Your Own *Hexapod Walker* Robot! Part I

by George York and Shelley J. Christopher

Constructing a robot doesn't have to be complicated. Build-it-yourself robot kits are available for roboticists at every level, from the novice to the expert. To prove that just about anyone can build their own bot, this series will provide thorough step-by-step instructions on constructing an affordable, introductory system.



We will be building *Lynxmotion, Inc.'s H1-KT Hexapod Walker I* robot, providing detailed instructions and illustrations that will guide you through every facet of the fabrication process. How to modify the *Hexapod Walker* by adding an Infrared Proximity Detector (IRPD) which allows the robot to react to its environment, will also be covered.

Jim Frye of *Lynxmotion, Inc.* explains on their web site *(lynxmotion.com)* that their kits are designed for people with some model or electronic kit assembly experience, and they know of high school students and parent/child teams who have successfully assembled their kits. *Lynxmotion's* goal is to manufacture and distribute affordable, high quality kits to universities, high schools, and other roboticists. Their kits are designed to fill a gap in the educational, hobby or 'personal' robotics arena, and they caution that their projects have been known to spark imaginations as well as to make one think.

To get you thinking, the *H1-KT* kit provides all of the components necessary to assemble a fully functional robot. Also provided is an illustrated assembly manual. However, any robotics engineer will admit that robot construction is replete with modification and experimentation. In this series, we will deviate from the assembly manual, but the result will still be a fully functional *Hexapod Walker*.

Since this *Hexapod* has the potential to resemble an alien insectoid creature, this construction series will run concurrently with articles on how to bring your bot to life by creating robotic outer coverings, or "skins." In that series (see ***The Hexapod Walker Comes To LIFE*** on page 58), the *H1-KT* will become a *Mantid Hexaptera*, or MH6 for short, with full step-by-step instructions on skin fabrication. You can build the *H1-KT* here, and bring it to life there.

This article is Part 1 of the basic *H1-KT* construction process. Part 2 of the series will focus on the electronics, including the Infrared Proximity Detector (IRPD), and will explain programming and software.

Before beginning any construction project, inventory the materials and verify that all of the pieces are included in the kit. The construction process will begin with the mechanical section, which includes the servos and hardware pieces (Part 1).

It is useful to categorize these sections as Part 1 and 2 so that the project can be coordinated. This will help the inexperienced roboticist to visualize where they will start and how they will finish. Be careful NOT to remove the components that are on the anti-static pad in the bag. Static sensitive semiconductors can be damaged by improper handling.

Since this project is running concurrently with the skin fabrication series, some procedural modifications were required. For simplification purposes, the *Hexapod* fabrication

Materials Required to Get Started

Sandpaper or emery board
Xacto knife
Double-sided Foam Tape
(Scotch / 3M Heavy Duty
Mounting Tape found at
most office supply stores)
Straight-edge
Marking Pen
Phillips screwdriver
Cyanoacrylate Glue
(super glue)
Support Structure

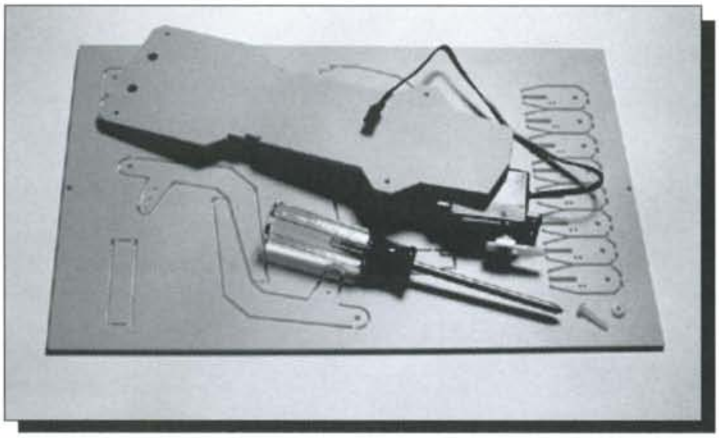


Photo 1

begins here with the construction of the *Hexapod Walker* body and legs, and the attachment of the servomotors. As the mechanical structure takes form, the fabricator will see where and how the electronic circuits will fit for Part 2. When the structure is complete it will be available for measurement and visualization of what the "creature" will look like that is being contemplated for the companion article.

Getting Started

1. Take the plastic panel that has the *Hexapod's* pieces machined into it and cut the pieces from the panel using a sharp *Xacto* blade. Cutting out the pieces is suggested because snapping or twisting them from the panel could distort or ruin their shape. Once removed from the panel, there should be eight pieces that will form the hinges, six legs, and three pieces for the horizontal leg assembly (see Photo 1). (NOTE: Do *not* lose any of these pieces! Extra pieces are not included in the kit and every piece has a specific purpose.)

2. Once the parts are removed, they will require some clean up at the points where they were connected to the panel. With a piece of sandpaper, or an emery board, sand the edges of the pieces until smooth.

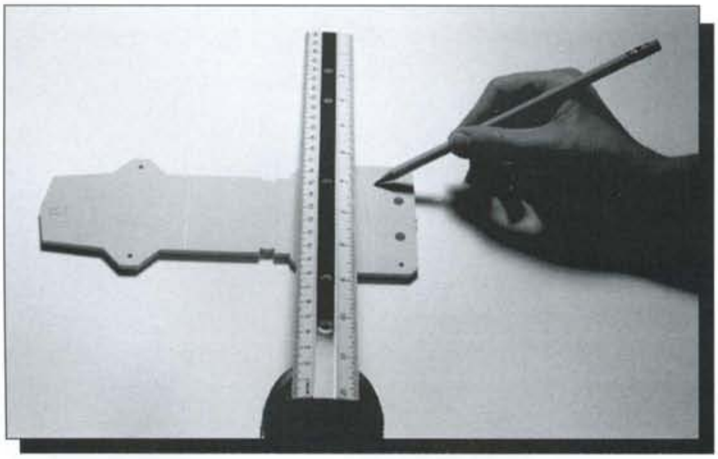


Photo 2

3. Identify one side of the body structure as the "top" side and one side as the "bottom."

4. On the bottom of the body structure, measure 3-3/8s inches (85.7 mm) from the nose of the body (the narrow end) toward the center of the body. With a marking pen, mark this point. Draw a line perpendicular to the longitudinal axis across the structure, at the mark. This measurement will be the reference for the placement of the center servomotor.

5. A second reference line is required on the bottom of the body structure. This time, measure 1 inch (25.4 mm) from the tail of the body (the wide end) toward the center of the body. With a marking pen, place a mark at this point. Now draw another perpendicular line across the structure at the mark. This reference line will help with the placement of the right and left servomotors (see Photo 2).

Setting the Servomotors to Mid Position (90 degrees)

A First Step guide is provided for setting the servomotors to mid position by using the microcontroller (the servomotors must be in mid position before attaching them to the robot body structure). These instructions could be followed, but that would require beginning the construction process with the electronics assembly. Below is a shortcut that you can use. Have available the servomotors, plastic couplers, plastic horns, and mounting screws. (This may look like many steps, but the process is really quite easy.)

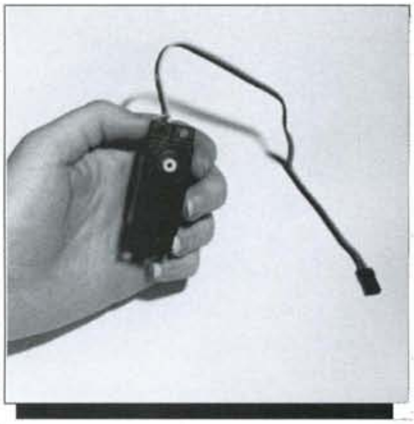


Photo 3

Centering Two Servomotors

1. Hold a servo so that the coupling gear (the white round piece protruding from the servo) is at the top. The manufacturer's label is located beneath the servo coupling gear (see Photo 3).

2. Select a coupler (the small, white plastic piece that is square, but has a circular hole in the middle) and press its round back onto the servo coupling gear. By hand, rotate the attached coupler on the servo coupling gear clockwise until the gear stops. When the gear stops, remove the coupler by gently pulling it from the servo coupling gear. This process sets the servo at 0 degrees.

3. Still holding the servo, place the coupler back on the servo coupling gear, with the coupler's edged sides as near to parallel as possible, to the long sides of the servo. The "open" sides of the coupler are nearly parallel to the top and bottom of the servo (see Photo 4).

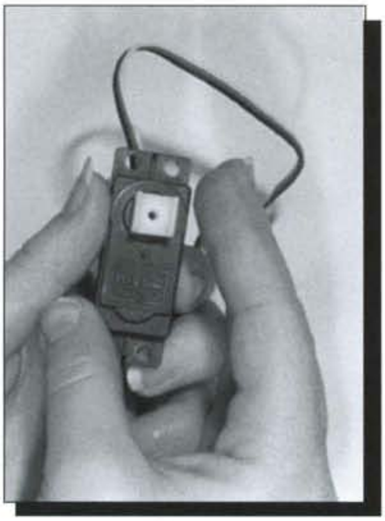


Photo 4

4. Repeat steps 1 through 3 above on one more servomotor. The third servomotor is reserved for later.

Aligning the Center Servomotor

1. Select one of the two servomotors set at 0 degrees to function as the CENTER servo. By hand, rotate the coupler counterclockwise 90 degrees (until the edged sides of the coupler are parallel to the top and bottom short sides of the servo).

2. Select a plastic horn and place it on the coupler so that the horn is pointing to the left. The horn has a ridged or "teethed" side, which will directly engage the coupler. This allows for an adjustable fit that will seat properly and securely. Be sure that the horn is placed on the coupler so that the flat end of the horn is flush with the right edge of the coupler. This will give the horn enough clearance to function once attached to the robot body. Fasten the horn onto the coupler with a screw using a Phillips screwdriver. Tighten until snug, but not too tight (minor adjustments may later be required) (see Photo 5).

Aligning the Left Servomotor

1. The second of the two servomotors that were set at 0 degrees will function as the LEFT side servo. Rotate the coupler counterclockwise 90 degrees (until the edged side is now parallel to the top and bottom of the servo).

2. Place a plastic horn on the coupler so that the horn is pointing left (as in step 2 above). HOWEVER, be sure that the horn is placed on the coupler so that the screw attaches the horn at the center of the adjustable opening. Fasten the horn onto the coupler with a mounting screw and tighten until snug. Do not over-tighten.

Place the center and left servomotors aside and be sure that they can be identified later. It is suggested that these servos be marked with "center servo" and "left servo" stickers for identification purposes.

Aligning the Right Servomotor

1. The third servomotor will function as the RIGHT servo. Press the round back of a coupler onto the servo coupling gear. Rotate the attached coupler on the servo coupling gear counterclockwise until the gear stops. Remove the coupler by gently pulling it from the servo coupling gear. This process will set the right servo motor at 0 degrees.

2. Still holding the servo, place the coupler back on the servo coupling gear, with the coupler's edged sides as near parallel to the sides of the servo as possible (the open sides of the coupler will be nearly parallel to the top and bottom of the servo.)

3. Rotate the coupler clockwise 90 degrees.

4. Place a plastic horn on the coupler so that the horn is pointing to the right. Attach the horn to the coupler. Be sure that the screw is placed at the center of the open adjustment of the horn. Attach until snug.

5. Mark the right servo for identification.

Attaching the Servos to the Robot's Body Structure

There are now three servos set at 90 degrees ready to be attached to the bottom of the body structure.

1. Place the center servo aft of the line that was drawn at 3-3/8s inches. The servo's horn will be pointing upward. Place the side of the servo that is opposite the attached horn at the line. The side of the servo that has the horn attached will be facing the back-end of the robot body

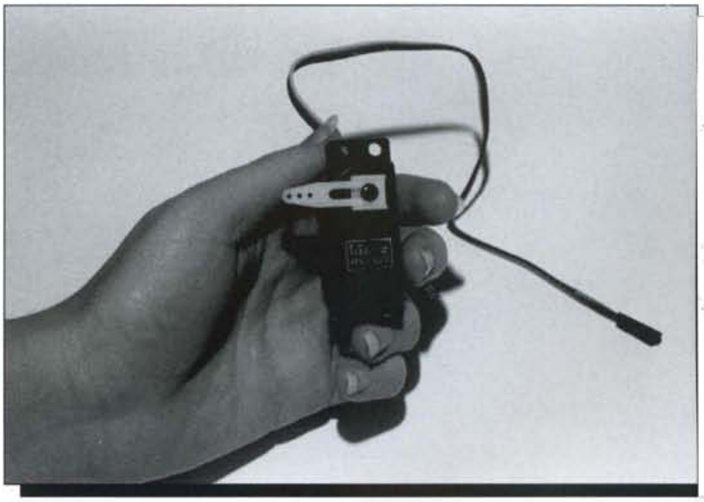


Photo 5

(see Photo 6). The servo must be centered at this line. Attach the servo in place with the double-stick tape.

2. Place the left servo and right servo on the body structure for a trial fit. Each servo's horn coupler must point downward (with the robot resting on its back). Place them forward of the line that was drawn 1 inch from the back of the robot body structure. Lay them on their side, so that the horns point down and appear to "hang" over the side of the body structure.

3. Once the servos have been positioned satisfactorily, mark their placement with a marking pen. Attach the servos to the body structure with double-stick tape.

4. When the servos have been taped in place as shown in the photographs, place the body with attached servos on a support structure, such as a large roll of tape. The servomotors should be touching the support structure so that the adjustable horns are not supporting the entire assembly. The servos should now be "hanging" underneath the upside down body structure.

Constructing the Vertical Leg Assembly (For the Center Legs)

To construct the vertical leg assembly, have available the two plastic brackets, the small rectangular spacer, and the two leg pieces that have the protruding plastic tabs with three holes, all previously cut from the plastic panel. Also, have available two plastic screws and nuts. The following process is performed while the robot is resting on the support structure, and the servos are "hanging" underneath the robot body.

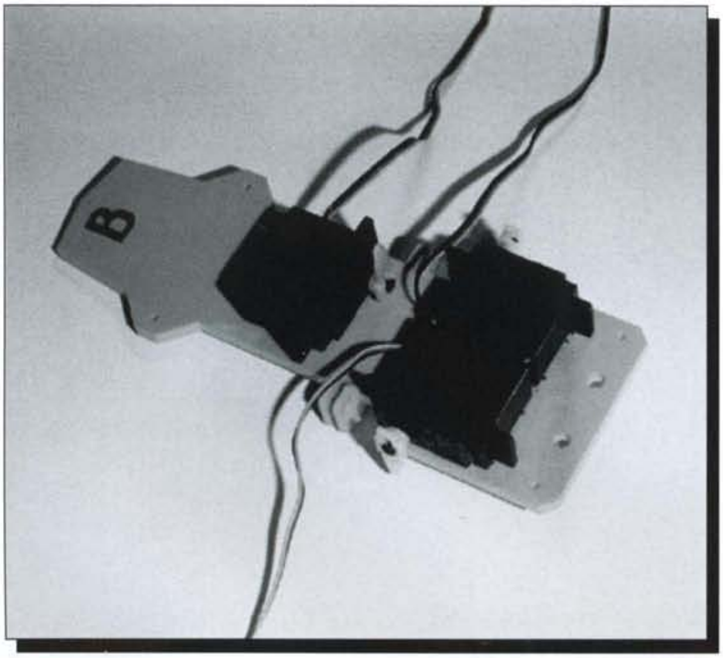


Photo 6

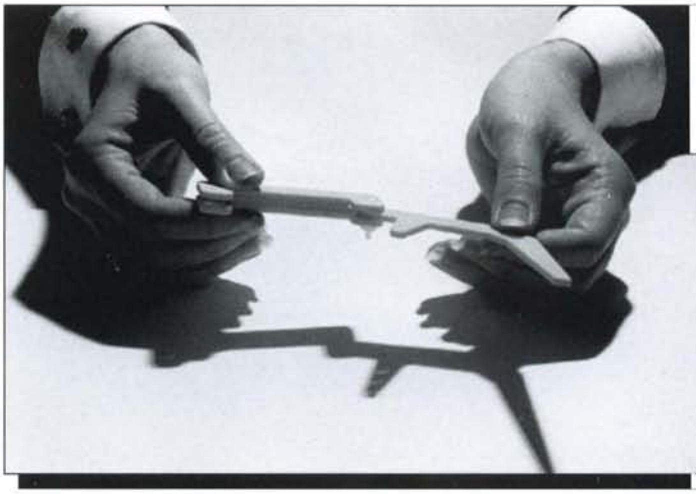


Photo 7

1. Using the two plastic brackets and the small rectangular spacer, sandwich the small rectangular spacer between the two brackets (see Photo 7). The rectangular spacer should be adjusted to fit at a center placement while between the brackets.

2. While holding the sandwich in place, insert one of the plastic legs, hole end first, into one end of the sandwich. Be sure to align the holes on all three pieces (the two brackets and the leg).

3. Insert a plastic screw through the aligned holes. Attach the nut on the opposite side and screw into place for a snug fit. This process may cause the rectangular spacer to move from its centered, in-between placement with the two brackets. However, once the second leg is secured at the opposite end, the rectangular piece will snugly fit between the two brackets. Take the second leg (with the same protruding plastic tab with three holes) and complete the process.

4. Once the center leg assembly is completely constructed, it is super-glued onto the body structure. The leg assembly will be placed in the pre-cut slot on the body. Note that the brackets will fit snugly into the slots. First, confirm proper placement. Then dab super glue on the middle of the sandwich, along the brackets and the rectangular spacer, where the leg assembly will directly contact the body structure. Use the superglue conservatively, and do not allow glue to touch the joints of the leg assembly, where the plastic screws have been placed (see Photo 8).

5. Hold the leg assembly in place until secure.

Attaching the Horizontal Leg Assemblies

Eight plastic pieces will form the four hinges to be used in the fabrication of the horizontal leg assemblies (as well as four screws and nuts). In this process, the four legs will be added to the body structure. This is accomplished while the robot body is resting on the support structure, with the servo motors "underneath" the upside down body structure.

1. Take two of the plastic pieces which will form a hinge, and hold them at a back corner of the robot (it does not matter which side). The robot body will be sandwiched between the two support hinges. NOTE: There are three holes in each hinge: two small holes and one larger hole (the hinge point). Be sure that the hinges to be attached to the *back* of the robot structure are placed with the small holes facing forward toward the front of the body structure. Put a screw through the hinge point and plastic body structure, and attach with a nut.

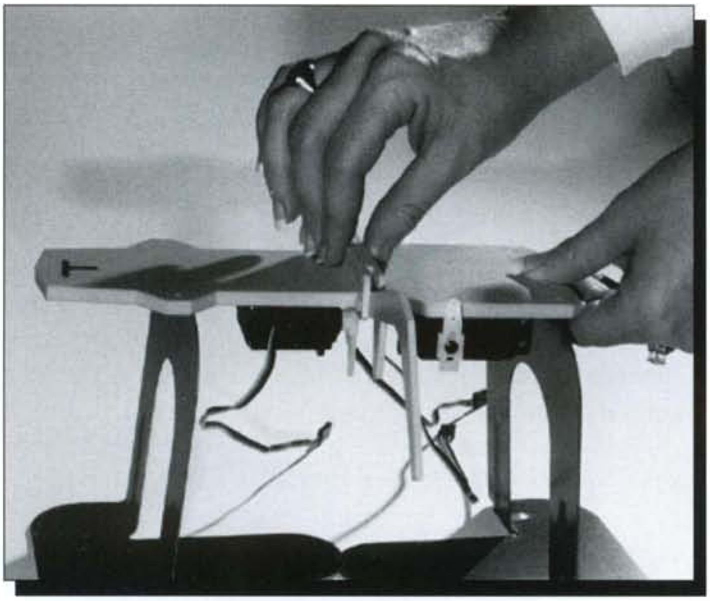


Photo 8

2. Repeat this process for each corner. (NOTE: The small holes of the two hinges attached to the *back* of the structure face forward, toward the *front* of the robot body.) The small holes of the two hinges attached to the *front* of the body structure will face backward, toward the aft end of the robot body. Check your work closely. It will be too late to make corrections once the leg is glued in at the next step.

3. There are four legs remaining to be attached. Apply super glue to the top and bottom edge of the notched end of the leg. Seat the tab into the slot of the horizontal hinge. Push the leg all the way through the slits until it is "seated" and firmly in place. Complete this process for the remaining legs (see Photo 9).

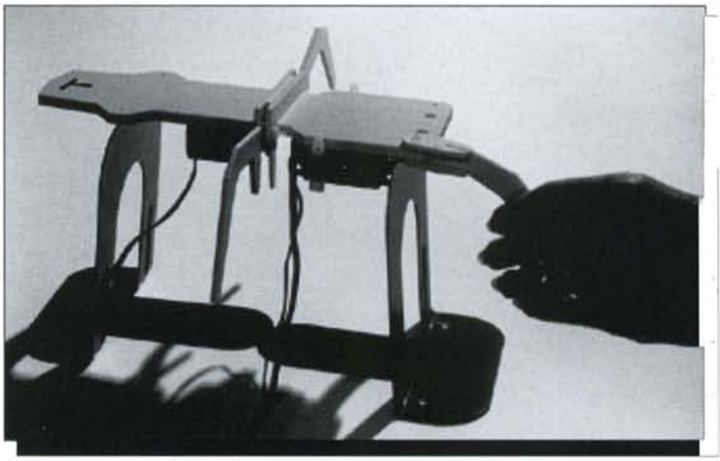


Photo 9

What's Next?

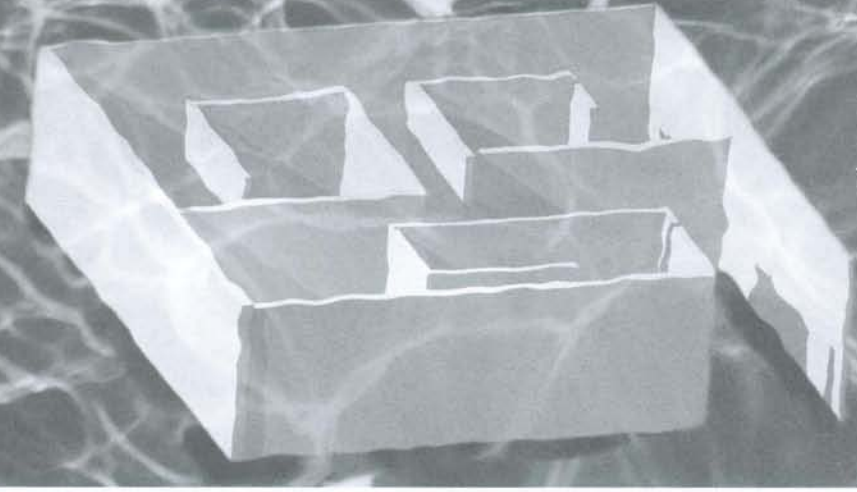
The *Hexapod Walker* has just begun to take shape. The six legs are attached as well as the servomotors. However, this is only the beginning of the exciting robotic construction process! Part 2 of this construction series will feature working with the electronics and programming of the assembly.

First, there was the *H1-KT* kit. Now there is a partially constructed 'bot. We encourage you to follow along in this special two-part series. Also, take a look at how you can bring this 'bot to life with robotic skin fabrication on page 58.

Author's biographies shown on page 62. RS&T

Popular Micromouse Algorithms, Part IV

A* Algorithm



by Tak Auyeung, PhD
University of California, Davis

Introduction

This is the fourth in a series of articles about various algorithms ("algorithm" is a fancy term for "method") used by micromouse robots *to solve the micromouse maze in competition*. These competitions have been organized annually all over the U.S. by various chapters of IEEE. They have proven to be very popular. Depending on the intended contestants, the exact rules may differ among the competitions. The main objective, however, remains the same: use a robotic "mouse" *to solve the maze as quickly as possible*. The basic premise of the problem is that the micromouse robot does not know the configuration of the maze before its first run. The coordinate of the destination, on the other hand, is known. The robot is allowed to store information and repeat solving the maze within a time limit.

