Appendix A

Robot Glossary

This appendix section briefly describes each of the robot projects discussed in this thesis. The glossary is organized in alphabetical order by the robot's name.

Baygon, the Happy 'Bot *Baygon* was a robot produced by the team of Jorge A. Calvo, Jose Luis Elizondo, and Michel Montvelisky for the *Robo-Cup* contest. They realized they would not be able to qualify for the main contest, so they invited their friends to see their robot perform a surprise song-and-dance routine during the qualification round. See page 181.

Bertha A contestant in *Robo-Puck*, *Bertha* was a successful puck-fetcher that used a single infrared sensor to locate the puck. *Bertha* was designed by Cheryl Aittama and Chris Palmer and is pictured on page 80.

Blind The second robot of the David Hogg/Katie Lilienkamp team, *Blind* competed in the *Robo-Puck* contest with a method for capturing the puck that did not require sensing it. *Blind* exploited the fact that the initial position of the puck was a known distance from the initial position of the robot. *Blind* is pictured in a three-sequence animation shown beginning on page 177.

Crazy Train The champion of the *Robo-Pong* contest, this robot was a collector-style tank-like machine which scooped balls into its body. *Crazy Train* was built by John Kerwin, Gregory Gancarz, John Lum, and Dave Lum. See page 136.

50/50 A contestant in the *Robo-Pong* contest, this was an aggressor robot that used a radar-like infrared sensor that could locate and track the opponent robot independently from its own movement. Built by Michael Gull, Henry Chung, Alex Wu, and James Sarvis, *50/50* lost when it apparently detected a reflection of the opponent's infrared beacon rather than the beacon itself. See page 255.

Groucho A contestant in the *Robo-Pong* contest, *Groucho* was a capable collector-style robot that was so named because it had a "face" that employed LEGO beams to simulate the eyebrows on the famous comedian. Also, its creators, Ed Tobin, Matthew Lee Domsch, and Adam Skwersky wore "Groucho glasses" on the eve of the contest. *Groucho* is shown in Figure 4-1 and discussed on page 138.

Juicy Chicken A contestant in the *Robo-Cup* contest, this robot was a shooter that had a separate baffle mechanism intended to assist the shooter in delivering balls to the goal. The baffle would position itself immediately outside the goal and deflect balls shot toward it into the goal. Additionally, the baffle carried an incandescent lamp which the shooter could track for aiming. Created by Bill Kaliardos, Thomas D. Wu, and Paula Bontá, *Juicy Chicken* was a complex design which was never successfully debugged. See page 86.

Mutton Jeff A sophisticated dual-robot design created by the brothers Matt and Tim Wall, *Mutton Jeff* solved the *Robo-Pong* contest with separate offensive and defensive robots joined by an umbilical link. The offensive half, which carried the computer, shot balls over to the other robot's side while the defensive half prevented the opponent from crossing over the playing field median. See page 179.

The Shotgun A contestant in *Robo-Puck*, this robot was notable for its creative use of infrared sensing in a strategy based on shooting a claw at the puck. Because of its speed, the robot was unbeatable when it fired accurately, but it sometimes missed its target. *The Shotgun* was created by Rajeev Surati and Tim Wall and is discussed on page 79.

Stupid The first robot by the Hogg/Lilienkamp design team, *Stupid* solved the *King of the Hill* contest without using any form of electronic computation. A clever mechanical design caused *Stupid* to climb uphill when power was applied to its motors. It defied the organizers' attempt to create a contest that required programming as part of a viable solution. See page 176.

Stupid Scorpion A contestant in *Robo-Pong*, this robot was created in a last-ditch effort after multiple attempts at more sophisticated robots had failed. *Stupid Scorpion* simply drove back and forth, reversing direction each time it detected that it had gotten stuck. Remarkably, it consistent hit more balls onto its opponent's side than it brought back onto its own side, and placed third overall in the contest. None were more suprised by its success than its creators, Richard Davis, Ben Renaud, and Francis Njie.

The viability of robots like *Stupid Scorpion* was interpreted as a flaw in the *Robo-Pong* contest, and subsequent contests were designed with a higher degree of difficulty (see discussion beginning on page 87).

Appendix B

Contest Design

This appendix provides the full text for each of the Robot Design competitions from the years 1989 to 1992.

B.1 King of the Mountain, 1989

```
                   The Third Annual 6.270 Contest:


                         BATTLE OF THE LEGOS



             Sponsored by SIX APPEAL, The Course Six Social Group
                  with funding from EEC S and Microsoft Corp.

            Organizers:  Mike Parker, Randy Sargent, and Fred Martin



DESCRIPTION
-----------

Each contestant will be given all parts necessary to build a
computer-controlled LEGO robot, including electronics, LEGO, motors,
and sensors.  We will spend IAP building the robots, and will hold
several workshops to discuss different aspects of the design,
including the electronic hardware, mechanical hardware, and software &
interface.  In February, a competition will be held to determine the
winning machine!  Over $500 worth of prizes will be awarded, and all
```

contestants will be able TO KEEP their $150 Computerized Lego Robot
Kits.

THE PARTS & PLAYING FIELD

*** The Game ***

The competition will be a King of the Hill battle. The goal of contest will be to reach the highest elevation on a hill-shaped playing field, and to do so in an elegant and entertaining fashion. Limited offensive and defensive weaponry is encouraged; this will be further described.

The playing field will be shaped like a slice off of the top of a hemisphere, with a flat patch at the top. Viewed from above, it will be a perfect circle with a diameter measuring approximately eight feet. The hill will have a positive slope in the direction of its apex at all points, except for the flat patch at the top.

There will be obstacles on the hill, in the form of rectangular or cylindrical blocks, mounted firmly into the playing surface. These obstacles will be no larger than one foot on an edge or in diameter, and will be painted bright red.

The playing field will be suspended above the ground, and machines may be pushed off or may fall off of the playing field. Cushioning material to catch falling machines will be installed.

The hill will be painted flat black in color. A highly reflective mylar "warning track" will be mounted in a circular stripe around the edge of the field. With an appropriate light sensor, it will be easy for a vehicle to determine that it is located on this warning track.

```
=========== extremely crude image of cross-section of hill ============


                           top of hill
   an obstacle __         _____________
              |  |   -----             -------                 warning
           ---|  |---                         ------------      track
 ****------                                               ------****



=======================================================================
```

*** The Materials ***

Each entrant will receive a hardware package valued at approximately $150, which will become the property of the entrant after the contest. This package will include:

* an ample supply of LEGO, including beams, axles, gears, wheels, and other parts from the LEGO Technics line.

* electronic hardware and plans for building a robot serial interface, which is compatible with any standard serial line.

* actuators, including motors, solenoids, lamps, and buzzers.

* sensors, including photocells, touch and mercury switches, and phototransistors.

* miscellaneous supplies, such as perfboard, glue, and wire.

Contestants will also be given documentation that will include schematic diagrams for various sensor designs and other helpful suggestions.

*** The Computer ***

All machines must be solely controlled by computer. The afore-mentioned "robot serial interface" is ideally suited for this purpose. You may use any computer you wish, and write your control programs in any language you wish. Software for accessing the serial port of standard Athena VaxStations will be provided, and VaxStations will be available the eve of the contest.

*** The Robot Serial Interface ***

The standard interface can power up to four motors, and receive information from up to eight sensor. Following is a technical description of the interface.

When the robot controller receives a serial byte from your computer, it writes it out to a register. A pair of "motor driver" chips (supplied) are wired to this register. Each two bits of this eight-bit register control one motor. Writing a "1-0" into a certain two-bit field will turn the corresponding motor on in one direction, while writing a "0-1" into the same two bits will turn the motor on in the other direction.

Each of these four "motor ports" can be used to control TWO non-directional actuators (e.g., lamps) in an on-off fashion.

The robot controller interface continuously broadcasts eight bits of digital sensory information back to the host computer, as standard serial bytes. Assuming a serial rate of 9600 baud, you will receive sensor samples at nearly 900 Hz.

A simple hack to get "analog" sensory information: use your analog sensor (such as a photocell) to control the timing cycle of a 555 counter chip. The higher the resistance of the sensor, the longer the timing pulse; tune the thing to give you pulses in the range of 1 ms to 100 ms.

Run this timing pulse into one of your eight sensor ports. Now, back at your computer, write software to count the number of sequential samples that are 1's. This will tell you how long the timing pulse is, which corresponds to the amplitude of signal received by your sensor.

This scheme will give you a data rate of up to 10 Hz, which is not bad.

THE OBJECTIVE AND JUDGING

The objective of the contest is to be King of the Mountain! to climb the highest up the mountain by the end of a two minute round. And, to do so in an elegant fashion, demonstrating ingenious and efficatious design.

Robots will battle in randomly chosen groups of three. In every battle, the performance of each machine will be scored. For each level of the contest, the robots averaging the highest scores will be selected to compete on the next level. The final score of every robot will be the average of all its battle scores. The robot with the highest average score will win the contest (2nd and 3rd prizes awarded also).

Scoring and selection of robots for battle will be done by a panel of three judges. Half of a robot's score in a battle will be based on its ACHIEVEMENT at being king of the mountain (e.g., it having reached a higher elevation than the other robots in the battle round).