Most micromouse robots can only sense whether there is a wall directly ahead, to the left and to the right. Based on such limited information, the robot must rely on proven algorithms to systematically explore the maze to find a path to the destination. A smart robot even tries to find the shortest path to the destination to minimize the time it takes to travel there. In previous issues, we discussed the Wall Hugging, Depth-first Search, and Flood-fill algorithms. In the July 1998 issue, we employed the Wall Hugging algorithm and discovered its main fault. In essence, if the robot swims along the shoreline of a lake, it will never reach an island in the middle. Therefore, if the island is its destination, it will fail. The failure is, of course, attributable to the inadequate algorithm. In the November issue, we discussed another popular algorithm used to solve the micromouse maze, the Depth-first Search (DFS). The main advantage of the DFS is that it guarantees that the robot will explore all cells that can be reached from its starting cell, and map them. Unfortunately, DFS does not provide the shortest path to the destination. A faster, less intuitive method is a little more complicated than the wall hugging and DFS techniques, but has the advantage of finding the destination without having to explore the entire maze. That method was the subject of the January issue's algorithm article – the Flood-fill algorithm. This method may find a way to the destination without exploring all of the cells, and therefore save search time. However, it may not find the shortest path, a necessity for accomplishing the objective of the competition – *to solve the maze as quickly as possible*.

Finding the Shortest Path

Given two robots that can move at the same speed, the one that takes a shorter path will yield a shorter run time. It is, therefore, important to find the shortest path from the starting state (i.e., the *starting cell* and the *initial direction*) to the destination cell.

Assuming the entire maze is known (such as from a DFS exploration), the A* algorithm guarantees to find the shortest path. Note that the A* algorithm is a *planning* method, not an *exploration* method. During the execution of the wall hugging, DFS or flood-fill methods, the robot actually moves around. However, during the execution of the A* algorithm, the robot does not move. Instead, the robot just plans the shortest path based on the known configuration of the maze. The A* algorithm deals with "states" instead of just "cells." A state represents the robot being at a cell *and* facing a certain direction. In other words, the robot at cell X facing north is in a different state from the robot at cell X facing south.

In addition to "states," the A* algorithm relies on a *heuristic function*, or rule-of-thumb, to estimate the length from any state to the destination. Furthermore, the A* algorithm requires the heuristic process to provide an admissible estimate. "Admissible" is merely a fancy term for "underestimating." In other words, the A* algorithm relies on a rule-of-thumb that always underestimates the actual distance from a state to the destination.

In order to express the A* algorithm, it is helpful to introduce a *sorted list*. A sorted list, as the name implies, is a list of items that is ordered by some criterion. As items are inserted or deleted from the list, adjustments are made to ensure the list remains sorted. In the following discussion, we can assume the *head* of the list is the item with the least value. For our discussion, each item in the list has an assigned value derived from a group of data. This data group includes: a state; the previous state; the "action" that gets from the previous state to this state; the known minimum cost to get from the starting state to this state (computed and known); and the value from the starting state to this state added to the underestimated value to reach the destination from this state.

The Method

In this algorithm, "cost(A)" is the cost of an arbitrary action A, and h(X) is the estimated distance from an arbitrary state X to the destination. It is important to realize that the actions being evaluated are not confined to moving forward one cell and making 90-degree turns. Moving ahead two cells is *not* the same as moving ahead one cell twice. Moving ahead two cells is a more efficient action with its own lower total cost, because it takes time to accelerate and decelerate. If the robot is to move forward two cells, it can accelerate to a higher speed, then decelerate to slow and stop at the end of the second cell. These are not the only considerations when a robot moves.

The actual number of distinct actions, or moves, available to the robot is limited only by its physical ability. Some robots must stop to turn. That action has a greater cost value than if it were able to turn on the go without stopping. Two heuristic formulas will be discussed, the first assumes the robot stops before turning and cannot move diagonally, and the second assumes diagonal movement is possible. The heuristic function applicable to the diagonal-capable case is automatically an underestimate for the other one. Therefore, the heuristic function that uses the length of the straight line is applicable to both the diagonal capable and diagonal incapable cases.

Also, note that in the algorithm, we talk about "reachable state N from state X via action A." There is no mentioning of cells in the algorithm. This is because an action may leave the robot in the same cell or a few cells away from the current cell. A 90-degree turn leaves the robot in the same cell, while a "move forward three cells" action leaves the robot three cells from the current cell.

Notations and Variables

Let us first explain the notation and variables involved in this algorithm. The algorithm keeps track of states and other information about states via items with five components (for the lack of a better term, call these items quint-tuples). In a quint-tuple (X,Y,P,D,A), the components have the following meanings:

Component 1: the state X. This component indicates the state to which the information in the quint-tuple belongs.

Component 2: the previous state Y. This component indicates that state previous to this state when the other components are computed.

Component 3: P. The exact known minimum cost from the starting state to state X. Note that this is the exact cost of one of possibly many paths from the starting state to state X.

Component 4: D. The estimated cost of a path from the starting state through state X to a destination. This is always an underestimate of the actual cost. In other words, this component is always less than the shortest path from the starting state through state X to a destination.

Component 5: A. The action taken at state Y to reach state X.

In the algorithm, "considered" is a set of quint-tuples that represent states that have been considered. A "frontier" is a *sorted list* of quint-tuples that represent states that are being considered. The list is sorted in increasing value order, first to last, by the fourth component of the quint-tuple. Effectively, the first quint-tuple of "frontier" represents the most probable state under consideration that should be on the shortest path from the starting state to a destination.

In pseudocode the algorithm reads as follows:

```
update(X,P)
 for all (Z,X,P1,D,A) in considered or frontier do
   if (P+cost(A) < P1) then
         remove (Z,X,P1,D,A)
         insert (Z,X,P+cost(A),P+cost(A)+h(Z),A) to frontier
         update(Z,P+cost(A))
   end if
 end for
end update
A*(S)
   empty the considered set
   create (S,null,0,h(S),null) in the sorted list frontier
   while the destination is not considered do
         remove the first item (X,Y,P,D,A1) from frontier
         put (X,Y,P,D,A1) to considered
         for each state N reachable from state X with an
         action A do
            if (N,X1,P1,D1,A2) does not exist in frontier
            or considered
                  add (N,X,P+cost(A),P+cost(A)+h(N),A)
                  to frontier
            else
                  if (P1 > P+cost(A)) then
                     delete (N,X1,P1,D1,A2)
                           add (N,X,P+cost(A),
                           P+cost(A)+h(N),A) to frontier
                     update(N,P+cost(A))
                  end if
            end if
         end for
   end while
end A*
```

Regarding the A* algorithm, the most crucial statements to understand are the **bolded** ones. These lines associate state N with two numbers. One of the possibly many exact costs from the starting state S to state X is P. As a result, the exact cost of one of the possibly many paths from the starting state through state X to state N is P plus the cost of the action A (P+cost(A)). *Note that this exact cost may not be the cost of the shortest path from the starting state to state N*. This is why we need to check if a path to N has already been found. If one exists, the exact costs of the two paths are compared, and only the shortest one is stored. Furthermore, if a path to state N is already found, it implies that the paths of other states from state N may have been computed (represented by quint-tuples in the "considered" set or "frontier" sorted list). The utility procedure *update* recomputes the quint-tuples of all states whose exact cost depends on the path from the starting state to N.

Heuristic Functions

First, we need to decide what actions can be taken and the associated costs:

- Move forward one cell: 700 ms.
- Move forward two cells: 1200 ms.
- 90-degree turn: 500 ms.

It is the programmer's responsibility to determine the heuristic function (h).

Two Heuristic Examples

This first heuristic function is applicable (admissible) if the robot stops before turning and cannot move diagonally.

h = 600 ms * (abs(x1-x2)+abs(y1-y2))
WHERE
h– the heuristic.
(x1,y1) – coordinate of the cell represented by the current state
(x2,y2) – coordinate of the destination

600 ms – any time estimate that is LESS than or equal to the time required for a move of a single cell, in this case 600 ms. Although it takes 700 ms to execute the "move forward one cell" command, it takes only 600 ms to *move one cell* when the robot executes the "move forward two cell" command.

abs – the absolute value.

abs(x1-x2)+abs(y1-y2) – This is the minimum number of cells to travel even if there were no walls. The horizontal offset of the current cell from the destination is abs(x1-x2), the robot must move abs(x1-x2) cells horizontally even if there are no walls. Similarly, abs(y1-y2) indicates the minimum number of cells to travel vertically even if there are no walls. The sum of horizontal offset and vertical offset is the minimum number of cells to travel in order to reach cell (x2,y2) from cell (x1,y1).

Let us consider the following maze.

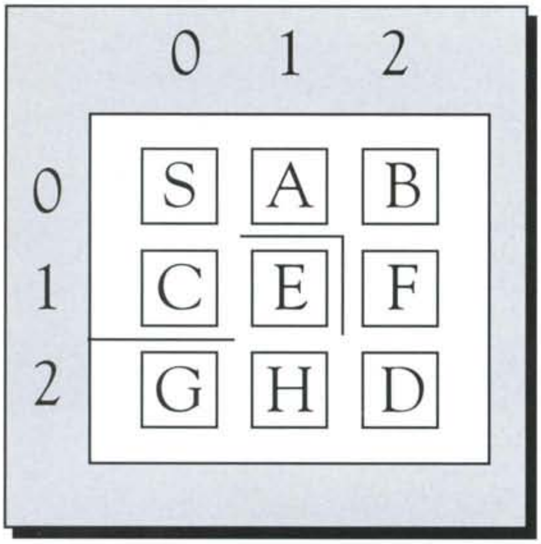


Figure 1. Maze Setup.

At this point you may want to review some basic information provided in previous articles in this series. This information is provided in the sidebar on this page.

Let us consider getting from cell C (0,1) to cell D (2,2). Even is there were no walls, the robot still needs to travel two cells horizontally and one cell vertically. This adds up to three cells to travel, at the minimum. Traveling each cell costs at least 600 ms, therefore three times 600 ms (1800 ms) is an underestimate of time to travel from cell C to cell D.

This second heuristic function is applicable if the robot could move diagonally.

h = 600 ms * (sqrt((x1-x2)^2+(y1-y2)^2))
WHERE
h – the heuristic
600 ms – same as above
sqrt – square root of
sqrt((x1-x2)^2+(y1-y2)^2) – is the shortest distance from cell (x1,y1) to cell (x2,y2). Note that this distance cannot be longer than the sum of horizontal and vertical offset.

Let us again consider moving from C (0,1) to cell D (2,2). The straight line leading from cell C to cell D is sqrt(1+2) or about 1.732 cell lengths. Since the robot cannot move faster than 600 ms per cell, 1.732 times 600 ms (1039 ms) is

TERMINOLOGY

Maze board - a square board with regularly spaced pegs. In the Micromouse competitions, the pegs are 18 cm apart vertically and horizontally on the maze board, center-to-center.

Partition - a piece of "wall" that fits between two pegs. In the Micromouse competitions, the partitions are 5 cm high.

Cell - a square formed by pegs at the corners. We will use the convention that the cell at the upper left corner is cell [0,0]. See Figure 1.

Start location - cell [0,0].

Destination - Some location with known coordinates that the mouse needs to go to.

Reachable - If cell B is reachable from cell A, there is a path from cell A to cell B (and vice versa).

Basic Robot Abilities

The physical design of a robot can affect what the robot can do. Regardless of design of the robot, let us assume the robot has the following basic abilities:

Turn 90 degrees clockwise and counter-clockwise.
Go forward one cell and stop.
Sense a wall in front of the robot.
Sense walls to the right and to the left of the mouse.
Once a wall is sensed, remember the wall in internal memory (i.e., build a map).
Keep track of the current coordinate.
Knows if the robot is at the destination.

an underestimate of the time to move from cell C to cell D, even if there are no walls.

Both of these heuristics

h = 600 ms * (abs(x1-x2)+abs(y1-y2)), and

h = 600 ms * (sqrt((x1-x2)^2+(y1-y2)^2))

are valid because they underestimate the actual time needed. For that matter, if you chose h = 0 it would also be a valid heuristic, although it's fairly useless.

Remember, the A* algorithm does not need to evaluate all the possible paths to the destination to find the shortest. Instead, the A* algorithm maintains a list of candidate path "frontiers," sorted by the estimated length of the path from the starting state to a certain state through to the destination.

Let us "eyeball" the maze and intuitively try to find the shortest path. At the first glance, path SCEHD is about the same length as SABFD. However, as soon as we factor in the cost for turning, path SCEHD is clearly more costly. We will use this maze to illustrate the A* algorithm. For illustration, we need to keep track of the sorted list "Frontier" (bold face represent newly inserted quint-tuples) and the considered set. We also use the convention Cell$_{DIR}$ to represent the location (Cell) and direction (DIR, as in North, East, South and West). The starting configuration is S_S, i.e. Cell (S) and $_{\text{DIR (South)}}$. For our *trace table* example, we assume the robot stops before turning and cannot move diagonally. Refer to *Notations and Variables* while calculating the value of components in the 'Frontier' with particular attention to the *actual* values back to the starting state of S_S.

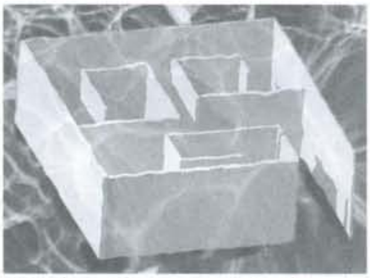
TRACE TABLE

Frontier	Considered	Comments
(S_S,null,0,2400,null)		This is the start up configuration. We need to move the distance of at least four cells, and we can only go as fast as 600 ms per cell. Therefore, the actual time to get to the destination cannot be shorter than 2400 ms.
(C_S,S_S,700,2500,Move South 1) **(S_E,S_S,500,2900,Turn East)** **(S_W,S_S,500,2900,Turn West)**	(S_S,null,0,2400,null)	We can turn 90 degree at cell S, each costing 500 ms or, we can go south, costing 700 ms. However, note that the 90-degree turns add the full 500 ms to the estimated cost to goal, whereas the move south operation only adds 100 ms to the estimated cost to goal.
(S_E,S_S,500,2900,Turn East) (S_W,S_S,500,2900,Turn West) **(C_E,C_S,1200,3000,Turn East)** **(C_W,C_S,1200,3000,Turn West)**	(S_S,null,0,2400,null) (C_S,S_S,700,2500,Move South)	From cell C facing south, the only options are to turn 90-degree (to the east and west). Each turn adds 500 ms to the cost to this state as well as the estimated cost.
(S_W,S_S,500,2900,Turn West) **(B_E,S_E,1700,2900,Move East 2)** (C_E,C_S,1200,3000,Turn East) (C_W,C_S,1200,3000,Turn West) **(A_E,S_E,1200,3000,Move East 1)** **(S_N,S_E,1000,3400,Turn North)**	(S_S,null,0,2400,null) (C_S,S_S,700,2500,Move South) (S_E,S_S,500,2900,Turn East)	From cell S facing east, we can turn 90 degree and face south. But this state is already considered, and the new method is more costly. We can also turn 90 degree and face north. This state has not been considered yet. Furthermore, we can move to cell A or cell B facing east.
(B_E,S_E,1700,2900,Move East) (A_E,S_E,1200,3000,Move East 1) (C_E,C_S,1200,3000,Turn East) (C_W,C_S,1200,3000,Turn West) (S_N,S_E,1000,3400,Turn North)	(S_S,null,0,2400,null) (C_S,S_S,700,2500,Move South) (S_E,S_S,500,2900,Turn East) (S_W,S_S,500,2900,Turn West)	From cell S facing west, turning 90 degree to face north and south are the only options. Both states are considered, and the old methods, costing 0 and 1000 respectively, are less costly than the new methods. No new items are added to the "frontier" sorted list.
(A_E,S_E,1200,3000,Move East 1) (C_E,C_S,1200,3000,Turn East) (C_W,C_S,1200,3000,Turn West) (S_N,S_E,1000,3400,Turn North) **(B_N,B_E,2200,3400,Turn North)** **(B_S,B_E,2200,3400,Turn South)**	(S_S,null,0,2400,null) (C_S,S_S,700,2500,Move South) (S_E,S_S,500,2900,Turn East) (S_W,S_S,500,2900,Turn West (B_E,S_E,1700,2900,Move East 2)	From cell B facing east, turning 90 degrees to face north and south are the only options. Both new states have not been considered yet, therefore they are both added to the sorted list.

We stop the trace here. The determined may work out the entire trace (i.e., until a quint-tuple in the considered set represents one of the destination states, D_N, D_E, D_S or D_W). The minimum cost path from S_S to $D_?$ should be (S_S, S_E, B_E, B_S, D_S).

Topics for Further Research

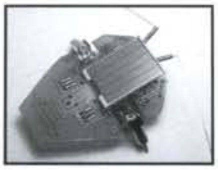
Besides the methods described in these articles, there are many other methods to solve the Micromouse maze. Furthermore, the methods described assume the robot can only implement simple actions. In reality, the robot can be designed to perform in a more complex manner. For instance, the Micromouse rules do not rule out a robot that can sense walls that are out of the current cell. Such information can be used to construct the maze map without exhaustively executing the depth-first search exploration. Similarly, the flood-fill exploration method may use such extra information to shorten the exploration phase. One interesting modification to the flood-fill exploration is to refine the most optimistic distance from destination.



Although possessing the entire maze map is sufficient to let the A* algorithm compute the shortest path, it is not necessary. If the A* algorithm has a heuristic function that closely estimates the actual cost, it is possible that only a small number of states (and therefore cells) need to be considered. This means that an exploration method that only explores necessary cells needed for the A* algorithm may be worth looking into.

While the Micromouse scenario may seem toy-like and limited in scope, it is an excellent introduction to navigational issues of larger robots. Algorithms such as Depth-first Search, Flood-fill and A* are only the beginning of algorithms needed by autonomous robots in the real world.

*We met **Tak Auyeung** at the IEEE Region 6 micromouse competition at UCD, where Tak teaches the UCD Micromouse Lab. In his other life, he's the software development group leader for embedded controllers at Zworld.*



Intelligent Evolving Soccer Robots Part II

Soft Computing Enhances Learning Process in Robots

by Mohammed Jamshidi, PhD., Denise D. Padilla, & Marco de Oliveira

This feature concludes our two-part series on intelligent evolving soccerbots. Part I explored fuzzy logic and fuzzy control, and was included in the November 1998 issue of RS&T. In Part II, fuzzy control, fuzzy behavior, and a generic algorithm for evolving an intelligent robot soccer team are discussed.

One of the current research topics at NASA's Center for Autonomous Control Engineering (ACE) in robotics is the use of multiple autonomous robots for the cooperative execution of tasks. Today *soft computing* - fuzzy logic, neural networks, genetic algorithms, genetic programming, and probabilistic reasoning - has resulted in more user-friendly products that learn to adopt their user's operational characteristics. These robots offer several advantages over monolithic robots, such as robustness through redundancy in numbers, reduced size and weight, and lower production costs.

Fuzzy logic and fuzzy control, a sensory-fusion, and the rule hierarchy approach, allow these complex systems to function. Finally, a hierarchical fuzzy system optimized through evolutionary methods acts as a controller. In conjunction with an algorithm, these systems are employed by a team of robots that play soccer.

Soccerbots must identify and evaluate situations, and react accordingly

As described by Tunstel [1] and implemented by our group at ACE, each robot's overall behavior emerges from the interaction among multiple fuzzy logic controllers, each of which is responsible for a distinct behavior. In the case of the robot soccer test bed, a robot's behavior is governed by these fuzzy modules: *attack*, *defend*, *pass*, *intercept*, *block*, *shoot*, *avoid*, and supported by the non-fuzzy modules, *sense*, *talk*, *move*, *kick*, *and turn*.

Fuzzy logic controllers implement control algorithms consisting of linguistic rules. These rules are usually in the form of *IF x is A and y is B then z is* C (x, y, z being the inputs and output variables), and an inference engine, which executes and aggregates the rules' output. The rules represent mappings between input and output variables. Zadeh [2], in his seminal paper, chose the word "fuzzy" for the continuum of logical values between 0 (completely false) and 1 (completely true). The theory of fuzzy logic deals with two problems: (1) fuzzy set theory, which deals with the ambiguity found in semantics, and (2) fuzzy measurement theory, which deals with the ambiguous nature of judgments and evaluations. The primary motivation of fuzzy logic is to exploit tolerance for some inexactness and imprecision. Fuzzy logic and classical logic differ in the sense that the former can handle both symbolic and numerical manipulation, while the latter can handle symbolic manipulation only. In fuzzy logic, one can see that everything is a matter of degree.

Fuzzy control systems are *rule-based* systems in which a set of so-called fuzzy rules represents a control decision mechanism that substitutes for or emulates a skilled human operator. A fuzzy controller typically takes the form of a set of if-then rules whose *antecedents* (if part) and *consequents* (then part) are themselves membership functions. Consequents from different rules are numerically combined and are then collapsed (typically taking the *centroid* of the combined distribution) to yield a real-number (binary) output. Thus, a fuzzy controller works in a way similar to a conventional system: it accepts an input value, performs some calculations, and generates an output value.

Fuzzy Control for Robot Soccer

If robots are to collectively exhibit real time reasoning and collaboration, as soccer teammates must, they should be able to autonomously identify situations, evaluate them, and react accordingly. A fuzzy logic controller that mimics human reasoning by using rules, and incorporating concepts of imprecision in knowledge representations, is therefore ideal for this purpose (see Ross [3]).
The robot agents employ a fuzzy system (see Figure 1 below).

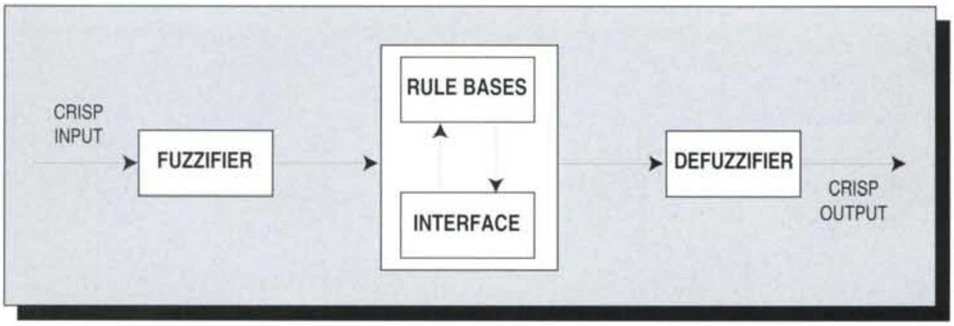


Figure 1. Process of a Fuzzy System.

The fuzzy system consists of several layers of hierarchical, interconnected fuzzy logic controllers (see Figure 2 below).

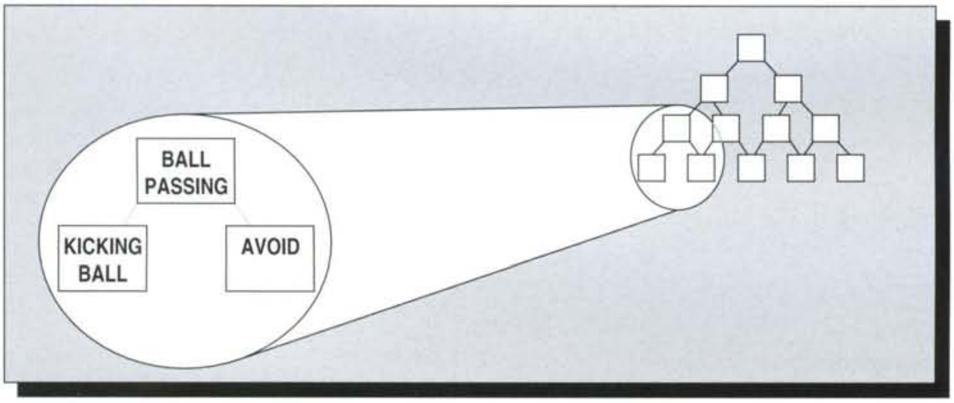


Figure 2. Soccer Robot Hierarchical Fuzzy Behavior Controller.

Each controller represents a behavior, and from its interaction with other controllers emerges the overall robot behavior. The lowermost modules correspond to the primitive behaviors displayed by the agents, such as obstacle avoidance and ball handling. As one advances up the hierarchical network, the fuzzy controllers encode more complex behaviors, affecting the response of the modules localized below them (feeding outputs directly to the inputs of lower levels or modulating their outputs). The more advanced functions are responsible for behaviors such as ball passing, attacking runs and defensive actions (see Mirabal, et al. [4]).

A genetic algorithm (discussed below) is used to evolve a fuzzy linguistic rule base starting from a fuzzy controller produced by the authors using a trial-and-error design approach. An initial population is formed by randomly altering a common-sense set of fuzzy rule bases created by the researchers (see Jeong and Lee [5], and Tunstel and Jamshidi [6]). Each search point or individual of the population, in our case a fuzzy rule, represents a candidate solution and is evaluated according to a fitness function. The fitness function assigns higher values to desired characteristics.

In the robot soccer application, data concerning the ball and players' positions, as well as strategic-play information is relayed to the robots via a radio link. Strategic-play information, generated by a laptop coach, consists of directives that affect the robot's higher-level behavior modules. The analysis of the game play and the emission of directives are realized by a different hierarchical fuzzy system running on the laptop. A second genetic algorithm will be employed to optimize the components of this tactical fuzzy system.

Fuzzy Behavior

Autonomous navigation behavior can be decomposed into several interconnected special purpose behaviors. The interconnection of the behaviors is a hierarchical network. It is hierarchical in the sense that the overall system behavior is decomposed into a bottom-up hierarchy of increasing behavioral complexity. The behavioral activity at a given level is a function of the behavioral activities at the level(s) below. The lowermost modules, referred to as the primitive behaviors, are responsible for the simple self-contained behaviors such as detect, move and kick. The intermediate behaviors, located in the mid-section of the hierarchical structure, interact with the primitive behaviors to perform specific tasks, such as score, block, and intercept. The primitive behaviors map inputs to control outputs (typically common actuators), while higher-level behaviors act as fuzzy decision systems which map goal and strategy information to dynamically adaptive weights associated with each primitive behavior (see Tunstel and Jamshidi [6]).

Evolving Intelligent Robots Through Genetic Algorithms

Genetic algorithms (GAs) use the concepts of Darwin's theory of natural selection and evolution. The theory is based on the assumption that the existence of all living things is based on the rule of "survival of the fittest" (see Ross [3]). New classes of living things come into existence through reproduction, crossover, and mutation among existing organisms. First, an initial population of different possible solutions is created by randomly altering a common-sense set of fuzzy rule bases created by the researchers. Each search point, an individual of the population (in our case a fuzzy rule), is tested for its performance according to a fitness function. Among the possible solutions, a fraction of the solutions is chosen, according to a cutoff value, to succeed into the next generation while the others are eliminated. This is the concept of survival of the fittest. The selected solutions undergo the processes of reproduction, crossover, and mutation to create a new generation of solutions that are expected to perform better than the previous generation by combining their traits (parameters). New generations are produced and evaluated repeatedly until there is convergence within a generation. First, in our application, the parameter set of the problem that characterizes the solution is coded in the form of a finite string of bits. For example, we create independent bit strings for the antecedent and the consequent of the fuzzy if-then rules, and then the strings of each parameter are concatenated to make one string that represents the whole parameter set. The length of the bit strings is based on the handling capacity of the computer being used. The strings are created randomly and then decoded, or mapped, into a set of parameters they represent. This set of parameters is then passed through a numerical model of the problem that yields a solution based on the input set. The quality of the solution is evaluated according to a fitness function whose value is assigned to the string. The *fitness values* of each string in the population are determined and an average is computed. The *relative* fitness of each string is determined by dividing the fitness value of each string by the average fitness value. Reproduction is the process by which strings with higher fitness values are duplicated in the next generation. The total number of strings must remain constant for computational efficiency, therefore strings with fitness values below the cutoff are eliminated. After reproduction, crossover occurs between two new randomly selected strings at a random location (see Figure 3). The bits to the right of the location are interchanged as a way of information exchange and the combination of desirable qualities. These new strings comprise the new generation and the process of decoding and evaluating is repeated.

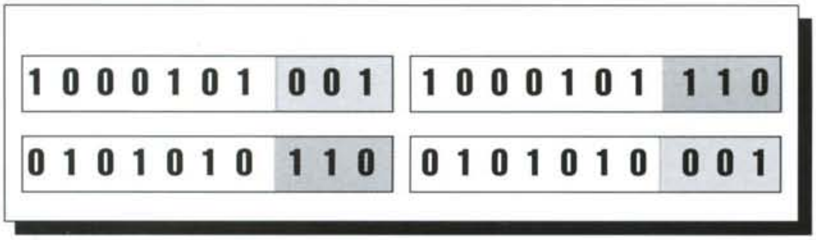
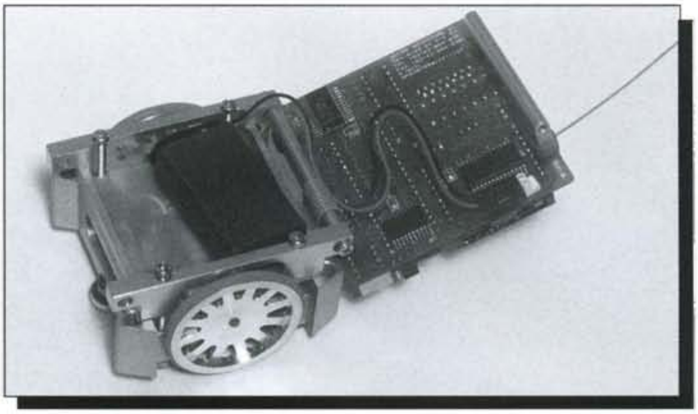


Figure 3. Crossover in strings.



Soccerbot with cover and controller board off.

Although most of the searching power is involved with reproduction and crossover, mutation is important when an optimum solution cannot be found. Mutation changes the value at a certain string location and is rare. It occurs perhaps once in a thousand bit string locations. In any problem space there are an infinite number of solutions, and genetic algorithms, augmented by mutation, search for a solution from a broad spectrum of possibilities.

Soccer Robot Sensor Interface

The robots are provided with information from a global sensor. It is provided by a remote, stand-alone, video-processing system composed of a Cognachrome (see Reference [7]) video processing board, an NTSC CCD camera, and a laptop computer. This global sensor extracts the positions of distinct objects contained in the camera's field of vision. These objects are associated with areas of different color (see Cognachrome [7]). Then the information extracted from the video frames is transmitted to the robots in the form of approximate locations of the objects. This configuration is adequate for our test-bed application since the teams, the field and its boundaries, and the ball have different colors.

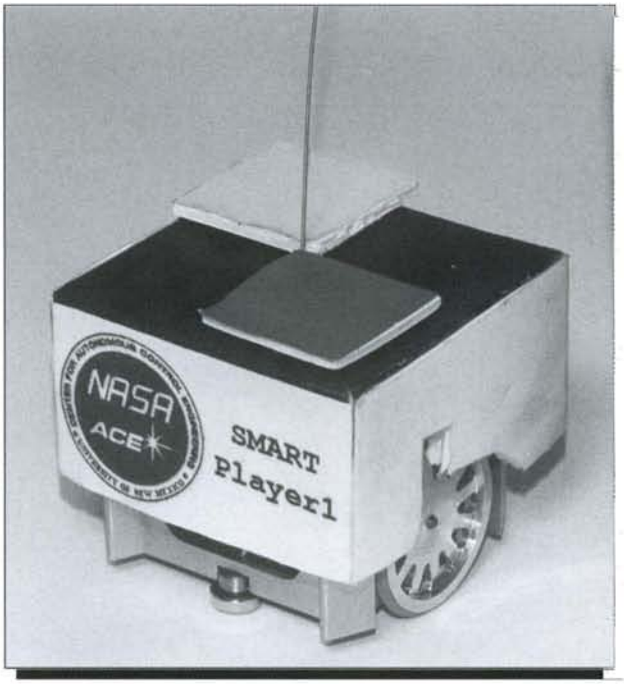
The sensor processing routines extract higher level information (such as distance-to-ball, possession-ball, distance-to-opponent, direction-nearest-obstacle, distance-nearest-obstacle, distance-ball-to-nearest-agent, speed-agent) from the position information provided by the global sensor. This data is then stored in a buffer area that is accessible to all fuzzy behavior modules. Between global information broadcasts, the current positions contained in the local buffer may be approximately updated by dead-reckoning methods.

Although the global update cycle is short, the game play speed is high, so dead reckoning may lead to appreciable positioning errors.

The mechanics of the environment perception and communication is illustrated in Figure 4.

Simulation

Simulation was used for creating and validating efficient rules for the robots. We used the Soccer Server Simulator for our simulation (see Noda [8,9]). The Soccer Server consists of two programs, "soccerserver" and "soccermonitor." "Soccerserver" is a server program that simulates movements of the ball and players. "Soccermonitor" is a program that displays a virtual field on the computer running the system. Client programs control our virtual player's actions and connect to the Soccer Server Simulator by a User Data Protocol (UDP) socket. The client sends commands to control an assigned virtual player and receives information from the player's virtual sensors.



NASA's Center for Autonomous Control Engineering (ACE) soccer-playing robot.

The communications between the server and its clients are composed of control commands and sensor information. There are six control commands: move *(x, y)*, *turn moment*, *dash power*, *kick (power, direction)*, *say message*, and *change-view (angle-width, quality)*. There are two sensor information commands: *see* and *hear*. Other parameters used are coach commands (a privileged client takes the place of the referee module in the server), object names, and play modes.

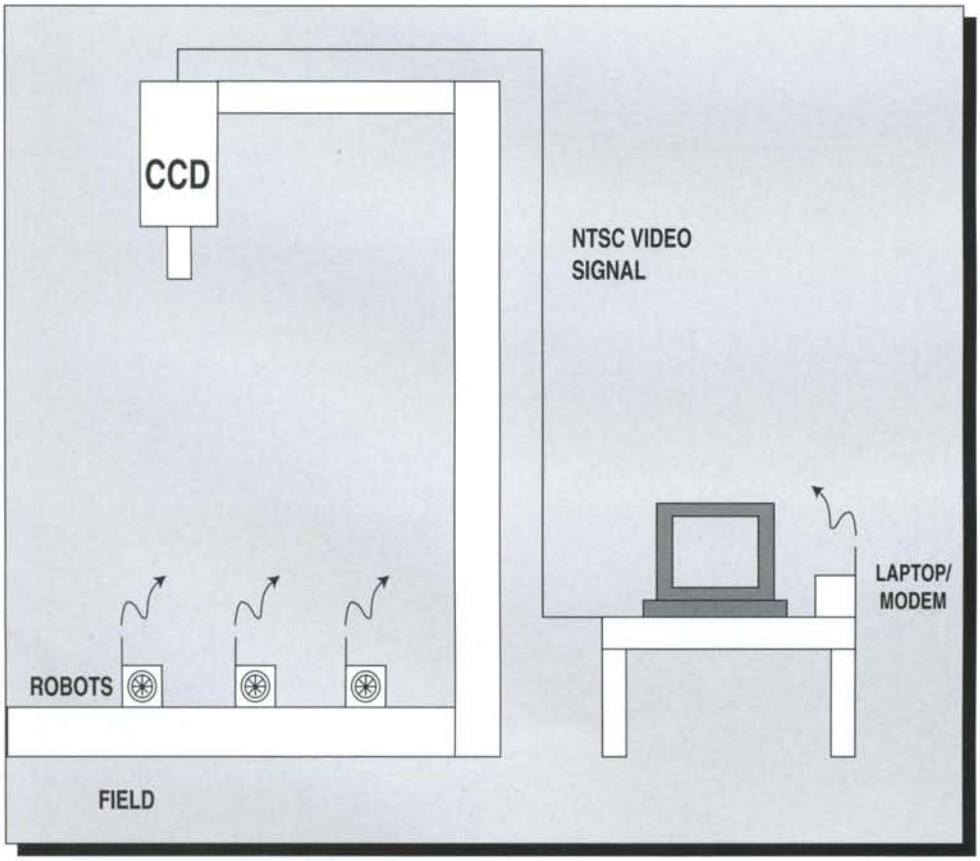


Figure 4. Illustration of Environment Perception and Communication.

All communication between the server and each client is done by using ASCII strings, permitting clients to be written in any programming language (C, C++... etc.). More detailed information about the Soccer Server is available at its web-site (see Noda [8]).

The Future of Intelligent Robots

Soccer-playing robots are the tip of the iceberg in the rapidly developing field of soft computing. Robot builders who accept the challenge will find many exciting and worthwhile applications for this technology. As more builders and teams enter robot soccer competitions, the enthusiasm and experience generated will open opportunities for finding team robotic solutions applicable to current and future problems.

Robots employing basic teamwork to complete complex tasks are, no doubt, in our future. Some interesting applications for cooperative robots include unknown terrain exploration, hazardous facilities maintenance and human support roles. Fuzzy logic and fuzzy control, with an application to fuzzy-genetic algorithm behavior control, could well lead the way in these endeavors. The technology introduced here places an emphasis on the most popular of all its applications - control. Fuzzy controllers are simple to implement in a laboratory environment, either on a personal computer or on a chip-level board. Today, fuzzy system technology is widely used in control, pattern recognition, medicine, finance, and marketing applications. It can be used as a tool for the solution to problems where a mathematical model is neither available nor feasible. Finally, new avenues should be opened for new software design and analysis of control systems using the power and efficiency of fuzzy logic, neural networks, and genetic algorithms. One such effort is NASA ACE's SoftLab© software environment, in which all soft-computing paradigms are being utilized for identification, control, simulation, and real-time implementation (see Akbarzadeh, et al. [10]).

RS&T

REFERENCES

[1] Tunstel, E. "Mobile Robot Autonomy via Hierarchical Fuzzy Behavior Control," Proc. of 6th Intl. Symp. on Robotics & Manuf.(ISRAM'96), 2nd World Automation Congress, Montpellier, France, May 1996, Vol.3, TSI Press, Albuquerque, NM.

[2] Zadeh, L.A. (1965), "Fuzzy Sets," Information and Control, Vol. 8, pp. 335-353.

[3] Ross, T. J., Fuzzy Logic with Engineering Applications, McGraw-Hill, Inc., New York, NY, 1995.

[4] Mirabal, S., Padilla, D. and de Oliveira, M.A.A., "A Hierarchical Fuzzy Controller Based Autonomous Soccer Player," Proc. 3rd World Automation Congress, Anchorage, Alaska, USA, May 1998, Vol. 7, TSI Press, Albuquerque, NM.

[5] I.K. Jeong and J.J. Lee, "Evolving Soccer-Playing Multi-Agents Using Modified Genetic Algorithm," Proc. of IEEE ICRA '97, Albuquerque, New Mexico, USA, 1997, pp. 3294.

[6] E. Tunstel and M. Jamshidi, "On Genetic Programming of Fuzzy Rule-Based Systems for Intelligent Control," International Journal of Intelligent Automation Soft Computing, Vol.3, No.1, 1997.

[7] Cognachrome User's Manual, Newton Research Labs.

[8] Noda, Itsuki, "Soccer Server Manual," http://ci.etl.go.jp/~noda/soccer/server/Documents.html

[9] Noda, Itsuki, "Soccer Server: a simulator of Robocup," JSAI AI-Symposium 95: Special Session on Robocup, December, 1995.

[10] M.R. Akbarzadeh-T., S. Shaikh, J. Ren, K. Kumbla, and M. Jamshidi, "Soft Computing Made Easy," Soft Computing Multimedia, and Image Processing, TSI Press, Vol. 8, pp. 849-854, Albuquerque, NM, 1998.

FIRST Robotics 1999 Competition Structure

The FIRST (For Inspiration and Recognition of Science and Technology) Foundation is a non-profit organization founded in 1989 by entrepreneur Dean Kamen. Headquartered in Manchester, New Hampshire, its purpose is to teach young people to appreciate technology. The FIRST Robotics Tournament was created to provide high school students with the opportunity to experience, hands-on, the possibilities of the world of engineering and science. It is a tournament in which teams of students, mentored by professionals, design and build specialized robots that compete against each other. Participation in the tournament has grown significantly since its inception in 1992, when 28 teams competed. This year, the number of teams approaches 300. The rules of the competition change each year, challenging the ingenuity of teams to the maximum. In the January issue of RS&T, we reported on the 1998 tournament, and in the May/June 1999 issue, we will present an article on the 1999 National Championship series. This article explains the fundamentals of the 1999 games.

Competition Structure

The 1999 competition is comprised of two phases, **qualification** and **elimination**. Ample opportunity is provided for a team to advance. The opportunity is enhanced and made more interesting by virtue of the **teaming**, or **alliance** concept. Four teams, paired to create alliances of two teams apiece, participate in each match. Each alliance works together to win a match, and alliances win or loose as a unit, but accumulate points individually. Thus, although it is always nice to win, the key element for getting through the qualifications and advancing to the elimination rounds is the individual team point total.

- Each alliance competes using two team-built robots, four robot operators, two human players, and four coaches, and the teams are arrayed against each other on opposite sides of the rectangular playing field.

- In the qualification phase, alliances are formed just prior to the start of each match, and last for the duration of that match. Teams change allies after each contest, and all teams play an equal number of qualifying matches. At the conclusion of this phase, the cumulative point result of each team is used to rank the teams. The eight highest-ranked teams qualify to proceed to the elimination matches.

- In the elimination phase, each of the eight qualified teams selects an ally from the remaining teams, and the eight alliances stay together throughout the phase. The winning alliance is determined in this best 2-of-3 series of elimination matches.

The Playing Field

The field is a carpeted rectangle about 24 by 28 feet. Around the perimeter of the field are four stations for human players and four stations for robot operators and coaches. Each alliance occupies a side opposite its opponents.

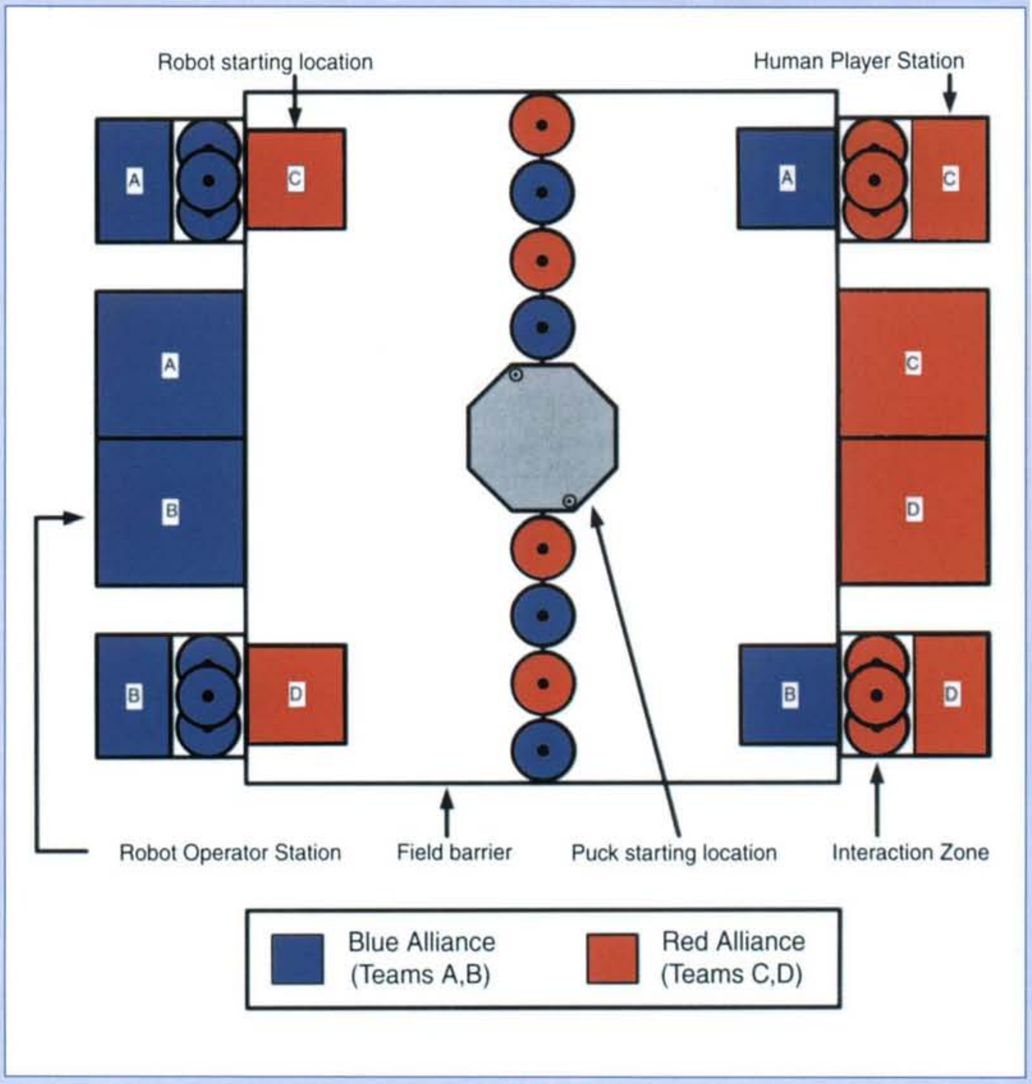


Figure 1. The 1999 FIRST Playing Field is a rectangle about 24 feet wide by 28 feet long

The Tools

Tools of the game, in addition to the two robots per alliance, are *Floppies* and a *Puck*.

- *Floppies* are pillow-like donut-shaped disks almost 26 inches across. Ten Floppies are assigned to each alliance, and they are color-coded.

- The *Puck* is a low octagonal platform 5 feet 6 inches across, that rolls on caster wheels.

The Setup

Prior to the start of each match, the robots, *Puck* and *Floppies* are placed on the field and at the player stations as shown in Figure 1.

Gameplay

Matches are two minutes long in all phases of gameplay, up to and including the National Championships. Speed is always of the essence, and teams must be able to adjust to team pairings, station position, and color assignments on very short notice. A team's ability to adapt quickly and remain poised is essential.

Points are scored by picking up *Floppies* with the robots, and placing them on the *Puck*, or elevating them over the playing field. One point is awarded for each *Floppy* placed on the *Puck*, and for those elevated over the playing surface, but at a height of eight feet or less. For each *Floppy* elevated over eight feet, three points are awarded.

Any robot actually on the *Puck* at the end of play multiplies its team score by a factor of three, and a score multiplier of two is awarded if the *Puck* is positioned wholly within the opposing alliance's side of the field at the end of play. Provisions are made for tie-breakers.

FIRST Robotics

Gunn High School FIRST Team

Students Take Charge of Engineering As a Team Designs and Builds Its Robot

by Floyd Painter

The 1999 FIRST (For Inspiration and Recognition of Science and Technology) National Championship tournament will be held at Disney World's EPCOT Center, Orlando, FL during April 22-24. Tournament activity commenced January 9th with the kickoff, and provision of the game description and rules to participating teams, who then had six weeks to design and build their robots. All seven regionals have been held, and teams are planning their trips to EPCOT. Robot Science & Technology decided soon after kickoff to observe and write about the developmental phase of a team effort. The editor visited Harry M. Gunn High School in Palo Alto, CA a little over two weeks into the design and build phase, and writes about the team and its faculty sponsor, their robot *G-Force*, and a mentor who helped them through the process. This article continues RS&T's coverage of FIRST activities. For information on last year's tournament, see FIRST Robotics, Partnering Industry and Youth, on page 6 of the January issue of RS&T.

Gabe Rotberg, Haru Yamamoto, Kia Seng Tang, Alvin Cheng and Zac Fisher are members of the robotics team at Harry M. Gunn High School, Palo Alto, CA. Here, they ponder the intricacies of chain drive over a breadboard model of the base of their robot, G-Force.

At about three hundred high schools (and at least one elementary school) across the United States, robotics team members, along with their faculty sponsors and mentors, are savoring the experiences gained during the regionals. Teams are by now very familiar with their mechanical charges, and have assessed their chances for success in the national championship series. Two-hundred-twenty teams, their driving skills and teamwork polished and strengthened in the intense competition of the regionals, will take their robots to EPCOT. Not all teams that entered the tournament compete at EPCOT, and there are many reasons for that. If a team is not going this year, you can bet they will try to make it next year.

But the majority do take on the big one because it extends the thrill of competition for another whole cycle, and after all, a national championship is at stake. Maybe things weren't just right in the regional, but the robot performed well enough, and you learned a great deal. One thing you learned is that on a given day, you can win even under the most adverse conditions, so persistence is important. Victory can come via a superior driving effort, a more efficient robot or the unexpected breakdown of an opposing robot (aren't breakdowns always unexpected?). These things and many more can affect the outcome of a match, but basically it is the spirit of FIRST that inspires most teams to press on, no matter what happens.

The founders of FIRST have done a great job of making it clear to all participants that competing, in the best way that you can, is the most important thing about the tournament. Winning is the goal, and teamwork enables success. This is true in all phases of the competition, from the time of interpretation of the game rules, through design and fabrication of the robot, to actual game play. But if the ultimate goal is not achieved, the pleasure of competing well, and displaying true sportsmanship is reward enough. It becomes the force that will bring you back next year, resolved to do better.

Gunn High School and *G-Force*

The robotics team at the Harry M. Gunn High School in Palo Alto, CA probably typifies the organizations participating in the 1999 tournament. *Robot Science & Technology* visited the school when the team was in its third week of robot construction, and found an eager group of students hard at work designing and building ***their*** machine.

The faculty sponsor at Gunn is physics teach Bill Dunbar, who obviously enjoys guiding his 60 students in meeting the challenges of FIRST competition. As he pointed out, his is "pretty much a student-run, student-oriented team. They take charge of the whole operation. One of my roles is to make sure they don't have a lot of obstacles." He went on to say, "Robotics in general is a really good way for team members to tie together a lot of the technologies they've learned about; electronics, mechanics and structures, for instance." Team members that RS&T talked to felt that Dunbar was a great faculty sponsor because he let them work out their problems by themselves. He only weighed in when he saw that they were on the verge of taking an inefficient path to problem solution. He is "Cool."

A Quarterback, Not A Wallflower

Dunbar pointed out that the students made all the critical decisions right from the start, including the basic one about the role for which the robot should be designed. With the alliance concept, teams work with an unknown quantity, another robot about which they know nothing. "In our two previous FIRST competitions, we noticed that whether it's our robot or someone else's, they are not always reliable. We'll probably have at least one match where our partner does nothing but sit and make grinding noises. The students had a choice early on, about either making a robot that was a good partner, or one that could do everything. They recognized that making a good partner robot, say a linebacker, would be easier, but they wanted to make a quarterback instead, one that does everything and takes charge," he said.

Warming to the subject, Dunbar continued, "With this year's rules, you can have a robot that does one thing well, like taking other robots out of the competition and pushing them around, filling a kind of spoiler role. The partner robot then has to score most of the points. If somebody in the top eight teams recognizes that as the strong suit of your robot, you can get into the finals by being a kind of wallflower, and be picked for the elimination phase for being a good complement for the other robot. The students didn't want to do that; it wasn't what they wanted to get out of this tournament."

Concurrency in Design and Construction

The Gunn High School robotics team operates out of the school's old woodworking and metal shops, adjacent to each other in the industrial arts building. Sadly, the well-equipped facilities are not used for their intended purposes due to curriculum cutbacks. In the center of the old woodshop sits a full-size breadboard model of a part of *G-Force*, the 1999 robot. It is mostly made of wood, and surrounded by young people. Not far away is the stripped-down robot from the 1998 competition, a silent sentinel undoubtedly watching closely, but unable to give advice as the work on its successor proceeds at a deliberate pace. Bill Dunbar addressed this work-in-progress: "It's a mechanical breadboard, and they are trying various concepts to see if they can get something to work. We are designing, building and testing in parallel. We know the kind of basic functionality of the robot and we're designing around that. Since we only have six weeks to finish the whole project, we don't have time to do research and design later. We've got to do everything at once."

Encouragement Helps

Dunbar spotted something he liked at the breadboard: "Hey that's not bad! Wow, good job you guys! That's the best I've seen yet!" The team members beamed at the encouragement as a *Floppy* was maneuvered into position just as they had planned.

I asked if Gunn team members from previous years had gone on to college to study things like artificial intelligence, computer science and engineering. "Yes," he said, "and it has prompted students to go in directions we had not anticipated. One student in particular liked to videotape team activity, a requirement for the FIRST Chairman's Award, and she decided to go into video production. We've had a couple of students who have decided that they want to go into manufacturing, specifically to become machinists. It's really refreshing to hear a youngster say, 'I just like to build stuff. I want to make stuff.'"

Bill Dunbar is a veteran of three FIRST competitions, and the initial exposure was in 1997: "We were so proud of ourselves just to have a robot that worked. That first year will always be kind of special to me because everything was so unknown. We had just a handful of kids and didn't know what the heck we were doing. It taught me something. I learned that high school students can accomplish tasks far more difficult than what we normally give them credit for Given the resources and something they are enthusiastic about, it's difficult to see any way to stop them. A couple o lessons learned from our experiences are that appearance doesn't matter much, and if you build something right the first time, it won't come back to haunt you."

The Base and Arms Take Shape

In the old metal shop, we approached a group of five young men working on the robot's full-scale base breadboard. Zac Fisher told us they were "working on a solution for mounting the robot motor, and fitting a sprocket and chain to a drive wheel." The other team members, Gabe Rotberg, Haru Yamamoto, Kia Seng Tang, and Alvin Cheng were all involved hands-on, confirming that the chosen path was the best way to solve the problem.

We then approached two other team members working in the metal shop, Katrina Leni and Brad Phlum (Brad is a co-captain of the team). They were fabricating an aluminum lifting arm for the robot, taking their time and doing it right. The arm looked as if it had been professionally fabricated. As we watched, Katrina fashioned a wooden insert for the hollow, tapered piece of aluminum, pointing out that "it would add significant strength to the arm." The insert fit

nicely into the arm. I asked Brad how the team was progressing. He said, "We're doing pretty well. We had an ambitious production schedule, and we were off schedule for awhile, but now we've caught up. We were even able to do some mock-ups on schedule last week."

Back in the wood shop, I talked to the other co-captain, Daniel Lehrbaum, who was working with a team gathered around the main body breadboard model. Daniel is a senior at Gunn, and he hopes to be admitted to Carnegie Mellon University next fall. I pointed out that CMU had a very active robotics program, and he gave us a knowing smile. Bill Dunbar commented that two students from last year's team were now at CMU.

Jenny Li, Osbert Feng and Alvin Cheng were hard at work on the pickup and lifting apparatus. Satisfied, they turned their attention to a subassembly that seemed to be a source of concern.

Katrina Leni helps fashion an arm for G-Force.

Ken Phlum, Brad's twin brother, awaited his turn to get near the machine, into the hands-on action.

One thing you could not help noticing was that every move was discussed by the team before action was taken. Something else was easy to notice; everyone was having fun.

A Programmer's Work Is Never Done

Benjamin Hebert, a sophomore, was testing the controller box supplied by FIRST to each competing team as a part of the robot kit. Basically, it is a microprocessor with a *Parallax* stamp, and Ben is responsible for programming it. It stores the code that determines movement of the robot parts when the two FIRST-supplied joysticks are actuated. The signals generated are sent to the robot via radio controllers. He pointed out, "It has a fair amount of capability, but they don't provide a lot of insight into how to program it, so for me it is a major project. It seems like every other day we have to reprogram because it's not right." Of course, Ben is caught in a continual learning loop, because his programming must change as the robot design changes, and at this point the design is in a state of constant flux. Ben has two more years of FIRST activity ahead of him, and seemed pleased at the prospect.

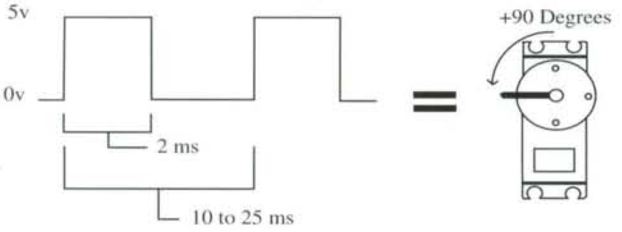
Engineering Support

The Ames Research Center of the National Aeronautics and Space Administration (NASA), is a sponsor of the Gunn High School team. It is located on the former Naval Air Station at Moffett Field nearby. NASA Ames also sponsors several other teams in the area by lending engineering expertise to their projects. Ames also hosted the FIRST California Regional in February. Another major Gunn supporter is the Xerox Palo Alto Research Center (PARC). Some of the support was informal, and engineers from NASA, PARC and other support/mentoring organizations dropped in on the activity from time to time and offered advice. As Bill Dunbar pointed out, there was also a more formal arrangement: "Every Saturday night we had a design review, where we stepped back and calculated how things were going, and our supporting engineers showed up for that. The students

(continued from page 8)

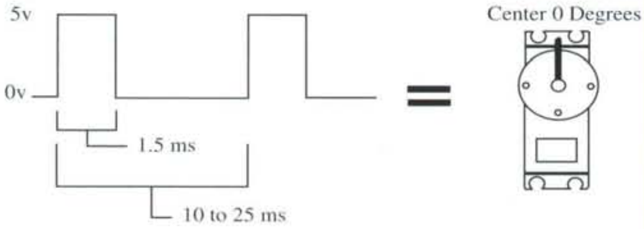
Fluffy Controller Board Schematic

FIGURE 2



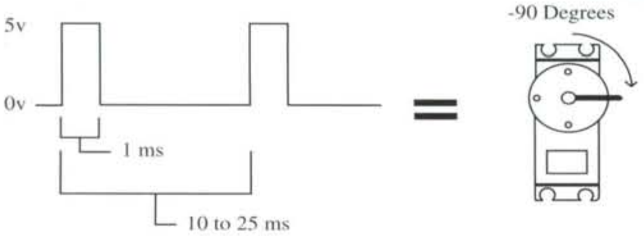
For example: 2 ms high pulse = servomotor's +90°,

FIGURE 3



1.5 ms high pulse = servomotor's 0° center position,

FIGURE 4



and 1 ms high pulse = servomotor's –90° position.

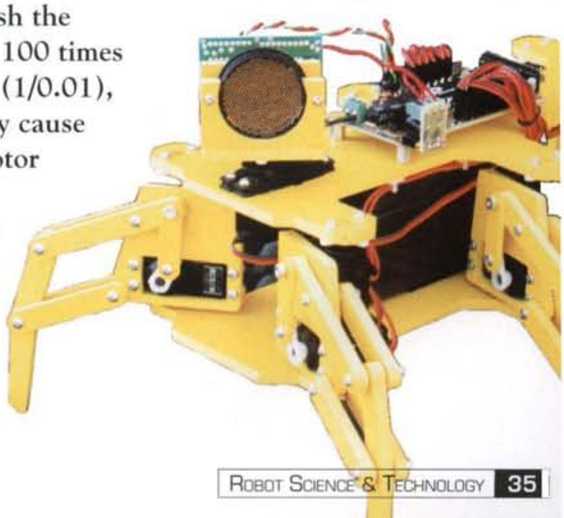
During the refresh period, the servomotor compares its actual position, as fed back by the potentiometer in the servomotor, to the position command signal in the pulse stream. If the two do not match, the servomotor will then move to the position commanded. Without the presence of a refresh signal the servomotor would move to a position and stay there until some outside force, such as using your fingers to turn its output horn, caused it to move. With the refresh process, if you tried to move the servomotor manually it would fight to stay right where it is. Then the only way to move it is to send it a new position command.

Code Example

In the following code example, the *BasicX* chip will send out two separate pulse signals. The servomotor wired to pin 21 will move to its 1.5 ms or center position and the servomotor wired to pin 22 will move to its 2 ms or +90° position (see Figure 2):

```
Dim Servomotor1 As Single              'Make Servomotor1 a
                                        Floating-Point Variable

Dim Servomotor2 As Single              'Make Servomotor2 a
                                        Floating-Point Variable

Sub Main()                             'The Main Subroutine

Servomotor1 = 0.0015                   'Store 0.0015 in the
                                        Servomotor1 variable

Servomotor2 = 0.0020                   'Store 0.0020 in the
                                        Servomotor2 variable

Do                                     'Top of our Do Loop

Call PulseOut(21,servomotor1,1)        'Send a.0015 second (1.5 ms)
                                        high pulse out pin 21

Call PulseOut(22,servomotor2,1)        'Send a.0020 second (2.0 ms)
                                        high pulse out pin 22

Call Sleep(0.015)                      'Do nothing for 15 ms this
                                        will make the low pulse

Loop                                   Bottom of our Do Loop

End Sub                                Marks the end of this
                                       subroutine
```

Once running, this code will instruct the *BasicX* chip to send a 1.5 ms high pulse out pin 21 (see Figure 3) and a 2 ms pulse out pin 22 . A 1 ms high pulse would move the servomotor to the -90° position (see Figure 4). The Sleep(0.015) command is used to set the duration between the high pulses thus giving us our low duration. Now servomotor1 has a refresh rate of about 17 ms (2 ms high + 15 ms low). By lowering the value of the Sleep command to 0.010 you would increase the incidence of the refresh cycle by changing the total value from 17 ms to 12 ms. At 17 ms the servomotor would refresh its position at a rate of 58.8 times a second (1/.017). At 12 ms, the refresh cycle occurs 83.3 times per second (1/.012). **(Note: a refresh rate of 10 ms (2 ms high + 8 ms low) would refresh the servomotor 100 times per second (1/0.01), and this may cause the servomotor to heat up and damage itself.)**



In the following code example, the multitasking ability of the *BasicX* to control servomotors, stepping them through their 1 ms to 2 ms position range, is demonstrated:

```
Dim Stack1(1 to 40) as Integer        'Make a block of memory for
                                       our Task to run with
Dim Servomotorpos As Integer          'Make Servomotorpos an
                                       Integer Variable
Dim Neg as Integer                    'Make Neg an Integer Variable
Sub Main( )                           'Start of the Main Subroutine
Servomotorpos = 1000                  'Make the Variable
                                       Servomotorpos = 1000
CallTask "Servomotor,"Stack1          'Start our Task Servomotor
                                       up with its memory block
Do                                    'Top of the Do Loop
Servomotorpos = Servomotorpos + 1     'Add one to Servomotorpos
                                       current value
If Servomotorpos = 2000 then          'If Servomotorpos = 1000
For Neg = 1 to 1000                   'Give Variable Neg a count
                                       range of 1 to 1000
Servomotorpos = 2000 - Neg            'Take the current Value of
                                       Neg away from
                                       Servomotorpos
Call Sleep(2)                         'Do nothing for 2/512
                                       of a second
Next                                  'Add 1 to Neg Count and go
                                       to for till Neg = 1000
End If                                'End If statement usage
Call Sleep(2)                         'Do nothing for 2/512 of
                                       a second
Loop                                  'Go to Do on top and
                                       start over
End sub                               'Marks the End of this
                                       subroutine
Sub Servomotor()                      'Start of the Servomotor Task
Do                                    'Top of the Do Loop
Call PulseOut(21,Servomotorpos,1)     'Send a high pulse out pin 21
                                       Servomotorpos is the length
Call PulseOut(22,Servomotorpos,1)     'Send a high pulse out pin 22
                                       Servomotorpos is the length
Call Sleep(8)                         'Do nothing for 8/512
                                       of a second
Loop                                  'Bottom of our Do Loop
End Sub                               'Marks the end of this Task
```

Since the subroutine "Servomotor" is a Task, once it is started it will continue to put the ever-changing value of Servomotorpos out pin 21 and 22 as a high-going pulse regardless of what the Main subroutine is doing. As you can see by this example, 'tasking' is a very valuable feature of the *BasicX* microprocessor. Tasking allows you to have multiple program loops running at the same time and exchanging information, this greatly improves the speed and performance of your program.

The Controller Board

Fluffy's controller board (see Figure 5) uses one chip to run all of its servomotor outputs as well as the joystick port and four multipurpose I/O lines. With the exception of the ground connection, which is common throughout the controller board, the board's power system is divided at the voltage-input connector into two sections. Section one takes the +5.2 to +7.2 volt input and runs it through an LM2931 (a +5-volt low dropout regulator). The regulated +5 volt output is then smoothed out by a 16v 330μF capacitor before being routed to the Vcc pins of both the *BasicX* and 32K serial peripheral interface (SPI) EEPROM. The regulated +5 volt supply is also used by pin 1 of the joystick interface connector. Section two of the power system directly routes the +5.2 to + 7.2 volt input to the center pin of each servomotor connector.

Pin1s from each of the 13 servomotor connectors are routed though a 470 Ohm resistor to their own I/O pin of the chip. Pin 2s of all the servomotor connectors tie to the + battery supply-in-connector on the controller board. Pin 3s of the servomotor connectors all tie to the common ground of the controller board.

The I/O lines that run the *Polaroid* sonar unit come from pins 32 and 28 of the *BasicX* chip; both pins are routed through their own 470 Ohm resistor. I/O pin 28 drives the initiate (INIT) input of the sonar board, whereas pin 32 is used to receive the ECHO output signal from the sonar unit. I/O pins 1 through 4 are used to read the position information from a computer joystick plugged into the joystick port of the controller board. I/O pins 3 and 4 are each wired though a 470 Ohm resistor and used in conjunction with the Rctime command to read the X and Y position value of the joystick handle controller. I/O pins 1 and 2 are both tied high (connected to the +5 volt output) through a 100K Ohm resistor and used to read the two button outputs of the joystick. One side of the reset button (not shown on the schematic) is tied to ground and the other is connected through a 100K Ohm resistor to pin 9 of the *BasicX* chip.

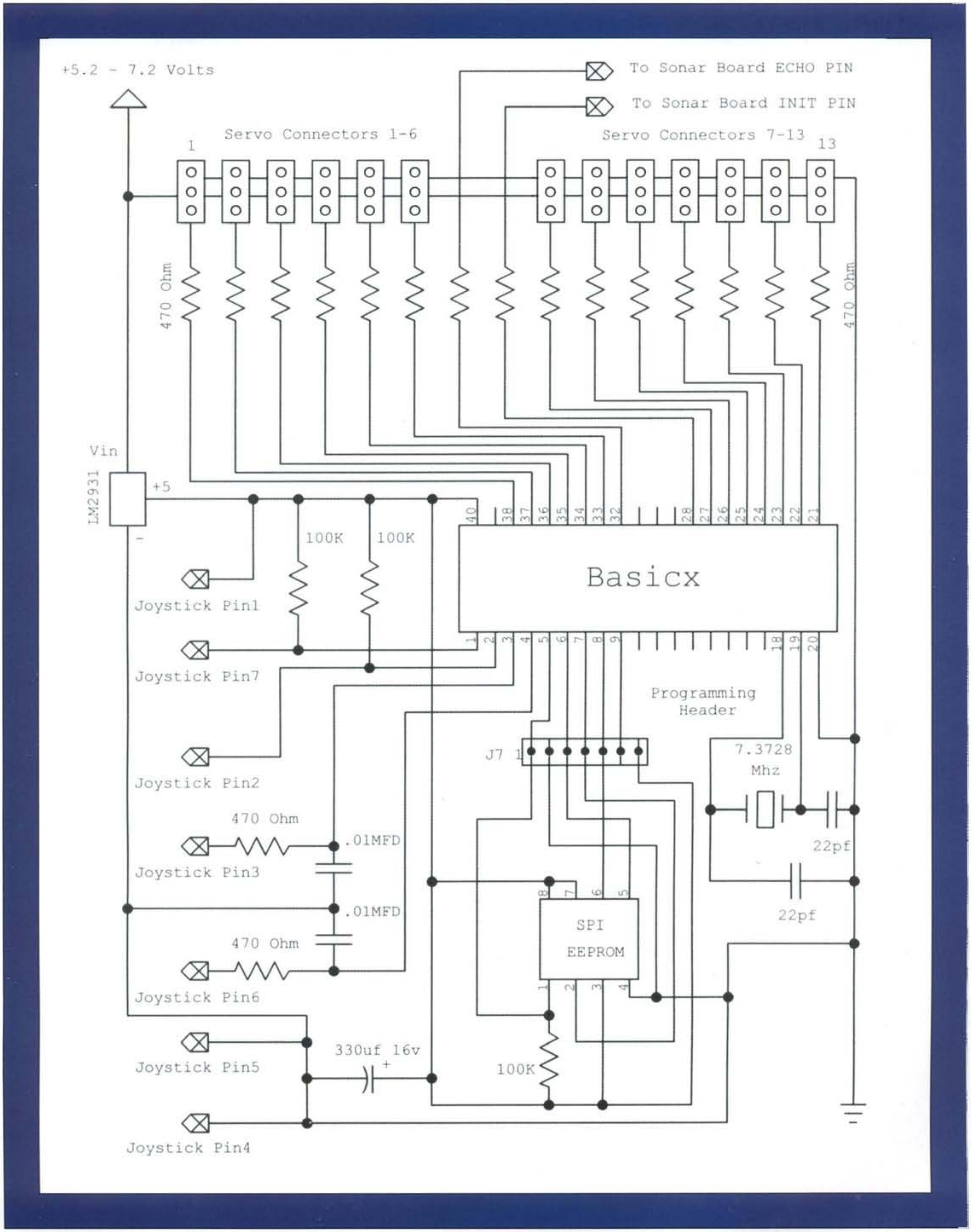
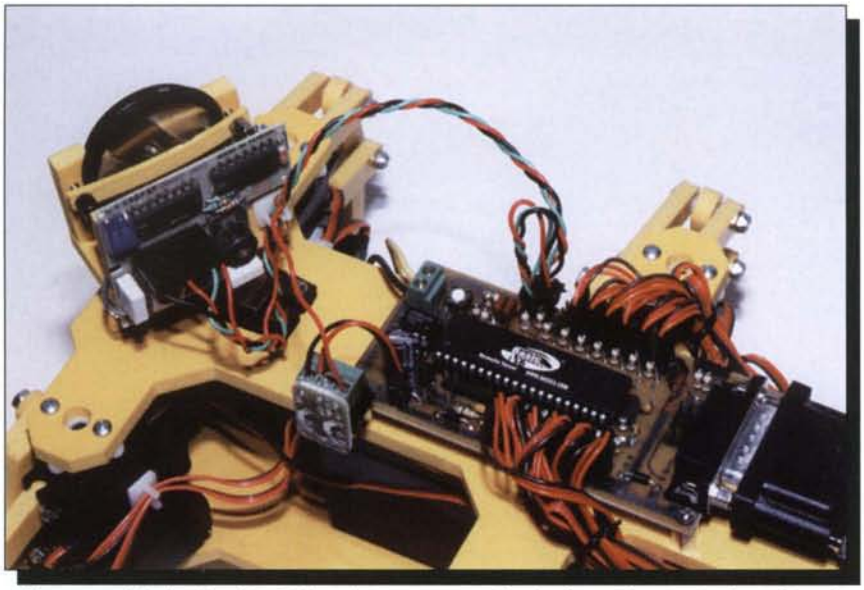


Figure 5. Schematic for Fluffy's Controller Board.



This view of the back of the Polaroid sonar sensor clearly shows the suite's driver board.

How the *Polaroid* sensor system works

The *Polaroid* sonar unit consists of two parts, the sensor module and the driver board. Here's how the sensor works. Once initiated (INIT), the driver board sends a series of high voltage pulses to the sensor unit. The sensor unit then sends these pulses out as sound waves. If any of these sound waves strike an object they are reflected back and received by the sensor, which sends a return pulse (ECHO) back to the driver board. The difference in time between when the pulse was sent and received tells you two things. First, there is something there because the pulse had to reflect off an object to come back. Second, if you measured the time interval that elapsed for the pulse to return after it was sent, you can use this value to determine how far away the object is. To take advantage of this, set the driver board's INIT high (which sends out one pulse set), and wait for the ECHO to go high (ECHO goes high when the sensor receives a valid return pulse). The total time from setting INIT high and ECHO line going high is your distance value. Once ECHO goes high and the distance value is obtained, INIT must be set to low before the sensor can be used again.

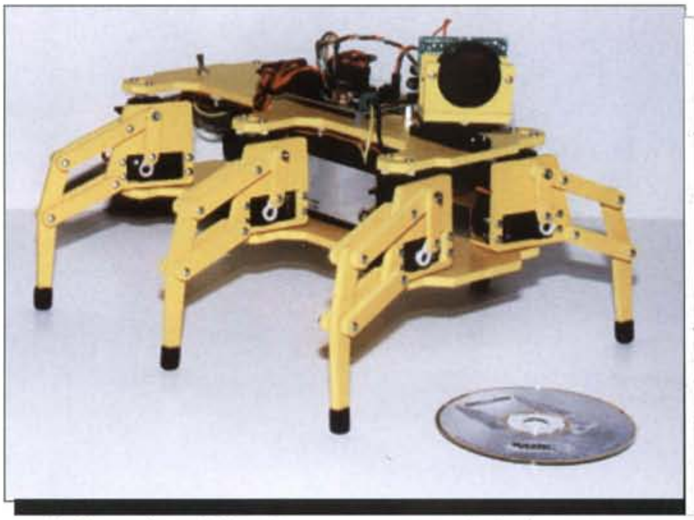
Wiring up the Polaroid sonar unit.

Since there are about four versions of the *Polaroid* sonar board with three totally different pin-out configurations, I elected not to include any specific pin-out diagrams in this article. Four connections must be made to make use of most sonar boards. Power (+5 volts), and ground are two of these connections. Since the sonar driver board can pull up to 2 Amps during the brief period of time that it is sending out pulses, it is recommended the board be powered with its own 1-Amp 5-volt regulator (these regulators can supply a 2 Amp burst as needed). The other two connections are INIT and ECHO. INIT is wired to pin 28 of the *BasicX* chip through a 470 Ohm resistor. ECHO is tied to the sonar driver board's +5 volts via a 5K Ohm resistor and to pin 32 through a 470 Ohm resistor.

The Code

To help you gain a better understanding of the inner workings of *Fluffy's* code, it will be broken down into manageable pieces. The line "Sub Main()" marks the beginning of our first piece (the main subroutine). In the Main the first thing to do is call subroutine - center. This is where the center positions are assigned for all of the *Hexapod's* servomotors (you might need to play with these values as center positions vary slightly among servomotors). After assigning a center value to the servomotors, start the tripod tasks using the "CallTask" command. The two tasks just started operate all of *Fluffy's* servomotors:

- Tripod1 deals with three legs of the "tripod gait."
- Tripod2 deals with the remaining three legs, and feeds the position servomotor for the sonar unit.



Fluffy, with its Polaroid sonar sensor in right-look position, is shown with a compact disk for size comparison. The disk contains the BasicX compiler and environment for Windows 95/98.

The first thing each of these two tasks do is read the initial position variables for each of the leg segments (set by the center subroutine), and send those values out their assigned pins via the pulseout command. The usage for pulseout is "call pulseout (pin, variable, state)," so "call pulseout (21,L1,1) would tell the *BasicX* to send a pulse out pin 21. The length of the pulse would be determined by the value of L1, and 1 would tell it to send a high going pulse (+5 volts). Since both of these tasks will continue to run in the background until power is removed, you do not have to use the pulseout command again. All you have to do is change the value of the variable that is assigned to the servomotor you wish to move. Thanks to these two tasks, if you want to move the servomotor on pin 26 state in the code "L1 = 2000." Since Tripod1 task is always running in the background, the next time it comes around its loop it will detect and send the new value of L1 out as a pulseout command. The servomotor that is connected to I/O pin 26 will move to the position that the value of 2000 represents.

Joystick Operation

The next thing to do is determine if a joystick is plugged in. If a joystick is present, its pot2 is connected to pin 4 of the *BasicX*. If, when reading pin 4 of the *BasicX* a reading of 0 is obtained at the microprocessor, the program determines that there is no joystick present and "Task3" is executed. The robot begins running in autonomous mode (we will come back to this later). This is accomplished with the Rctime command. If any value other than 0 is returned, the microprocessor activates the joystick control mode.

This control mode reads the joystick's two pots, then branches to a subroutine that plays a "script" of servomotor positions that match the joystick position, i.e., if you move the joystick forward *Fluffy* will walk forward, and so on. The two joystick buttons also act as a means of control. If pushed, button one (I/O pin 1) will call the center subroutine and cause all the servomotors to move to their home positions. This is very helpful for initial servomotor hookup and alignment. If you press button two (pin 2), the *BasicX* program will call the low subroutine and cause the *Hexapod* to crouch. This has no real use, but it's really cool to watch! As mentioned earlier, if a joystick is not detected when the robot first boots up, the program calls Task3 and starts its autonomous mode. Once started, Task3 will make sure that the sonar servomotor initial position is straight-ahead. It then gets distance data from the sonar unit (I/O pin 28 and 32) to determine if there is any object in front of it closer than 23 inches. If no objects are detected closer than 23 inches, it calls the forward subroutine and starts the detection loop over again. If the sonar unit detects any object closer than 23 inches it will call the SonLook subroutine. The Sonlook subroutine will then take control of the variable 'Son' and direct the sonar servomotor to turn the sonar unit to the left and then to the right if there is still an object in its path. If after three attempts the Sonlook subroutine cannot find a clear path for the robot to travel, it will call the backward subroutine. After backing up, the robot will start looking for a clear path again.

Source Code

Fluffy's source code is extensive. It has been included in this feature as an addendum. Other useful information, including schematic drawings, are available at *BasicX.com*.

Conclusion

Fluffy evolved from a manually controlled walker, to an autonomous robot with sonar capability. As we have seen, it can be operated in either mode. The microprocessor is state-of-the art, but key components like the joystick and sonar suite were items found around the workshop. The stick is standard, available at any computer store, and the sonar suite was salvaged from an old camera. This demonstrates that elements of your construction do not have to be cutting edge or expensive to produce an interesting robot. The new microprocessor encourages experimentation, and there is plenty of real estate on the bot to accommodate other types of sensors. I hope *Fluffy's* story will inspire you to take on custom projects, and that you enjoy them as much as I have. If you have any questions or comments, I can be contacted at *chrish@netmedia.com*.

RS&T

***Chris Harriman** is an Applications Engineer at NetMedia, Inc. in Tucson, AZ. He has been a roboticist for over 14 years, and is also an amateur radio operator and private pilot.*

Resource List

The Complete ***BasicX Development System*** comes with one ***BasicX*** chip. A pre-built, tested and programmed ***Fluffy*** controller board, or a bare ***Fluffy*** controller board with schematic and parts source list are available. For additional information, contact:

NetMedia, Inc.
10940 N. Stallard Place
Tucson, AZ 85737
Phone: 520.544.4567
BasicX.com

The ***Hexapod*** 2 Kit Is available from:

Lynxmotion, Inc.
104 Partridge Road
Pekin, IL 61554
Phone: 309.382.1816
lynxmotion.com

The ***Polaroid*** Sensor and Driver Board are available from:

Acroname Inc.
Phone: 303.258.3161

Their Web-site contains very useful wiring and operation information on *Polaroid* sensors.

acroname.com

Addendum - Fluffy Source Code

```
'BasicX controller board
'Fluffy code v1 with sonar
'Compiled Size 1364 Bytes
'By Chris Harriman, 1998

Dim Stack1(1 to 40) as Byte     'Set 40 bytes to run task 1'
Dim Stack3(1 to 40) as Byte     'Set 40 bytes to run task 3'
Stack2(1 to 40) as Byte         'Set 40 bytes to run task 2'
Dim L1 As Integer               'Set variable L1 as integer
Dim L2 As Integer               'and so-on'
Dim L1i As Integer              'and so-on'
Dim L2i As Integer              'and so-on'
Dim S1 As Integer
Dim S2 As Integer
Dim S1i As Integer
Dim S2i As Integer
Dim Pot1 As Integer             'Joystick variable X Axis'
Dim Pot2 As Integer             'Joystick variable Y Axis'
Dim Son As Integer              'Sonar servo position variable
Dim Dis As Integer              'Sonar dist. measure variable'

Sub Main()

 Call Center position           'Set all servos to home position'
 Call Sleep(64)                 'Wait for 64/512 seconds'
 Calltask "Tripod1",Stack1      'Start tripod side one task up'
 Calltask "Tripod2",Stack2      'Start tripod side two task up'
 Call Putpin(4,0)
 Call Sleep(4)
 Pot2 = Rctime(4,0)
  If Pot2 = 0 Then              'If joystick is not present
  Calltask "task3",Stack3        start autonomous mode'
  Do
  Loop
  Else                          'Otherwise read joystick

  End if                         inputs'
Do
 Call Putpin(3,0)               'Discharge cap on pin3 to get
                                 ready to read pot1'
 Call Putpin(4,0)               'Discharge cap on pin4 to
                                 get ready to read pot2'
 Call Sleep(4)                  'Wait for caps to discharge'
 Pot1 = Rctime(4,0)             'Read value of joystick pot2
                                 on pin4'
 Pot2 = Rctime(3,0)             'Read value of joystick pot1
                                 on pin3'
 If Getpin(2) = 0 Then          'Read Joystick button 1'
 Call low                       'If joystick button 2 pressed
                                 go to sub low'
 Elseif Getpin(1) = 0 Then      'If joystick button 1 pressed
Call Center                      go to - Center'
 Elseif pot1 = 1 Then           'If joystick pushed forward
Call Forward                     go to - forward'
 Elseif pot1 > 400 Then         'If joystick is not forward
Call Backward                    but it is backward goto
                                 backward'
 Elseif pot2 = 1 Then           'If the joystick left go to -
Call Left                        left'
 Elseif pot2 > 400 Then         'If the joystick is not left but
Call Right                       it is right go to right'
 End if                         'End all if statements'
Loop                            'Go to do at the top'
End Sub                         'Marks the end of this sub'

Sub Tripod1()

 Do
  Call Pulseout(26,L1,1)        'Send value of L1 as a pulse
                                 out pin 26'
  Call Pulseout(22,L1,1)        'Send value of L1 as a pulse
                                 out pin 22'
  Call Pulseout(36,L1i,1)       'Send value of L1 as a pulse
                                 out pin 36'
  Call Pulseout(35,S1i,1)       'Send value of L1 as a pulse
                                 out pin 35'
  Call Pulseout(25,S1,1)        'Send value of L1 as a pulse
                                 out pin 25'
  Call Pulseout(21,S1,1)        'Send value of L1 as a pulse
                                 out pin 21'
 Loop
End Sub


Sub Tripod2()

 Do
  Call Pulseout(38,L2i,1)       'Send value of L1 as a pulse
                                 out pin 38'
  Call Pulseout(34,L2i,1)       'Send value of L1 as a pulse
                                 out pin 34'
  Call Pulseout(24,L2,1)        'Send value of L1 as a pulse
                                 out pin 24'
  Call Pulseout(23,S2,1)        'Send value of L1 as a pulse
                                 out pin 23'
  Call Pulseout(37,S2i,1)       'Send value of L1 as a pulse
                                 out pin 37'
  Call Pulseout(33,S2i,1)       'Send value of S2 as a pulse
                                 out pin 33'
  Call Pulseout(27,Son,1)       'Send value of Son as a pulse
                                 out pin 27'
 Loop
End Sub
```

```
Sub Task3()                  'Autonomous mode TASK'

Call Sleep(256)              'Wait for 256/512th seconds'
Do                           'Start of the Do loop'
Son = 1340                   'Move Sonar servo to center'
Call Sleep(128)
Call Putpin(28,1)            'Set INT on sonar board high'
Dis = Rctime(32,0            'Wait for ECHO to go low &
                              store in Dis'
Call Putpin(28,0)            'Disable Sonar'
Dis = Dis * 2 \ 300 + 6      'Turn Dis value into inches'
If Dis < 23 Then             'If Dis more than 23" then'
Call Sonlook                 'Go to sub Sonlook'
Else                         'Otherwise'
Call Forward                 'Call the Forward Sub'
End if                       'End if statement usage'
Loop                         'Go to the top'
End Sub


Sub Left()                   'Left turn script'

Call Sleep(64)               'Legs1 up'
L2 = 800
L2i = 1700
Call Sleep(64)               'Shoulder1 forward
S2 = 1100
S2i = 1100
Call Sleep(64)               'Legs1 down'
L2 = 500
L2i = 2000
Call Sleep(64)               'Legs2 up'
L1 = 800
L1i = 1700
Call Sleep(64)               'Shoulder1 back'
S2 = 1500
S2i = 1500
Call Sleep(64)               'Shoulder2 forward'
S1 = 1100
S1i = 1050
Call Sleep(64)               'Legs2 down'
L1 = 500
L1i = 2000
Call Sleep(64)               'Legs1 up'
L2 = 800
L2i = 1700
Call Sleep(64)               'Shoulder2 back'
S1 = 1500
S1i = 1450
End Sub


Sub Right()                  'Right turn script

Call Sleep(64)               'Legs1 up'
L2 = 800
L2i = 1700
```

```
Call Sleep(64)               'Shoulder1 forward'
S2 = 1500
S2i = 1500
Call Sleep(64)               'Legs1 down'
L2 = 500
L2i = 2000
Call Sleep(64)               'Legs2 up'
L1 = 800
L1i = 1700
Call Sleep(64)               'Shoulder1 back'
S2 = 1100
S2i = 1100
Call Sleep(64)               'Shoulder2 forward'
S1 = 1500
S1i = 1450
Call Sleep(64)               'Legs1 down'
L1 = 500
L1i = 2000
Call Sleep(64)               'Legs2 up'
L2 = 800
L2i = 1700
Call Sleep(64)               'Shoulder2 back'
S1 = 1000
S1i = 950
End Sub


Sub Forward()                'Walk forward script'

Call Sleep(64)               'Legs1 Up'
L1 = 800
L1i = 1700
Call Sleep(64)               'Shoulder1 Forward'
S1 = 1500
S1i = 950 '-50
Call Sleep(64)               'Legs1 Down'
L1 = 500
L1i = 2000
Call Sleep(64)               'Legs2 Up'
L2 = 800
L2i = 1700
Call Sleep(64)               'Shoulder1 Back'
S1 = 1000
S1i = 1450 '-50
Call Sleep(64)               'Shoulder2 Forward'
S2 = 1500
S2i = 1000
Call Sleep(64)               'Legs2 down'
L2 = 500
L2i = 2000
Call Sleep(64)               'Legs1 up'
L1 = 800
L1i = 1700
Call Sleep(64)               'Shoulder2 back'
S2 = 1000
S2i = 1500
End Sub
```

```
Sub Backward()              'Walk backward script'

Call sleep(64)              'Legs1 up'
 L1 = 900
 L1i = 1600
Call Sleep(64)              'Shoulder1 forward'
 S1 = 1100
 S1i = 1350
Call Sleep(64)              'Legs1 down'
 L1 = 500
 L1i = 2000
Call Sleep(64)              'Legs2 up'
 L2 = 900
 L2i = 1600
Call Sleep(64)              'Shoulder1 back'
 S1 = 1400
 S1i = 1050
Call Sleep(64)              'Shoulder2 forward'
 S2 = 1100
 S2i = 1400
Call Sleep(64)              'Legs2 down'
 L2 = 500
 L2i = 2000
Call Sleep(64)              'Legs1 up'
 L1 = 900
 L1i = 1600
Call Sleep(64)              'Shoulder2 back'
 S2 = 1400
 S2i = 1100
End Sub


Sub Center()                'Move all servos to home pos'

Call Sleep(64)
 Son = 1340
 L1 = 500
 L1i = 2000
 L2 = 500
 L2i = 2000
 S1 = 1250
 S1i = 1200
 S2 = 1250
 S2i = 1250
End Sub

Sub Low()                   'Squat all leg servos go down'
 L1 = 1250
 L1i = 1250
 L2 = 1250
 L2i = 1250
End Sub


Sub Sonlook()               'Look around for new path
                             sub'
Dim X as byte               'Set X as a byte variable'
X = 0                       'Make X = 0'
```

```
Do                          'Top of the DO loop'
X = X + 1                   'Loop counter add 1 to valu
Son = 1880                  'Move sonar servo to the le
Call Sleep(256)
Call putpin(28,1)           'Set INT on sonar board hig
Dis = Rctime(32,0)          'Wait for ECHO to go low
                             store in Dis'
Call Putpin(28,0)           'Disable Sonar'
 Dis = Dis * 2 \ 300 + 6    'Turn Dis value into inches
  If Dis > 23 Then          'If Dis more than 23" then'
   Call Left                'Call turn left 2 times'
   Call Left
  Exit Sub                  'After turning left exit sub'
  Else                      'If Dis less than 23" do this'
 End if                     'End if statement usage'
Son = 880                   'Move sonar servo to the righ
Call Sleep(256)
Call Putpin(28,1)           'Set INT on sonar board hig
Dis = Rctime(32,0)          'Wait for ECHO to go low
                             store in DIS'
Call Putpin(28,0)           'Disable Sonar'
 Dis = Dis * 2 \ 300 + 6    'Turn Dis value into inches
  If Dis > 23 Then          'If Dis more than 23" then'
   Call Right               'Call turn right 2 times'
   Call Right
  Exit Sub                  'After turning right exit sul
  Else                      'If Dis less than 23" do this'
 End If                     'End if statement usage'
 If X = 3 then              'If timeout counter X = 3 the
  Call Backward             'Call Backward 2 times'
  Call Backward
  Exit Sub                  'After Backing up exit sub'
  Else                      'If X = less than 3 then'
 End if                     'End if statement usage'
 Loop                       'Go to Do on top'
End Sub
```

From Conception to Creation

How to Construct a Robot Warrior Part II

by Ronni Katz

After covering the basic steps in preparing to build your robot in Part I, this article will describe how my entry, *Chew Toy* was constructed. The robot is not finished, but what needs to be done to complete him will be discussed.

Building 'Bot!

The metal ammo box originally purchased because it was considered to be an inexpensive, yet effective, way to house the electronics, wound up becoming the structural backbone of the robot. All the weapons systems and other features on *Chew Toy* are attached to the ammo box. The metal of the ammo box is not as tough as desired but it should provide adequate protection from impacts. All the electronics of the robot will go inside as well as the stationary axle that is a part of the robot's drive train. The axle - a long steel rod that goes lengthwise through the center of the ammo box - will be doing double duty as part of the drive mechanism and as a means of holding the batteries securely in place (see Photo 1). In the photo, you will see that the batteries have additional padding to cushion them from impact shocks.

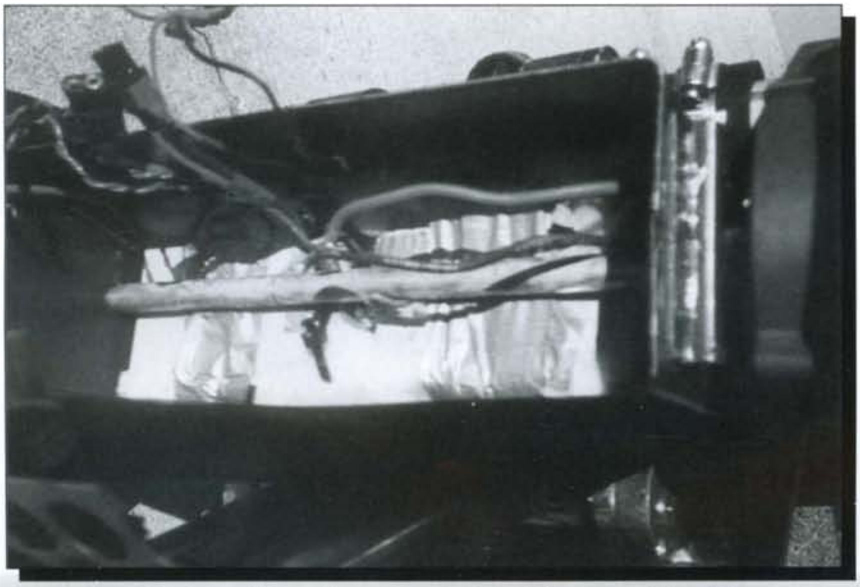


Photo 1. Ammo box with stationary axle and batteries with padding.

The robot's motive power is supplied by a pair of kiddy car motors (power wheel motors) that were inexpensive and found in the same surplus catalog as the ammo box. Because of the low price, extra motors were purchased and used for experiments. Testing of these motors was conducted to achieve maximum performance. It was found that when these 12-volt units are run at 24 volts, a good amount of power was produced. Subjecting motors to higher than rated voltage occurs frequently at robotic competitions. It's risky and requires a lot of trial and error testing to determine how much extra voltage the motors can handle. *Chew Toy's* motors were broken in before being tested to their voltage limits. It is also important to cool the motors properly. Breaking in the motors and cooling them well will prevent their melting. This was learned the hard way during the test phase! Knowing a few motors would fail during testing, extras were purchased to insure an adequate supply.

We chose motors that were easy to modify and that were designed to use a stationary axle. Working from the outside in, the motor casing is solidly attached to the chassis. The armature of the motor is mounted on a hollow shaft, or torque tube, that turns on the motor stationary axle. Attached to this torque tube is a plate that transmits the motor's power to the gearbox input. The motors use a three gear reduction system which gives a motor-to-wheel ratio of 110 to 1, greatly increasing the torque delivered to the drive wheels – no chains or belts here! The wheels are also designed to fit on a stationary axle and have bearings so all that was needed was to drill holes through the

Photo 2. Wheel shown bolted to drive plate.

wheels and the drive plate of the gear box and bolt them together. If you look at how a wheel is arranged on the axle (see Photo 2) you can see there is a washer over the axle with a cotter pin securing the wheel in place. Where the wheel is bolted to the drive plate of the gearbox is also visible. The wheels are decent sized with deep treads for added traction.

The ammo box was destined to receive all the electronics. It took time to determine the arrangement of all the items inside the limited space. Inside the ammo case would be the *Vantec* speed controller, the radio and its battery pack, two Futaba servos driving standard micro switches to switch the weapons systems, and three relays for the weapons systems - two for the arm mechanism and one for the saw motors. An evening of careful planning and trial and error assembly found the configuration that worked best. They all fit, albeit in a densely packed configuration. Between the axle and the rear of the box are the batteries - two high rate discharge *Yuasa MPH 1-12* batteries. These batteries can supply 100 Amps or more. They were chosen for their high discharge rate, something most gel cell batteries are incapable of, as it was needed to run the saw motors. Quality varies widely among gel cell manufacturers. The *Yuasas* ran $26 each, not inexpensive, but battery quality is an area where you can't afford to skimp. Everything was fit in and tested - the robot was driven around as a mechanical ammo box to be certain the design worked. The axle through the center of the robot, the gearbox, and the wheels helps to brace the batteries in place. The motors are held in place by hose clamps over PVC pipe. It may not look very pretty but the parts are inexpensive, effective and easily obtainable – more parts that came from your local hardware store! Yes, most of this robot's parts were obtained from scrap yards, hardware stores, scavenged materials and a surplus catalogue (or two).

Although work on the basic drive box was completed and initial testing showed the design to be a solid one, there was still much more work to be done.

Armor & Weapons

Chew Toy's armor and weapons system were constructed from a combination of surplus catalogue goodies and scavenged parts. The prow of the robot was fabricated of steel obtained from a rack mounted computer system. A 1/4-inch aluminum plate, part of the support structure for the weapons system, came out of a dumpster. Cut into the desired shape with a jig saw, it was honed with a *Dremel* tool, and welded to the main support structure (the ammo box). The weapon support structure fits neatly between the two fan outlets. Attached to the front part of its underside is an inexpensive small furniture castor. When the prow (the arm) is down, that foremost wheel is not visible but in Photo 3 it can be clearly seen. It's bolted to the front of the machine and supports the two pillow boxes that hold the saw bearings. The bearings used for the weapons system were designed for misalignment – the bearings are sitting in a rubber gasket, which can move around slightly. Therefore, the builder doesn't have to be precise on alignment. Just stick the bearings in there, slide the axle through them and clamp it down to get a system that is reasonably strong and spins. The central theme of *Chew Toy* was building a robot cheaply and easily and the KISS (Keep it Simple Simon) weapons array helps to do just that!

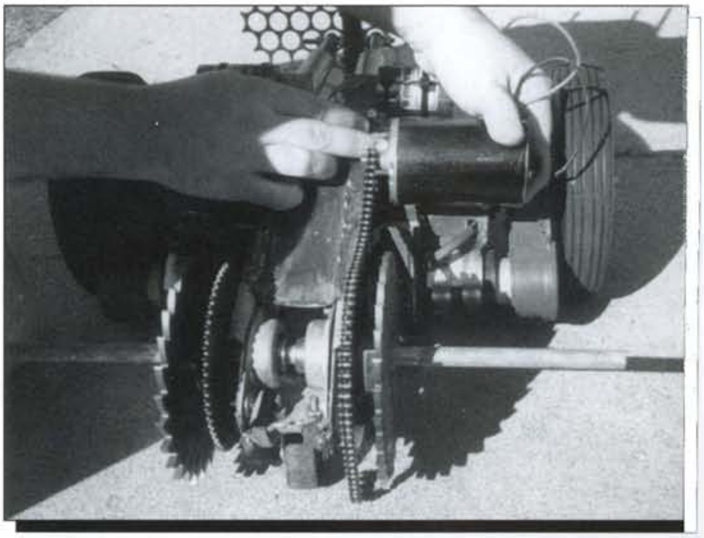


Photo 3. Front view with arm up. Note: motor and chain drive for saws, and caster under pillow boxes.

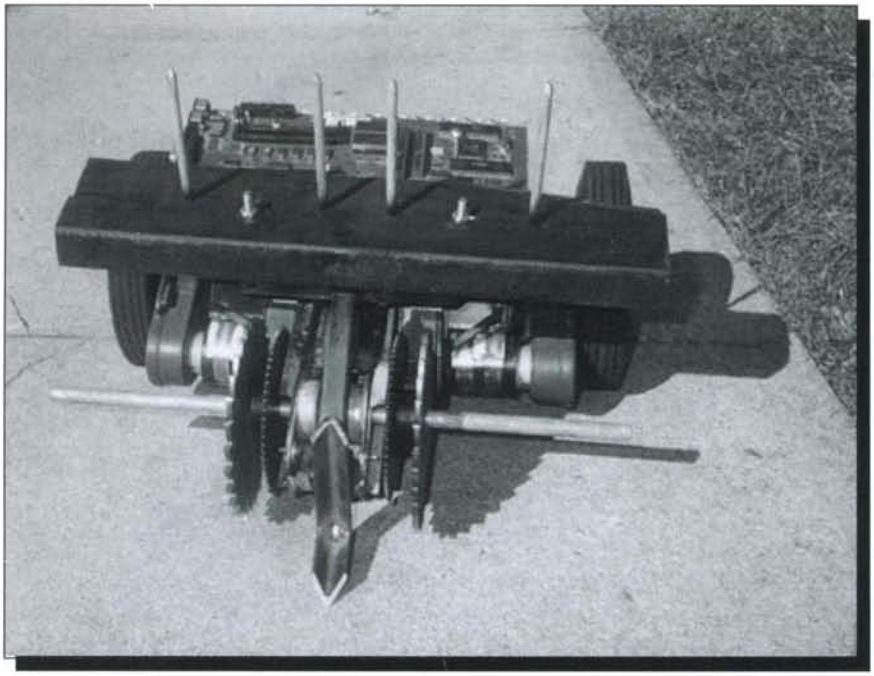


Photo 4. Front view with arm down showing 2 x 4 and nails.

The large rod you see mounted to the front of the robot in Photo 3 is the saw axle. The saws are milling tools that were picked up at Fazio's, a metal scrap yard. *Berg* sprockets and chains were used to construct the saw's drive. The shoulder on the sprocket will be cut down with a lathe and the sprocket bolted to the saw, making one unit. Although combining the saw mechanism in this way makes the unit heavier, it is desirable in this case because of the increased spinning momentum. The design will allow *Chew Toy's* saws to strike an opponent and keep on spinning and doing damage instead of stopping abruptly.

Chew Toy's weapon is a rotary spinning mass. The design is very simple: two milling saws on each side of the prow that are driven by a chain sprocket mechanism. As you can see in the photo, a large chain sprocket was used – it takes chain reduction out of the system and in doing so, transmits the maximum amount of torque. These saws were designed for low speed and high torque. The idea is to pull an opponent into the "mouth" area of the robot to "chew" on it and send many parts flying. The motors that power the saws will be mounted on a support structure welded to the front of the robot (this is not built yet). The saw motors will also be running on 24 volts instead of the recommended 12. These motors will only get intermittent use; thus, the reduction in life span from this hard usage should not pose a problem. If one motor should blow out, the second one will be able to power the saws. These motors were found through a surplus supply catalog. Although I had no specs on their design, or knew anything about who made them, they were inexpensive and testing proved they had the necessary torque and would work well for the intended purpose.

The arm was originally intended to right the robot if it were turned on its back, but then became a weapon in its own right. The arm is made out of angle iron from the local hardware store. Welded on to the ammo box and attached to the front is a little bent piece of steel with a hook. The initial welding on *Chew Toy* was farmed out. Drew Lindsey (Team *SPIKE*), who now has both welding equipment and skill at using it, did later welds! The original arm conception has evolved considerably, and the appearance has changed as we continued our improvisation. Things were added as the inspiration hit. The old motherboard and perforated metal screening were attached as armor. The 2 x 4 with nails was incorporated to make sure the robot could right itself should it be flipped. The nails, and the reach they added, were necessary to accomplish the flipping. When the arm is lowered (see Photo 4) the nailed 2 x 4 gives the robot additional protection. More armor in the form of circuit boards, perforated metal and another 2 x 4 to protect the robot's rear will be added when construction is nearing completion.

The arm actuators seen in Photo 5 were donated by *Motion Systems*. These actuators have three inches of throw, which gives us about 70 degrees of travel, enough to flip the robot upright. If the robot is flipped, it will be resting on the nails and the process of raising the arm rolls the robot back onto its wheels.

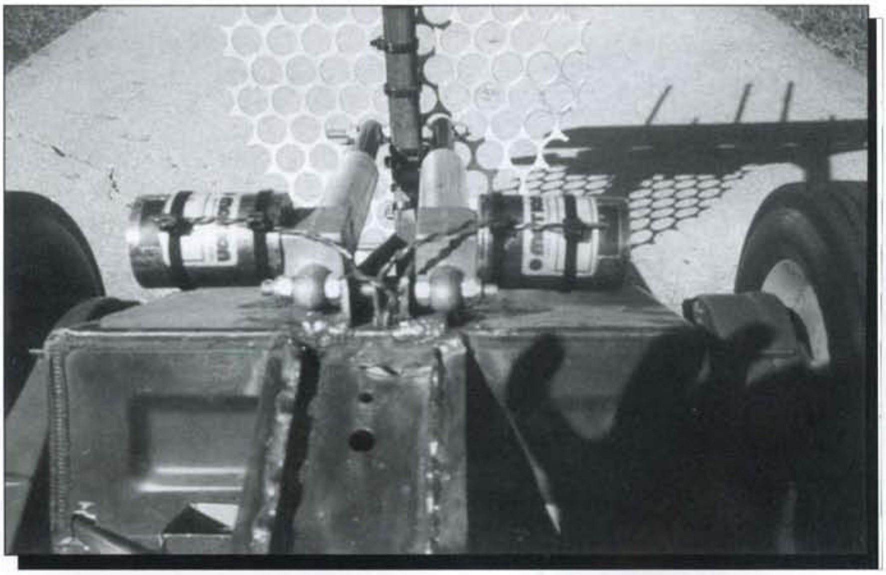


Photo 5. Top view showing actuation arms.

When the arm is lowered, the hook part fits neatly between the saw blades, allowing the saws to do their work. Raising the arm provides 70 pounds of lifting force, which should be enough to pick up an opponent and allow the saws to cut away at its underside. The lifting arm can also be used as an 'upper jaw.' The pressing force of the motors of this 'upper jaw' can trap an opponent between it and the 'lower jaw' prow. Saw-like teeth welded to the underside of the arm and the top of the prow will make a "mouth." *Chew Toy* will be able to live up to his name! He will look like a big nasty shark with a mouth full of razor sharp rusting teeth. Opponents will need tetanus shots if he bites them!

We were influenced by other robot designs we saw online when we designed our armor. The focus was on our weight class, our potential opponents. One robot, *The Missing Link*, had a huge nasty circular cutoff wheel on its front. These wheels are designed to cut though steel and it could cut through *Chew Toy's* frame without slowing. However, cutoff wheels bog down and get jammed when cutting through wood. Thick pine 2 x 4s became a part of *Chew Toy's* armor. This would slow *The Missing Link* and any other robot using weapons designed to cut steel. Many builders don't perceive wood to be good armor. Actually, a thick piece of pine is hard to cut through, especially if it is attached to a robot that is fighting back. Robots mounting large toothed wood cutting blades have a good chance against *Chew Toy's* pine armor. Hopefully, he won't be standing still long enough to give them the chance! The nails attached to the pine 2 x 4 give additional protection. Saws trying to cut through the wood may hit the nails causing them to jam, break or lose teeth. The combination of nails in wood should make cheap yet effective armor - even if it looks ugly!


Engineering & Entertainment

* * *

Together they please you and the crowd.

Final Words

Despite its appearance, *Chew Toy* does have good engineering. Making a robot from available, inexpensive parts does not mean the design is poor. In designing *Chew Toy*, attention was paid to the overall layout, to where the center of gravity would be, and to giving the robot the ability to right itself from any orientation. The latter feature was a major design challenge. Paying heed to how the parts fit together, the location of the center of gravity, and the envelope of the robot in order for it to roll properly and right itself was an intricate problem. We took care not to repeat the mistakes of others. No blob with wheels that had everything encased in a box for us! We wanted the components to fit together intelligently for maximum utilization. The design allows its separate parts to perform a secondary function, such as the axle being an internal support for the batteries and the mo tors adding additional support to the robot's overall structure This result came from playing around with all the parts, try ing different configurations, and finding the best way to fit i all together.

Conceptually, we focused on three things: good overall de sign for maximum offensive and defensive capabilities, eas of driving for effective movement in the arena, and the crow pleasing effect of *Chew Toy*. The overall design is solid. I overcomes the majority of ways robots lose in combat. Mos robots don't lose because their armor is bad. Rather, it is because they are flipped over, something internal or externa breaks on impact, or it is hung up on something due to insufficient ground clearance. *Chew Toy's* electronics are wel cushioned against impact damage within the ammo box tha has additional welded steel. *Chew Toy's* arm can be used t right it should it be flipped and its weapons should prov effective in combat. Although an opponent could strike th exposed wheels, they are large and provide in excess of a inch of ground clearance, which is enough to drive over gras with no difficulty. *Chew Toy* will be hard to stop - it will stil be fully mobile with a chance to break free even if it run over a wedge or a lifter gets underneath. Its weapons are designed to rip chunks off other robots and drive over the debris without slowing. In the initial drive tests, grass and lawn hazards posed no problems.

Two items are very important in robotic combat, driving ability and pleasing the crowd. Battles have been lost due to poor control of a robot's movement in the arena. Test-drive the robot as much as possible before competing! Discover early how to compensate for odd quirks. Pleasing the crowd is also important; if the robots are tied, the vote of the crowd decides who wins. A robot with a good design, cool weapons that are entertaining to see in action, and able to show its capabilities best, is the ultimate objective. Saws are a favorite weapon - people like to see bits and pieces flying. Nothing gets more cheers than the flash of sparks that fly when one robot saws into the hide of another! People enjoy robotic mayhem and destruction and, although some of the weapons that get the most cheers don't really do much real damage, they impress the crowd which is part of what robotic combat is all about! *Chew Toy's* one special crowd pleaser is that it can do wheelies when put in reverse and then forward really fast!

Keeping in mind the importance of good overall design, ease of driving and pleasing the public (yourself included!) is the best thing you can do as you construct a combat robot warrior and enter it into competition!

Parts Suppliers–Part II

Among the most difficult things for the builder of a combat robot is locating sources of components. I hope this list of vendors will help. Some I have used and others have been used by fellow enthusiasts with good results. Contact them and ask for a catalog (have your business address handy in case they need it).

GENERAL EQUIPMENT/ SUPPLIES

McMASTER-CARR
mcmaster.com
(five locations – those listed plus Atlanta and Cleveland)
P O Box 54960
Los Angeles, CA 90054-0960
562.692.5911
la.sales@mcmaster.com

P O Box 440
New Brunswick, NJ 08903-0440
732.329.3200
nj.sales@mcmaster.com

P O Box 4355
Chicago, IL 60680-4355
630.833.0300
chi.sales@mcmaster.com

The first thing you should do is obtain a catalog from McMASTER-CARR. They have everything. You can build excellent robots using just parts from this catalog. Ninety-nine percent of their products are in stock, and if you order today, you can expect the part at your doorstep tomorrow.

* * * * * * *

MSC Industrial Supply Company
75 Maxess Road
Melville, NY 11747
800.645.7270
mscdirect.com

They carry many of the usual items plus a large selection of cutting tools (potential weapons), and machine shop equipment. They have their "big book" and they are not kidding! At 4075 pages, it is thicker than The Bible and PACKED with more parts than you can imagine. A great supplier, they are HIGHLY recommended by fellow robotic combatants. Team Delta used them to obtain steel and other parts for their excellent robots.

* * * * * * *

W.W. Grainger, Inc.
455 Knightsbridge Parkway
Lincolnshire, IL 60069-3639
800.877.0150
grainger.com

The Grainger catalog has 4106 pages but is in a smaller format than MSC. They have many of the same items as McMASTER-CARR and MSC, but they also carry many items not found in the other catalogs.

* * * * * * *

Small Parts, Inc.
P O Box 4650
Miami Lakes, FL 33014-0650
800.220.4242
smallparts.com

Small Parts, Inc. has a wide range of hard to find parts. Ask for catalog #17. It has a photo of our *Stealth* robot, which won the 1995 "FIRST" competition at Disney World in Florida.

* * * * * * *

Industrial Metal Supply Company
3303 N. Fernando Blvd
Burbank, CA 91504
818.848.4439
imsmetals.com

If you live in southern California these folks have three excellent shops you can visit. They also take phone orders and do business over the Internet. Very reliable – another company whose products have seen action at robotic combat events.

* * * * * * *

Robin Materials
1951-T Colony St.
Mountain View, CA 94043
650.966.1533
rmat.com

Robin Materials has specialty metals – such as titanium, tungsten, etc. They do custom work and their designs have seen "battle."

POWER TRANSMISSION

Stock Drive Products
2101 Jericho TPKE
P O Box 5416
New Hyde Park, NY 11040-5416
516.328.3300
sdp-si.com

Stock Drive probably has the largest selection of power transmission equipment including belts, chains, sprockets, pulleys, bearings, and gears of all the suppliers. Ask for their complete set of catalogs.

* * * * * * *

W. M. Berg, Inc.
499 Ocean Avenue
East Rockaway, NY 11518
800.232.2374
wmberg.com

Berg is an excellent source.

* * * * * * *

Boston Gear
14 Hayward Street
Quincy, MA 02171
888.999.9860
bostgear.com

Boston Gear carries some parts capable of heavier-duty than Stock Drive and Berg.

* * * * * * *

PIC Design
86 Benson Road
P O Box 1004
Middlebury, CT 06762
800.243.6125
pic-design.com

* * * * * * *

NORDEX, Inc.
50 Newtown Road
Danbury, CT 06810-6216
800.243.0986
nordex-inc.com

METALS

Metalmart, Inc.
12225 Coast Drive
Whittier, CA 90601
800.888.7766
metal-mart.com

* * * * * * *

Carpenter/Dynamet, Inc.
195 Museum Road
Washington, PA 15301
724.228.1000
cartech.com

PLASTICS

Cadillac Plastic
2855 Coolidge Highway Suite 300
Troy, MI 48084
800.274.1000
cadillacplastic.com

* * * * * * *

Commercial Plastics
commercialplastics.com
(two locations)
98-31 Jamaica Avenue
Richmond Hill, NY 11418
718.441.1500

3241 Keller Street
Santa Clara, CA 95054
800.325.1262

* * * * * * *

Consolidated Plastic Co., Inc.
8181 Darrow Road
Twinsburg, OH 44087
800.362.1000
no web site

* * * * * * *

Ryerson Plastics
2558 West 16th Street
Chicago, IL 60608
800.242.2114
no web site

RADIO CONTROL EQUIPMENT

This is where you will find items like radio transmitters, small motors, electronic speed controls, and tons of parts that could be used in feather or lightweight robots.

Tower Hobbies
P O Box 9078
Champaign, IL 61826-9078
800.637.6050
towerhobbies.com

* * * * * * *

Hobby Lobby
5614 Franklin Park Circle
Brentwood, TN 37027
615.373.1444
hobby-lobby.com

* * * * * * *

Ace Hobby Distributors, Inc.
116 W. 19th Street
Higginsville, MO 64037
800.322.7121
acehobby.com

* * * * * * *

Aveox, Inc.
31324 Via Colinas Suite 103
Westlake Village, CA 91362
818.597.8915
aveox.com

* * * * * * *

SIG Manufacturing Co., Inc.
P O Box 520
Montezuma, IA 50171-0520
800.247.5008
sigmfg.com

MISCELLANEOUS MECHANICALS

Cutter's Choice, Inc.
cutterschoice.com
(two locations)
1637 S. Main Street
P O Box 939
Willits, CA 95490
707.459.3509

- - - - - - - - - - - - - - - -

2008 East 33rd Street
P O Box 10308
Erie, PA 16514
814.898.1629

* * * * * * *

DU-BRO Products, Inc.
480 Bonner Road
Wauconda, IL 60084
800.848.9411
dubro.com

* * * * * * *

Kart World
1488 Mentor Avenue
Painesville, OH 44077
440.357.5569
kartworld.com

* * * * * * *

Northern
2800 Southcross Drive West
Burnsville, MN 55337
800.533.5545
northern-online.com

SURPLUS

C and H Sales Co.
2176 E. Colorado Blvd
Pasadena, CA 91107
800.325.9465
candhsales.com

ON-LINE DIRECTORIES OF SUPPLIERS

Arrick Robotics
robotics.com/mechpart.html

* * * * * * *

Team Delta
teamdelta.com/robopart.html

* * * * * * *

Thomas Register
http//thomasregister.com/index.html

* * * * * * *

Yuasa-Exide, Inc.
yuasa-exide.com/index.html

*Author **Ronni Katz** built ChewToy after 3 years of Robot Wars™ experience with Spike, and two years of interviewing fellow robot warriors. Her day job is programming, and her past life was landing on aircraft carriers. Her action-packed novel, Wing Commander, is available at major bookstores (ISBN 0-9662083-0-7) under the pen name Ron Karen. We at RS&T are proud to have her on our pedagogical team.*

SUBSCRIPTIONS & RENEWALS

PAID CHARTER SUBSCRIBERS

may extend their subscriptions by:

- Converting from 6 to 12 issues by sending a check for $19. This option will extend your subscription to a total of 12 issues.
- Renewing for a special charter renewal rate of $34. This option will add 12 issues to your current 6-issue plan for a total of 18 issues.
- Canadian Charter Subscribers. Renew, prepaid, for 12 more issues for only $49 US.
- Don't miss a single issue! Send a check or money order to: Accounting, *Robot Science & Technology*, 2351 Sunset Boulevard #170-253, Rocklin CA 95765.

NEW SUBSCRIBERS will receive 12 issues for $39 US. Prepaid subscribers will be eligible for future special offers. Subscribers outside the US must pre-pay - see international notes below.

LIBRARIES and SCHOOLS: *Robot Science & Technology* is an exciting and informative magazine about an educationally vital topic. It will prove to be one of the most popular periodicals on your bookshelves. Act now to make this practical magazine available to your students, researchers, engineers and educators. They will thank you for it.

For $99, in the US only, schools and libraries will receive 12 issues of the magazine. Prorate for 36, 10 or 9 month periods. Specify the date you wish to begin the subscription. Send a prepaid purchase order or request a pro forma invoice. We also accept institutional orders through subscription services.

Library Service Office
Robot Science & Technology
2351 Sunset Boulevard #170-253
Rocklin, CA 95765

International Notes:

- International subscriptions - $99 prepaid for 12 issues, individual or institution.
- Most of the cost of international subscriptions is for shipping by air.

Visa and Mastercard:

- Accepted world-wide through our secure server at www.RobotMag.com or call toll-free 888.510.7728 (US and Canada only) for credit card orders, 9 a.m. to 5 p.m. California time, Monday through Friday.

Subscriptions are risk-free! Full 90-day refund is guaranteed in the US and Canada.

Robot Science & Technology
2351 Sunset Boulevard #170-253
Rocklin, CA 95765
916.632.1000
888.510.7728 (US and Canada)
service@RobotMag.com
RobotMag.com

BASIC Controllable Robots

Build your own Robot We Make it Easy!

Lynxmotion

http://www.lynxmotion.com

MRA-KT *Mobile Arm Kit* **$250.00**
- Pre-Assembled Electronics.
- BASIC Controllable.
- Easy RS-232 Control from PC or Micro.
- Excellent Advanced Platform.
- Complete Kit, Works with Tracker / IRPD.

5AA-KT *5 Axis Arm Kit* **$195.00**
- Pre-Assembled Electronics.
- BASIC Controllable.
- Easy RS-232 Control from PC or Micro.
- Excellent Intermediate Project.
- Complete Kit.

3AA-KT *3 Axis Arm Kit* **$155.00**
- Pre-Assembled Servo Controller.
- BASIC Controllable.
- RS-232 from PC or Microcontroller
- Great Kit for the Beginner.
- Complete Kit.

H2-KT *Hexapod II Kit* **$375.00**
- Full Size Hexapod Robot.
- 2 Degrees of Freedom per Leg.
- Low Current Draw Leg Design.
- Excellent Advanced Project.
- Mechanics / Servos only, Add 2 SSC-01.

H1-KT *Hexapod Kit* **$150.00**
- Walks Forward, Reverse, Left, Right.
- BASIC Programmable.
- Requires Soldering.
- Great Intermediate Project.
- Complete Kit, Works well with IRPD.

CR-KT *Carpet Rover Kit* **$115.00**
- Designed for Carpet or Rough Terrain.
- BASIC Programmable.
- Requires Soldering.
- Great Intermediate Project.
- Complete Kit, Works with Tracker / IRPD.

MM-KT *Micromouse Kit* **$95.00**
- Designed for Table or Smooth Surfaces.
- BASIC Programmable.
- Requires Soldering.
- Great Intermediate Project.
- Complete Kit, Works with Tracker / IRPD.

BG-KT *Big Grip Kit* **$75.00**
- Designed to be added to a Mobile Base.
- Gripper Opens to 3 inches.
- Lifts up to 1 pound!
- Great Intermediate Project.
- Mechanics / Servos only, Add a FS-01.

Accessories

FS-01	First Step (Stamp 1) Microcontroller Kit	**$40.00**
SSC-01	SSC Serial Servo Controller Board	**$44.00**
IRPD-01	Infrared Proximity Detector Kit	**$30.00**
TRA-01	Tracker Line Following Sensor Kit	**$20.00**
DHB-01	Dual H-Bridge DC Motor Driver Kit	**$30.00**
SM-01	ServoMotion for Windows (General Purpose)	**$30.00**
RM-01	RoboMotion for Windows (For 5AA-KT)	**$30.00**
S6P-01	Hitec 42oz/in Servo 6 Pack	**$80.00**
SLS-01	Hitec 133oz/in Large Scale Servo	**$40.00**
BMR-01	Mobile Robots Book	**$48.00**
TRP-01	The Robotic Practitioner Back Issue Set (6)	**$40.00**

Ask for our free catalog!

Lynxmotion, Inc.
104 Partridge Road
Pekin, IL 61554-1403 USA
http://www.lynxmotion.com

Tel: 309-382-1816
Fax: 309-382-1254
Sales@lynxmotion.com
Tech@lynxmotion.com

ADVERTISER Index

Acroname .... page 13
303.258.3161 • fax 303.247.1892
www.acroname.com

ActivMedia .... pages 57,59,61
800.639.9481
www.activmedia.com/robots

Advanced Design, Inc. .... inside front cover
520.544.2390 • fax 520.575.0703
www.robix.com

Cybug .... page 29
JCM Electronic Services
www.members.home.net/cybug

Effective Engineering .... page 50
619.450.1024 • fax 619.452.3241
www.trix.com/effective/

General Robotics .... page 7
800.422.4265
www.edurobot.com

HVW Technologies .... page 64
403.730.8603 • fax 403.730.8903
www.hvwtech.com

Images Company .... page 62
718.698.8305
www.imagesco.com

Lemos International .... page 4
508.798.5004 • fax 508.798.4782
www.lemosint.com

Lynxmotion .... page 49
309.382.1816 • fax 309.382.1254
www.lynxmotion.com

Mister Computer .... page 55
619.464.2412
www.mister–computer.com

Mondo-tronics .... page 34
800.374.5764 • fax 415.491.4696
www.robotstore.com

NetMedia/BasicX .... page 1
520.544.4567 • fax 520.544.0800
www.BasicX.com

Parallax .... back cover
888.512.1024 • fax 916.624.8003
www.parallaxinc.com

Pontech .... inside back cover
714.985.9286 • fax 714.985.9288
www.pontech.com

Positive Logic Engineering .... page 17
www.positivelogic.net

RobotBooks.com .... page 28
www.robotbooks.com

Solarbotics .... page 19
403.818.3374 • fax 403.226.3741
www.solarbotics.com

Thrifty Distributors .... page 53
800.342.3610 • fax 610.647.6289
thriftydist@worldnet.att.net

Zagros Robotics .... page 23
314.768.1328
www.zagrosrobotics.com

DO YOU HAVE AN IDEA OR PRODUCT FOR THE NEWLY EXPLODING MARKET OF ROBOTICS?

We can help you sell your products to a specific market and make your advertising dollars work more effectively.

Are you diluting your marketing resources in electronics hobby, or ham radio magazines?
Or is your message getting lost in the classified section of big-money magazines?

RS&T takes your message directly to those who need and want your products.

Cost Effective

We are not like niche magazines; your ad in RS&T is placed to **gain maximum exposure.** We're small enough to give **personalized service** to every advertiser. We offer a **significantly better** cost per thousand readers (CPM) than any other quality tech magazine. So advertising in RS&T is easy and **economical for both small and large companies.**

Design Flexibility

Robot Science & Technology offers your team the flexibility to present your products to our **rapidly expanding market** in a number of ways. Use multiple inserts for **maximum effect.** Use a second color for **maximum impact** that's economical. Use a Reader Reply Card for **immediate response.** For companies that want more presence, we design premium pages within the magazine at no extra charge. Website space is also free to paid advertisers.

Expanded Distribution

You get **global reach** because our distribution is international, through carefully selected major bookstores, and our personal presence at major robot events. So your message reaches both the existing robotics community and the **swiftly developing market** in home, sport and classroom robotics.

Quality, Reliability, Professionalism

Highly experienced professionals have recently joined our team, with proven track records at high-quality international technical publications. This will allow us to eventually become a **monthly magazine.** RS&T has shorter lead-times than tradition-bound magazines, speeding your timely ad to qualified buyers.

Robot Science & Technology will work for you today!
So call 916.632.1000, get to know us, and ask for a media kit.

ROBOT
science & technology

Feedback Loop

More Hands-On Content, Please

First of all, I'd like to say I'm very excited about the subject matter of your magazine. However, I wish some of the articles were targeting beginning robot enthusiasts without degrees from MIT. If I already knew everything required to understand most of your articles, I wouldn't need to subscribe to Robot Science & Technology. I've had college courses in basic electronics and robotics fundamentals and still have no idea how to start building my first hobby robot. I subscribed to your magazine in hopes that I might find some useful information, but instead I'm getting discouraged. I think step-by-step projects with explanations of what each component actually does, and why, would make robotics less intimidating and increase your magazine's circulation.

- Russell Ingersoll

Dear Russell,

We hear you, understand your concern, and hope this issue has more of the type of content you are looking for. Just by way of background, our editorial philosophy calls for a balance, with technical features at one end of the spectrum and hands-on how-to articles at the other. This seems to satisfy most of our readership, and this issue of the magazine reflects that philosophy. Herein we cover the robotics gamut with articles on soft computing, algorithms, robot warriors, and how-to features. Two features fall into the latter category, wherein we provide step-by-step information on building, programming and customizing robots. Also, in a series commencing with this issue, we address the fabrication of skins to protect and enhance the performance and appearance of robots. We thank you for your input, and ask that more readers let us know what they like to see in the magazine.

Success is Just Another Phone Call Away

I recently subscribed to your magazine and have been following the "Basics of a Digital Brain" series. I called Motorola and tried to order the manual that is listed in the article, but they told me that the number I have does not correspond to any Motorola manual. Any help would be appreciated. Thanks.

-R.E.

Dear R.E.,

You must be talking about the M68HC11RM/AD manual referred to on page 44 of the July 1998 issue. We did a little checking around, and found that version 3.0 of the manual is available at the telephone number provided on the same page of the magazine: 1-800-441-2447.

We called the number to check on your problem, and the agent we spoke to at the Motorola Literature Distribution Center was very courteous and accommodating.

By the way, if you order less than five of these manuals, they are free. We suggest you try again.

A Delayed Action Cover Story

I am a dedicated reader and robot hobbyist. In issue four, something looked a little strange. I expected an in-depth article about the GrowBot since it was featured on the cover. I was a little disappointed that there wasn't one. Just something to consider for future articles.

- Michael Bloom

Dear Michael,

We intend to write about GrowBot soon. The compact, reliable robot is popular and offers considerable potential for expansion *(hence its name), so it is a great candidate for an article. The feature will conform to our format for the construction series, that is, step-by-step instructions accompanied by photographs and tips. The thing about GrowBot is that construction is simple, and it can be programmed to do lots of things. We will therefore highlight the versatility aspect of the robot and its modularity. The Growbot kit includes the BASIC Stamp Windows editor, and documentation includes source code ideas and tips. The project is rich with possibilities, so we are looking forward to it as much as you are. Don't touch that dial, Michael, and thanks for your comments.*

Everything You Ever Wanted to Know About The Splits, and Juggling Acts

Just a quick note to say how much I enjoyed reading the latest issue. Your content is markedly improving issue-to-issue. I especially enjoyed the FIRST article, which was well written, and had just the right balance of background and detail, although I am curious why you cannot print the complete article in one part of the magazine. I also enjoyed the latest Micromouse algorithms, and am sure that it will be even more appreciated by anyone not previously familiar with such schemes.

- Ian Cull

Dear Ian,

Thank you for your comments. We are expending a great deal of effort to make the magazine just as good as it can be, and it is nice to be appreciated.

As you have noticed, the integration of editorial content into the physical layout of the magazine usually calls for compromise. The FIRST article was actually subjected to two such compromises: (1) Most of the graphic images in the article were originally in color, but we printed some of the pages in a black and white section and (2), we split the story.

By way of background, over half of the magazine (36 of 68 pages) is printed in color: The first and last eight pages, and the middle sixteen, plus the covers and their flip sides. Naturally, there is competition for this scarce real estate provided by other articles with color graphics, and by advertisers with color ads.

In an ideal world, the FIRST article would flow from page to page without a split, and we would have printed all eleven pages together in a color section. But there was competition for the treasured color space, and the lead story, kicked off the editorial section and spilled over into black and white, but we recovered some color by splitting it into the back section. We hope the inconvenience of flipping around didn't spoil the story for you.

We are also gratified that you are enjoying the Micromouse Algorithm series. The A Algorithm article in this issue should prove of interest to you. Dr. Tak Auyeung has provided a truly readable and understandable explanation of this intriguing algorithm. It essentially assimilates previously gained knowledge to maneuver an autonomous robot through the maze on the shortest possible path. Wouldn't it be nice to apply this algorithm in our world when called upon to 'run errands?' Then again, we probably already do and just don't realize it! We will be hearing more from Dr. Auyeung in the future.*

Submission Guide

You Can Write For Robot Science & Technology

RS&T accepts quality feature articles from individuals, and our rules which govern acceptance of manuscripts are tight but fair. If you would like to write for us, just follow the guidelines provided here, and submit the manuscript to us. If your article meets our requirements, we will work with you, and the result could be an article in our magazine. You should:

Tailor your presentation. About half our subscribers are engineers and academicians, and most of the rest are students, so the presentation can range from intellectually stimulating for the most technically adept, to understandable for those who are not yet at the highest level. All who read our magazine are curious about all aspects of technology within the exciting science of robotics. They are interested in how things work, and ultimately might like to get involved at a hands-on level. Our desire is to provide something interesting and challenging for everyone.

Educate and inform. We want cutting-edge articles. We welcome articles on hardware or software, and those that concern microprocessors and programming robots with readily available computers, programs and materials. If your article is theoretical, you should lead every major idea in an intellectual way, but spend some time to explain what that idea means in an instructional manner, like a tutorial.

Be Original. It is important that you be the author of your article and the purveyor of your ideas. We will evaluate previously published material on disciplines related to robotics, like artificial intelligence, if the material has been extensively revised and made applicable to robotics through an original thought process. Our subscribers want to read about new technology, fresh ideas and new approaches to problem-solving.

Content

Short articles may contain 800 to about 3000 words, with the typical **feature** manuscript for RS&T containing 3000-3500 words, and several photographs, figures, graphs or tables. Be clear, concise, and accurate.

A general outline:

Introduction. This is your opportunity to acquaint the reader with your subject matter with general information. It is the roadmap for the discussion that follows.

Discussion. This is a detailed discussion of your subject matter, and it should unfold in a logical, ordered manner. Check all material presented as facts.

Conclusion. This does not necessarily have to be a review o the article, but it does need to summarize it in general terms It can also be used as a springboard to the next segment if the article is one of a series.

Details for text

The first page should include your name, address, telephone number(s), e-mail address, website URL and date.
We prefer that submissions be attached to e-mail as MS Word (PC or Mac) or WordPerfect files. If you do not have those programs, submit the article in ASCII, MS-DOS text, or rtf Do not embed graphics, but send them in separate e-mails to keep file size small.

If text is submitted in hardcopy, it should be on white, 8.5 x 11-inch paper, one side only, double-spaced in minimum 11-point type, and you should also provide it on a 3.5-inch floppy disk in the programs listed above.

Use short, crisp sentences and keep paragraphs to about three or four sentences. Use subheadings to break the text into easily readable and identifiable sections.

Verify the spelling of names, titles, and company names and that dates, phone numbers, and references are correct.
If you use acronyms, the first reference should be fully described, followed by the acronym.

Be sure to refer to all photographs, figures, tables and lists.

If you are writing a construction article, you should provide the reader with the following information:

- Parts list.
- Known sources for obtaining parts. Include the complete address, phone number, website URL and e-mail address.
- Any special equipment needed for construction or testing.

Photographs

Four to six photos are desired for each article submitted. Do not send negatives.

We prefer to scan photographs ourselves, and 4 x 6 or larger glossy color prints give us the most latitude. Electronic submission of photographs is not acceptable.

Keep the background neutral.

Focus, depth of field and detail are vital. Do not send photographs that are out of focus. Automatic focus cameras often produce fuzzy images.

Include the whole subject in the photograph.

If you are photographing a robot, we must see the whole figure, but we also want close-ups of the drive train, power plant, sensor array, or anything else that is captivating about your robot. If your article is a how-to piece, ensure that pictures of each phase of your project fit with the text.

Caption each photograph. Nobody knows better than you what you have photographed, so number each one on the back with a sticker to preclude scoring the photo. Reference them on a caption page enclosed with your story.

Other Graphics

All graphics must be 300 dpi, and a minimum of 2-inches square. We may redraw figures to maintain consistency among articles. Do not place graphics into a word processing document. Save them as separate files in the following formats (Mac or PC):

- **Schematics**—TIFF, EPS or BMP files.
- **Line art**—Adobe Illustrator, EPS, TIFF, or BMP.

Editorial Process

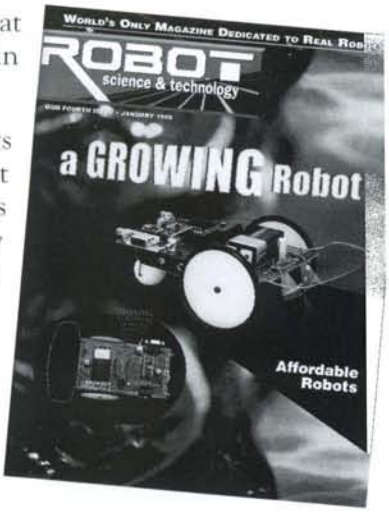
All material accepted for publication will be subject to editing by our staff. If your manuscript contains a large amount of misspellings, grammatical errors, or requires extensive editing for any other reason, it may be rejected. Do not rely solely on a spell-checker.

During the editing process, we may request clarification or additional material. Your responses should be as prompt as possible, and we would appreciate a twenty-four hour turnaround. Often, telephonic communication elicits additional information, so please be accessible.

The initial layout is proofread during staff review and by you. Again, a quick turnaround will be requested. Basically, you check for content, and we design the article.

Miscellaneous

Provide a brief background sketch, including (but not limited to) where you work, your education and e-mail address if you wish to be accessible to our readers. We encourage reader comments and responses to all articles.



Figures, listings, tables, and captions should be provided at the end of the article, or in separate files.

We will accept manuscripts for review, but they will not be returned unless a SASE is provided. Article ideas may be approved in advance, but no guarantee of acceptance is given until the final copy is reviewed.

All artwork must be submitted with the feature article. Send submissions to:

editor@robotmag.com

Mail hard copy and graphics to:

Editor
Robot Science & Technology
2351 Sunset Boulevard, #170-253
Rocklin, CA 95765
USA

NEW Product Update

The Board of Education - What's a Microcontroller?

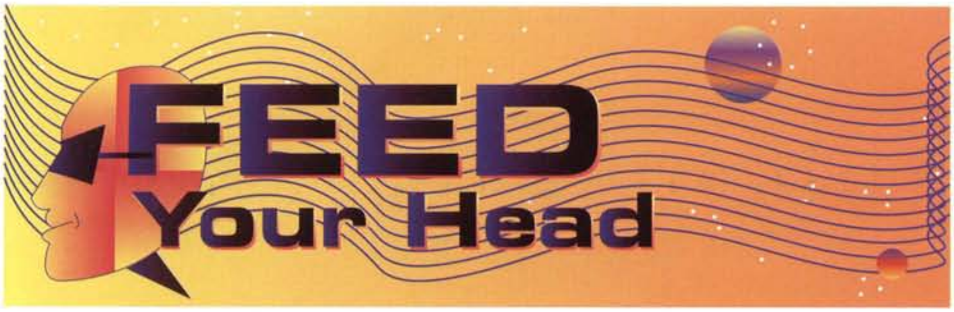
A Tutorial Handbook of Microcontroller Basic Knowledge and Experiments, plus a Companion Breadboard, Make for a Fun and Exciting Learning Experience.

This product is an example of creative thought soundly applied. We here at RS&T are regularly bombarded with inquiries concerning how to get started in robotics. ***This is the answer!*** The ***Board of Education***, a prototyping breadboard, introduces the student to electronic fundamentals without the pain normally associated with things electronic. Schematics, the bane of beginning students, suddenly become understandable as the experiments unfold. It pays to keep a copy of Radio Shack's *Getting Started in Electronics*, by Forrest M. Mims, III nearby while you progress through the lessons, but it is not absolutely necessary – more of a supplemental text, it is a handy reference and primer on the subject. So why is the ***Board of Education*** such a cool addition to electronic education? How about:

- Letting you build circuits without soldering.
- Trace lines that visually show the connections you are making.
- Explanations at each step that are clear and easily understood.
- Plug-in connections that can be undone immediately to correct a mistake.

All these features are present and more. Having made all the connections required for a given experiment, the fun begins. Loading the ***BASIC Stamp Editor*** onto your desktop in Windows and selecting the Stamp2 icon brings up the Stampw.exe icon. Just click on this icon, the funny green diagonal one, and the ***BASIC Stamp*** program is ready to roll though the supplied Programming Cable to your wired ***Board of Education!*** Type in the program, click on **Run,** and you are on your way. But, not so fast. That program was written for you, now you have to modify it to do something different. After sufficient cogitation, you modify the program successfully. What a feeling of accomplishment if you have never written or modified a program before! That is just the beginning of an intellectual journey that you ca pursue through five more experiments. Each experiment get progressively more challenging and rewarding. Learning ma never be this much fun again. And believe us at RS&T, yo will not only be enthralled, you will be delighted and o your way in the fascinating field of ***Robotics!***

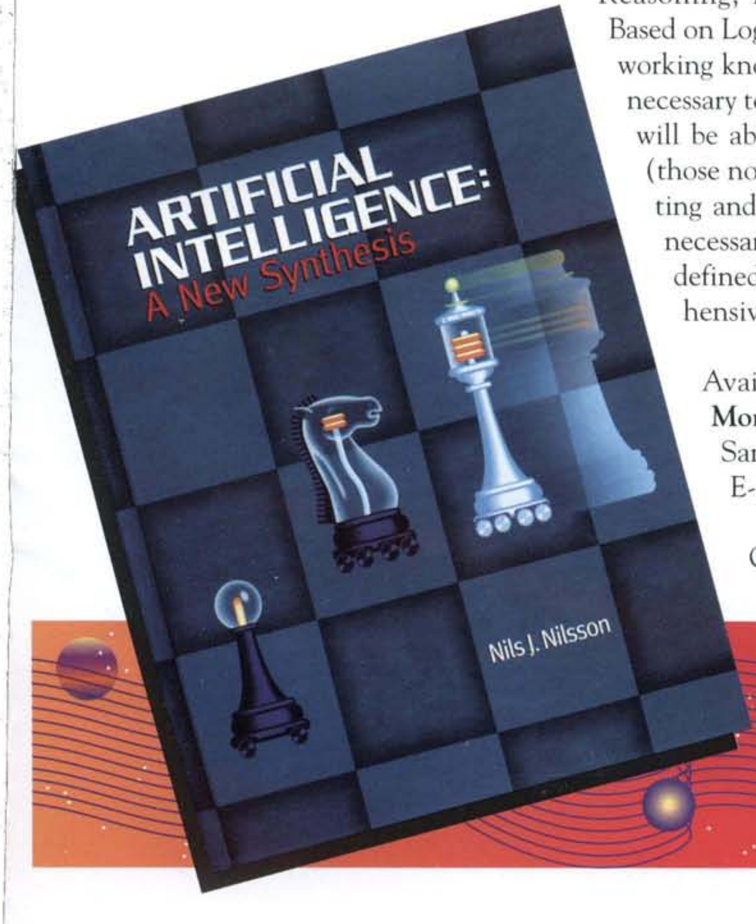
Available from… **Parallax, Inc.**
3805 Atherton Road, Suite 102
Rocklin, CA 95765.
Phone: 916.624.8333 • 888.512.1024
Fax: 916.624.8003
parallaxinc.com



Artificial Intelligence: A New Synthesis

by Nils J. Nilsson

This textbook is designed for use in an educational setting. Entering Artificial Intelligence students, directed and assisted by their professors, will find a wealth of information and ideas coherently arranged and concisely presented. There are five major sections to the book, exclusive of the Introduction. Each chapter in each section is followed by a listing of additional readings and discussion exercises. The work is very well indexed and referenced throughout. There is an extensive Bibliography containing the citations (some forty pages and approximately six hundred entries). After a brief introduction to AI, including an author's definition which is admittedly somewhat broad and circular (as is the nature of the beast), a history, and various approaches to the subject are discussed. The topic of Reactive Machines (Stimulus-Response, Neural Networks, etc.) leads off the meat of the text, which progresses through Searches in State Space, Knowledge and Reasoning, Planning Methods Based on Logic, and concludes with Communication and Integration. A working knowledge of algebra, trigonometry, and calculus is sometimes necessary to follow the logic. However, here is where an astute professor will be able to lead the novice through to solutions. Casual readers (those not familiar with AI and not immersed in an academic AI setting and receiving a daily dose of the subject matter), may find it necessary to refer frequently to the excellent index. Acronyms, once defined in the text, are not usually explained again. The comprehensive index solves this problem handily.

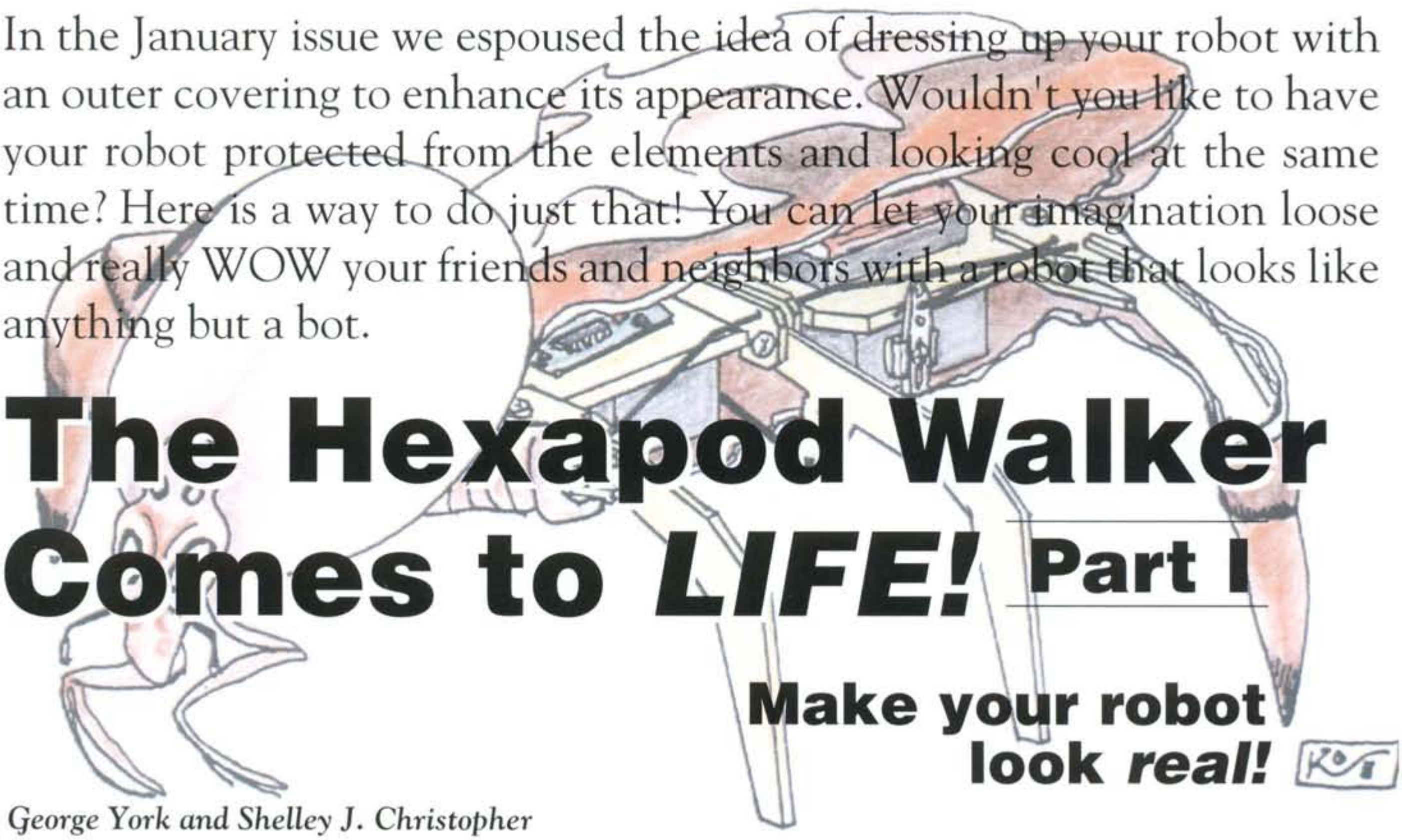


Available from...
Morgan Kaufmann Publishers, Inc.
San Francisco, CA.
E-mail: *orders@mkp.com*.

Order by mail...

Harcourt Brace & Co.
Order Fulfillment Department
6277 Sea Harbor Drive
Orlando, FL 32887

ISBN 1-55860-467-7, 536pp, hardcover, $59.95
S/H included for individual buyer if prepaid, call 800.845.7323 for details.



In the January issue we espoused the idea of dressing up your robot with an outer covering to enhance its appearance. Wouldn't you like to have your robot protected from the elements and looking cool at the same time? Here is a way to do just that! You can let your imagination loose and really WOW your friends and neighbors with a robot that looks like anything but a bot.

The Hexapod Walker Comes to *LIFE!* Part I

Make your robot look *real!*

by *George York and Shelley J. Christopher*

Can your robot use a face-lift? In this series of articles we'll show you how to create 'outer coverings' or rubber 'skins' for a robot – from A to Z. Putting skins on your bot might sound like a daunting prospect. However, when you finish this series you will realize that it isn't as tough as you thought. The process of rubber skin fabrication will be explained in full detail. The articles are designed for beginners who want to give it a try, and others who are experienced and would like to learn a few more tricks of the trade. In a companion series (See ***Building Your Own HEXAPOD Walker Robot*** on page 9), detailed instructions are provided on how to partially assemble *Lynxmotion, Inc.'s H1-KT Hexapod Walker* robot. In that series, we will also explain how to modify the robot by adding an Infrared Proximity Detector (IRPD). This article will use a fully assembled *Hexapod Walker* and illustrate how to bring it to life with full step-by-step instructions on skin fabrication.

Why Skins?

Robots perform certain tasks, but they generally look like ... well, ... robots. Why not create an outer covering for your bot that makes it look *alive*? A robot that is properly configured can be modified to look like an ant, a flying insect, or a Tyrannosaurus Rex. With some time and imagination, almost anyone can transform a robot into a wild animal or prehistoric dinosaur. The possibilities are endless, the advantages many, and the expense often minimal.

One advantage is that skin coverings can create or enhance a robot's personality. A robot with an IRPD can appear to have certain behaviors or mannerisms characteristic of living insects, mammals, or reptiles... you name it. An IRPD-equipped robot can avoid smashing into walls, furniture, and other obstacles. Marrying this reactionary behavior with a spider-like exterior can transform a robot into a creature tha looks real, and acts incredibly *alive* Such a robot can appear to detect a overhead predator by suddenly withdrawing or scurrying away. The effec is a robot that outwardly looks as if i has a multi-faceted personality capab of exhibiting fear, anger, excitement o bravery. A skinned robot suddenly becomes even more interesting and ca attract a fascinated audience.

However, outer coverings not only enhance "personalities" – they can cal attention to specific robotic behaviors The *Hexapod* can walk forward an backward, and turn left and right. Robotic rubber skins enhance this forwar and backward motion by calling attention to the extension of the robot's legs and their range of motion. Additionally, skin coverings can be more than just visually pleasing since their benefits extend beyond the purely aesthetic to the extremely practical. When used to cover a robotic prosthetic arm, skins can make the arm appear life-like, adding perceptive value to people's lives.

Whether outer coverings serve to attract or deflect attention, skin fabrication opens the door to a world of experimentation and creativity. The skinning process can expand an inventive mind and often produces rewarding results. It can be thought of as an extension of robot fabrication since it too requires vision, experimentation, testing, and modification.

It is the willingness to experiment that propels the science of robotics to higher levels of ingenuity. Still, some may fear that a skin will encumber a robot's functions, but skinning a robot can maintain and even enhance its capabilities since an outer covering can serve as a protective barrier. By acting as a shield, an outer covering can protect your bot's brains from accidental damage. Even robots with an IRPD can run into objects, tip-over, or crash down stairs, and a protective outer covering can save a lot of frustration, time and money. Why not create a cool looking robot that has built-in protection? All of this contributes to the bottom line, getting the most out of your robot – aesthetically and operationally.

How big is your 'Bot?

Creating rubber skins for a robot will involve sculpting, molding, casting, painting and final skin attachment. Before diving into these projects, the robot's *size*, *weight* and *range of motion* must be fully assessed. These three factors are significant because they determine the design and materials best suited for your robot's outer covering. The *Hexapod Walker* is our subject for this series, so let's get started.

Size Considerations

The walker is relatively small, standing three inches off the ground. As shown, it has six plastic four-inch-long legs, a width of approximately eight inches, and a length from tip to tail of approximately eight inches (plus or minus an inch to accommodate leg movement). The *Hexapod's* plastic chassis also supports four AA batteries, a 9-volt battery, nylon nuts and screws, control rod wires, and a microcontroller circuit board. Attached to the underside of the body's chassis are three servomotors.

Weight Bearing Considerations

With the robot's size analyzed with some attention to movement, the next consideration is how much weight in rubber skins the robot will be able to support. The *Hexapod Walker's* weight-bearing capability, as suggested by *Lynxmotion, Inc.*, is approximately 12 ounces. This does not include the addition of the IRPD. With the IRPD in the design, it should handle an outer covering weighing 10 ounces.

It is important that the rubber skins stay within this 10-ounce envelope. If the skins weigh much more than 10 ounces, the power source (batteries) may drain too quickly, or the servomotors may be incapable of moving the weight of the structure. Maintaining the integrity of the robot's mobility (its primary function) is the third major factor in this assessment process before skin fabrication begins.

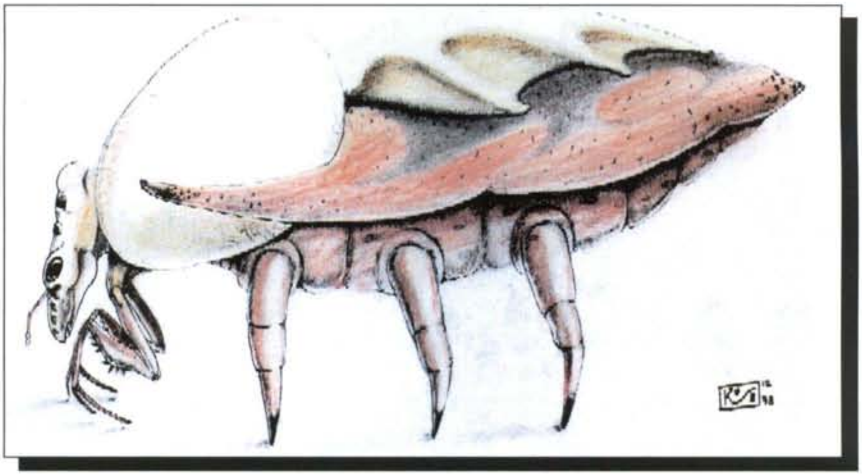
Range of Motion Considerations

A robot's method of movement and its range of motion, are major considerations *before* designing an outer covering. Whether a robot walks, crawls, or rolls, the mode and range of motion information will determine which materials are best suited for creating the robot's skin.

The *Hexapod Walker* actually *walks* while using the IRPD to track its surroundings. The robot achieves mobility by using six legs, which are controlled by three servomotors. These legs have over two inches of vertical leg lift, allowing the robot to walk in an alternating tripod gait. The outer covering must not prevent or hinder this ambulatory movement, so choosing materials that work with the walking function and accentuate the visual effect is crucial to the overall success of the design.

Choosing the Right Material for the Intended Purpose

Keep in mind that an outer covering may consist of several different parts, or pieces (an insectoid creature would have a head, wings, and shell). Each piece could even be made from a different material. For this project, however, rubber materials are best suited to moving with and enhancing the *Hexapod's* leg movements. Many rubber materials are easy to work with and can be found at art supply stores (see Resources on page 62).



Side view of Mantid Hexaptera (MH6).

With rubber, skins can be produced that are thick or thin, stiff or flexible depending on your choice of material. With the *Hexapod Walker* in mind, skins that are too heavy, cumbersome, or binding can obstruct the servomotors' ability to move the legs, and a drained power source could result. The strength of the servomotors is another factor. A certain amount of force is necessary to move both the legs *and* the outer covering while propelling the mass of the robot. Always consider how much skin flexibility is required so that movement will not be impeded.

The *Hexapod* would benefit most from an outer covering made of rubber that is thin and flexible, such as thin cast latex rubber. Cast latex is similar to the rubber used to make Halloween masks. It has good flexible elasticity, is lightweight, and would be less likely to obstruct the robot's leg movements than other, stiffer, materials. Portions of the *Hexapod's* outer covering could be cast with urethane resin. This resin is also lightweight, easily painted (the final step in making skins look real) and quite durable. However, unlike cast latex rubber, urethane resin has little or no flexibility. These skinning alternatives offer distinct advantages and disadvantages. They should be studied closely in order to maximize their benefits.

Other portions of the outer covering could be vacuformed (see the article *Vacuforming on a Shoestring* in the January 1999 issue). Vacuforming portions of the outer covering with a plastic material is another great way to keep weight to a minimum since vacuformed structures are thin and relatively lightweight. However, they are not very flexible, in fact, they are fairly rigid and self-supporting. Since they do not require support to hold their form, no weight is added to the overall structure beyond that of the skin itself.

The *Hexapod* can support 10 ounces with the IRPD mounted, therefore, any or all of these skinning mechanisms could be employed. It is important at this "planning stage" to consider materials that offer flexibility, durability, and longevity, while enhancing the function and purpose of the robot.

Creature Creating – Get Inspired!

Once the robot has been analyzed, and the materials selected that are best suited for enhancing the robot's function, the fun of "creature creating" begins.

Inspirational resources are virtually limitless. A good place to get started is at a public or university library. Most libraries have collections of reference books on dinosaurs, fossils, insects and reptiles. Read scientific magazines specializing in insects and mammals. Magazines, such as *National Geographic*, are also a good way to study animals from various regions of the world, especially for examples of unique eyes, coloring, size and shape.

Movies are anoth great source of insp ration. Watch *Jurass Park* and pay atten tion to the dinosaur movement. Watc the *Discovery Chann* and assess the move ment of birds, giraffe or deer. Pay particul attention to the wa spiders crawl, mic scurry and birds fl Computers can als be employed in thi creative process. Scan pictures into computer if software such as *Adob Photoshop* is available. A photograph a person can be scanned and modifie into an alien creature. Interesting featur from several different photos can be ex trapolated (or 'morphed') to create unique creature.

Search the Internet for "insects" and se what comes up. Colorado Stat University's Entomology Departmen has a great web-site (www.colostate.ed Depts/Entomology/ent.html) featurin numerous close-up, full color photos various insects.

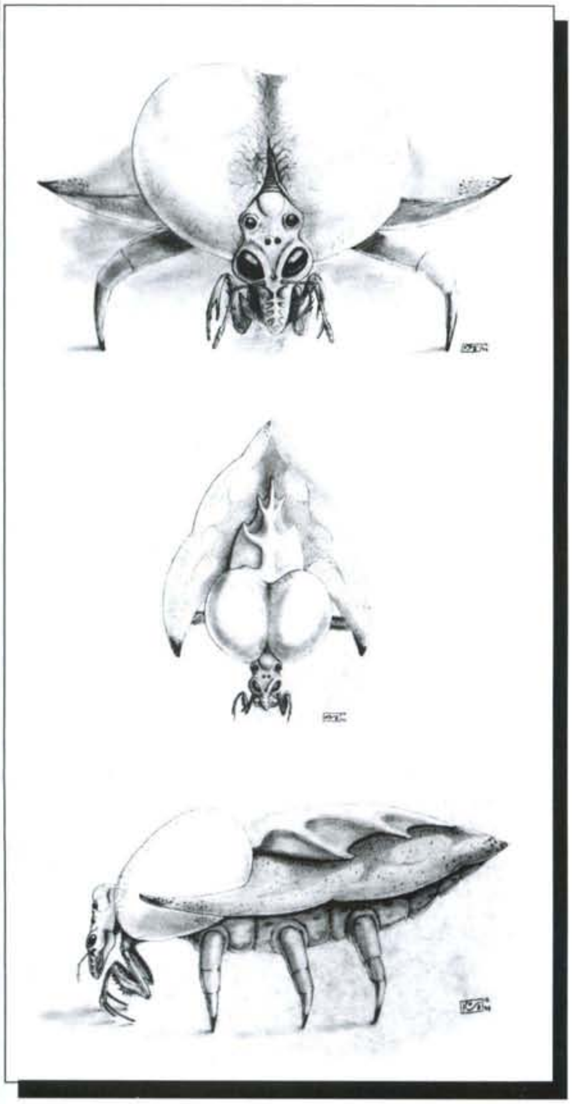
The best source, however, is always *you own imagination*. Get a piece of pape and a pencil, and start sketching!

Give Your 'Bot a Boost!

We designed a unique outer covering f the *Hexapod Walker*: an alien insectoid Several sketches were drawn before de ciding on the design pictured. Th sketches illustrate a view of the creatur from the top, front, and side perspec tives. These sketches will be useful as constant reference tool once the ski fabrication process begins.

They are useful 'blueprints' for the fab ricator to follow, allowing room for th robot's various components. Note in th sketches that the head is small, but th brain is quite large. The large brain-lik structure provides space for future modifications. It has enough room to hold the IRPD, as well as additional circuits or controls that may later be added to enhance the robotic creature. This brain-like shape was also selected because it infers a level of intelligence that the insectoid creature will exhibit when using its IRPD to react to its environment.

There will also be an understructure, or 'underbelly.' This will protect the servos and give an overall balanced appearance to the creature. The structure will also provide attachment points for the leg skins and afford some protection for the linkages that move the legs. There will be a 'top shell' which will function as a protective covering (real insects have a similar covering known in the scientific world as "chitin"). This top shell will protect the batteries and circuit board, giving the design symmetry and a realistic appearance overall. Not only will the "protective plate" provide additional protection for the robot's controls, it will also permit access to the batteries and circuit boards.



Three views of Mantid Hexaptera.

Always Plan for Access

Ready access to the robot's controls is necessary because the *Hexapod Walker*, like any robot, could malfunction after the skins have been attached. Batteries will need to be replaced occasionally and the servos may require some adjustment/modification. All these activities demand easy access to the internal mechanisms of the robot. Since the *Hexapod's* microcontroller or 'brain' is attached to the top of the structure, a hard protective plate is the most logical choice. This plate may be completely removable, or it may attach to the robotic structure with a hinge or *Velcro*™.

During the sketching process, not only is the creature's design conceptualized with these mechanical factors in mind, but such artistic details as eyes and skin coloring are also considered. Still to be decided is whether to modify the IRPD by placing its transmitting portions and detection devices in the creature's eyes. This would add to the aesthetics of the design, as well as the 'realistic' function that is intended.

When the creature's design was finalized it was decided to name the alien insectoid a "Mantid Hexaptera," "MH6" for short, which refers to the robot's name and six-legged structure. The name was given to create a personality, and a scientific- sounding name referring to its physical characteristics adds to the credibility of the overall design. It incorporates characteristics of a Praying Mantis as well as other wingless insects such as "shield bugs," larvae, and nymphs.

Remember that these artistic sketches provide an *example* of what could be created when outer coverings are designed. The designs are intended to give the beginner an idea of what can be achieved. A novice skin fabricator may want to follow the skin-making process by constructing a replica of this design. Alternatively, to increase the challenge, you can create a design of your own. The sketches shown may be modified to one's own interests and creative ability. Whatever the case, skin fabricators should always follow their own inclinations, and allow their creativity and imagination to materialize in a creature that is truly unique.

Some Final Considerations...

Creating skins for a robot may seem like a never-ending process of important factors and considerations, but skin fabrication can be relatively easy and extremely exciting as you bring your design to life! The thrill of accomplishment derived from robot fabrication can be further enhanced by creating an outer covering of your own.

This series of articles will bring to your attention just about every facet of the skin-making process. Other subjects to be discussed will include the incorporation of a "cooling system" in the skin design. The robotic skin shell may need a small gap to allow heat to escape, and/or holes drilled in the bottom of the structure for air cooling circulation purposes. Other modification issues to be discussed include adding antennae to detect movements and adding arms that move around and pick up objects.

Skin fabricators can be inventive by adding servos, detection schemes, or even an onboard camera. It may be possible to modify the robot to add visual recognition capability. A microprocessor that uses algorithms could be added to allow the robot to recognize what it's observing. A radio control receiver on the bot to facilitate communication could also be incorporated. Since the process of creating skins is often experimental, the modification possibilities are limited only by your own imagination and your bot capabilities.

Check out what's coming next...

The next article will give full details and instructions on th sculpting and molding processes. This is when the hands-o fun begins!

RS&T

Resources

Douglas and Sturgess, Inc.
730 Bryant Street
San Francisco, CA 94107
Phone: 415.896.6283

Tap Plastics, Inc.
6475 Sierra Lane
Dublin, CA 94568
Phone: 800.246.5055

Lynxmotion, Inc.
104 Partridge Road
Pekin, Illinois 61554-1403
Phone: 309.382.1816
Fax: 309.382.1254
lynxmotion.com

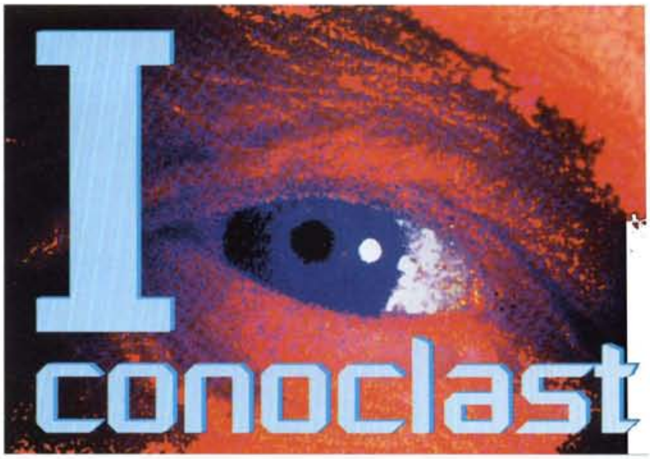
A Reference:
Alien Empire, An Exploration of the Lives of Insects
by Christopher O'Toole, Harper Collins Publishers

Shelley J. Christopher *is a freelance writer who holds B.A. and an M.A. in English, from Califo nia State University, Sacramento. She is writer who has focused much of her tim in the literary world. She is currently wor ing with professionals on writing projec such as this robotic construction and ski fabrication series. She can be reached xchrisr@aol.com.*

George York *is an electronics engineer and mechanic designer from Sacramento, California. A owner of YFX Studio, he has over 20 yea of experience with robots, animatroni and special effects. His many works i clude a life-like bust of Julius Caesar f Caesars Las Vegas, and a mermaid t for the Foote, Cone & Belding advertisi firm. York's expertise in robot fabricatio and his artistic ability, culminate in this pa allel series on robot construction and skin fabrication. H can be reached at yfxstudio.com*

WHAT'S A ROBOT TO DO?

Why are we still trapped in conventional thinking?



Say you're taking a quiet walk one fine evening when all of a sudden an alien spacecraft lands in front of you and a dapper looking creature emerges speaking your language. It explains that, for a short time only, customized mobile robots will be given to selected individuals as part of a galaxy-wide marketing survey. You have been chosen to receive one, and the question posed to you is: "What would you like your robot to do?" There's some fine print to the offer so forget the obvious illegal, unethical, and cosmically immoral applications. What application do you choose (specificity counts)?

The robot community seems to have exercised very little imagination, not to be confused with fantasy, when it comes to choosing robotic applications to pursue. A practical and specific choice of what's a robot to do seems to be a real conceptual problem. When talking to fellow gearheads, cruising robot web sites, or reading publications, one is introduced to a lot of different kinds of mobile robotic applications. There are the useful but mundane, such as grass cutting, carpet vacuuming, or personal service, like 'go get me a beer robot!' There is the hazardous task robot for cleaning up toxic spills, exploring volcanoes, cleaning up land mines, etc. There's the tireless observer robot for sentry, sampling, or messenger tasks, and there are the entertainment robots.

Outside
Think
the box.

There are other categories as well but my point is that most suggestions for mobile robot applications don't extend our capabilities or break new ground. Rather the applications tend to be something that is either currently being done or something that has a cost trade-off advantage. Home vacuuming, for instance, is something over which an individual has control: One can either do or hire it done. The difference between somebody or something doing it is one of cost and novelty, with reliability and effectiveness being a toss-up.

A shortcoming of many of these robot applications is that they tend to have the robot perform the task the way that humans do. Airplanes don't fly the way birds fly, personal conveyance vehicles don't walk, ships don't swim. So why do robots have to mimic the approach people take to perform a task? Is the best way for a robot to 'go get a beer' to recognize the verbal command, navigate to the fridge, open it, find, select and grasp a beer, close the fridge, navigate back to the origin of the command, or the new position of the speaker, and then present the beer container? Gotta be a better way.

A related point that seems to have been overlooked is that the robot's operating environment may have to be adapted to work with the robot. For instance, cars need roads and a vast infrastructure of combustible products, a service industry, associated products, etc. Perhaps the work environment suitable for a robot will have to go halfway to meet the task. In the above example, if the fridge were modified, perhaps like a vending machine, then the robot could select and convey the beverage of choice without costly and troublesome vision and manipulator systems. The robot could navigate throughout the house on tracks by using the only consistently uncluttered part of the home landscape, the ceiling. I only suggest these specific environmental modifications to argue that we need to break out of modeling robots on human behavior.

I admit to being somewhat disappointed with the amateur community as well. Not the beginners and tinkerers, but with the more serious amateurs who write the columns in various magazines. We look to these guys to show us what can be done, to challenge and motivate us. Instead we keep seeing the same line-following robots, light-seeking robots, or go-'round-in-circles robots. I'm not expecting R2D2, but surely robotics has more to offer than automated toys.

Imagination. We need more imaginative thinking. I guess we're just going to have to show the professionals the way.

April 18, 1999
Trinity College , Fire-Fighting Home Robot Contest

Hartford, Connecticut – www.trincoll.edu/robot

The challenge for entrants is to produce a robot that can move through a model of a single floor house, detect fire and put it out. There is a junior division for high school students and younger, and a senior division. Other events include seminars and a robotics exhibition. For information, contact jmendel141@aol.com. *See RS&T's Premier Collector's Edition for robot construction ideas. Also see www.RobotMag.com.*

April 22-24, 1999
FIRST National Championship

Orlando, Florida (EPCOT Center, Walt Disney World) – www.usfirst.org

The annual FIRST (For Inspiration and Recognition of Science and Technology) Foundation National Championship robotics tournament will be held at EPCOT, and 220 teams are scheduled to compete. Those teams have already competed in seven regional tournaments where they honed their control and teamwork skills, and pushed their robots to the limits. They are now ready for the big test. As during regional contests, qualification matches at the national championship will determine which teams qualify for the finals.

April 30-May 1, 1999
Student Robotic Challenge

Saginaw Valley State University, Michigan – www.sme.org

The 13th annual Student Robotic Challenge, formerly the Student Robotics Automation Contest, is sponsored by Robotics International of the Society of Manufacturing Engineers (RI/SME). The organization has historically identified robotic technological trends and highlighted them at this annual event. This year's SRC provides students with the opportunity to demonstrate knowledge and understanding of manufacturing processes and controls through robotics and automation competitions. Two new contests will require on-the-spot performance: **Flexible Manufacturing** participants will be given a change order for a product, and must reprogram a manufacturing production line to affect the change. The **Robot Simulation** contest requires students to program a robot in a virtual reality environment.

May 8, 1999
Western Canadian Robot Games

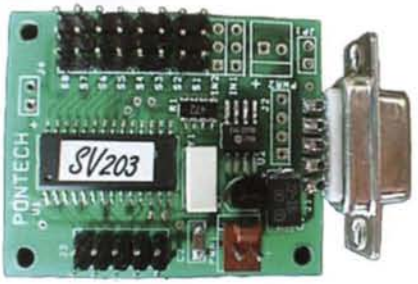
Calgary, Alberta, Canada – www.robotgames.com

The WCRGs are to be held at the Southern Alberta Inst tute of Technology, and are dedicated to the advancemer of interest in robotics. This year's events will include rc botic sumo wrestling, atomic hockey, Fire-Extinguishir Autonomous Robot (FEAR) contests, biological, electroni aesthetic and mechanical (BEAM) SolarRoller racing ar photovore contests, workshops and lectures.

May 24-27, 1999
Eastec '99 Exposition and Conference

West Springfield, Massachusetts – www.sme.org

This metalworking and manufacturing exposition is to t held in West Springfield, MA. The conference takes plac during May 24-27 at the Springfield Marriott Hotel in dowr town Springfield, and the exposition during May 25-27 a the Eastern States Exposition Grounds. The exposition wi feature the latest technology in machine tools, metalworl ing and manufacturing processes. Over 30 workshops wi be conducted during Eastec '99, as well as advanced course and clinics on various aspects of manufacturing and produc tion technologies. Sponsored by the Society c Manufacturing Engineers (SME), American Machine Toc Distributors' Association (AMTDA) and the Associatio for Manufacturing Technology (AMT).

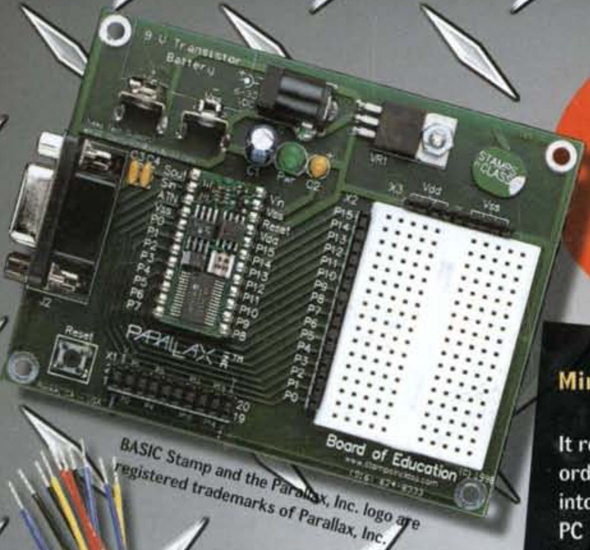

SV203
PONTECH


Pontech SV203
File
Setup
Device
Port
Term
S1
S2
S3
S4
S5
S6
S7
S8

BASIC ToolBox

#28190

Microcontroller Kit for the Electronics Apprentice!



BASIC Stamp and the Parallax, Inc. logo are registered trademarks of Parallax, Inc.

$180 value for

ONLY $149

+Shipping

Offer extended through May 31, 1999

Minimal Hardware Required!

It really doesn't take many tools to get started with BASIC Stamps - just an ordinary PC running DOS or Windows 95/98/NT. Loading your first program into a BASIC Stamp is a matter of connecting the Board of Education to your PC and running our software. We're even providing a couple of LEDs and resistors for your first project!

Learn BASIC Stamps using the Board of Education

During the month of February and March we're offering the BASIC ToolBox --which is similar to the BASIC Stamp II Starter Kit except we've replaced the carrier board with the new Board of Education. BASIC Stamp I/O pins, power and ground connections are accessible adjacent to the breadboard.

For beginners the Board of Education temporarily eliminates the need for soldering iron, and for advanced engineers it's a quick BASIC Stamp project area.

The **BASIC ToolBox #28190** ($149 + shipping) includes: BASIC Stamp II module, Board of Education, ten (10) jumper wires, BASIC Stamp Manual Version 1.9, (2) each LEDs and resistors, DOS & Win95/98/NT software, power supply and serial cable.

What's a Microcontroller?

For the real beginner take a look at our educational web site http://www.stampsinclass.com and download the "What's a Microcontroller?" Experiment Set, or purchase a copy for yourself (#28123 - $19)!

PARALLAX INC

To order: 916.624.8333 More info: http://www.parallaxinc.com