The other half of the score will be qualitatively based on the robot's

* INGENUITY of design (e.g., a smoothly operating machine)

* AESTHETICS of visual presentation (machine AND contestant)

* CLEVERNESS of approach to solving the problem

* ENTERTAINMENT VALUE as judged by audience reaction to the machine's performance.

The three impartial judges will be available for questioning at the robot-building sessions, especially for questions on what they are looking for in these qualitative areas.

THE RULES
----

1. In your quest to be King of the Hill, you may decide to equip your vehicle with various offensive and/or defensive weaponry. Violence (in good spirit, and only between robots) is encouraged. Absolutely no violence is allowed between people or people and machines. This means that dangerous machines (or contestants) will not be allowed.

2. All vehicles must be controlled through the robot serial interface provided.

3. All entries must mount their interface on the top of their vehicle, in a position where the interface connector can easily be accessed.

4. LEGO is the sole material to be used for structural means.

5. Non-LEGO materials (such as motors, sensors, and other hardware) must be "LEGOized" by gluing LEGO parts to them, and then mounting them on your machine using LEGO.

6. Only the motors that are provided may be used to power your vehicle. The motors must be operated with the power supply included in the basic kit, and in the standard way.

7. To help machines find each other, each of them will carry a halogen light bulb and an electronic buzzer.

    a. Both of these items must be mounted in an unobstructed fashion. The light must be mounted so that it is visible omni-directionally, and the buzzer so that its sound is heard (approximately) equally in all directions.

    b. The lamp and buzzer will blink and sound synchronously at a standard rate of about 4 Hz, using a circuit provided.

8. To encourage different designs, you will be given a sum of "contest discretionary funds" which may be used to purchase components to supplement the basic package already described. With this "funny money" you may wish to purchase additional LEGO parts, sensors and electronics, or other materials that will be available.

9. Parts trading between mutually consenting contest participants is NOT encouraged. Exceptions may be made in the case of (a) a shortage of contest parts, and (b) items being traded of comparable contest value.

10. Only materials provided by the contest sponsors may be used on the vehicle.

With regards to all issues of machine customization and standardization, decision of the judges is final. These rules are subject to amendment and interpretation by the judges.

ADMINISTRATION

Robot building sessions will take place Tuesdays, January 10, 17, 24, and 31, in the Electronic Classroom (37-312), from 6-9pm (with possible additional sessions on Tuesdays, February 6 and 13).

The Electronic Classroom is equipped with a MicroVax Athena Workstation for each contestant to test the control of his robot. (Contestants can also bring in their own computer.) We will also be building THE MOUNTAIN for contestants to test their robots.

Attendance to all sessions is expected. Here is a tentative schedule:

10 Jan	- Introduction. Passing out Kits and Rules
17 Jan	- Building the computer interface and sensors
24 Jan	- Building the LEGO robots, Mountain available for testing
31 Jan	- Programming the robots
6 Feb	- Testing and optimizing
	Contest food, ad, and prize planning
13 Feb	- "Dress Rehearsal" Contest
	Mega contest advertising
20 Feb	- Contest! 7pm in 34-101

This $5500 contest is being extensively funded by the EECS Department and Microsoft Corp. However, the department has requested that each contestant cover a small portion ($50) of the $150 of LEGOs and electronics being given him, to indicate his seriousness and to help cover costs. This is actually a great deal, since you get to KEEP all the LEGOs and electronics and any prizes you win!

This $50 entry fee is due and payable now. Please make checks out to "MIT SIX APPEAL" and write "6.270 Entry Fee" in the comment area (your check is your receipt). Please send checks to Michael B. Parker (address below). YOUR ENTRY FEE MUST BE RECEIVED BEFORE THURSDAY, 22 DECEMBER, 1988 IN ORDER TO GUARANTEE YOUR ENTRY IN THE CONTEST!

Happy Battles!

PS: Check your e-mail regularly for further notices!

CONTEST ORGANIZERS

Michael Parker (contest coordinator)
East Campus Monroe Rm 303
phone 225-6303

```
    E-mail   mbparker@athena.mit.edu

Randy Sargent (technical coordinator)
    Senior House Runkle Rm 604
    phone    225-6654
    E-mail   rsargent@athena.mit.edu

Fred Martin (technical coordinator)
    Bexley Rm 401
    phone    225-9641  253-9783
    E-mail   fredm@media-lab.media.mit.edu
```

B.2 Robo-Puck, 1990

OBJECT

To design and build an autonomous robot to keep the "puck" away from you opponents. The winner is the one who makes the last contact with the "puck."

Puck

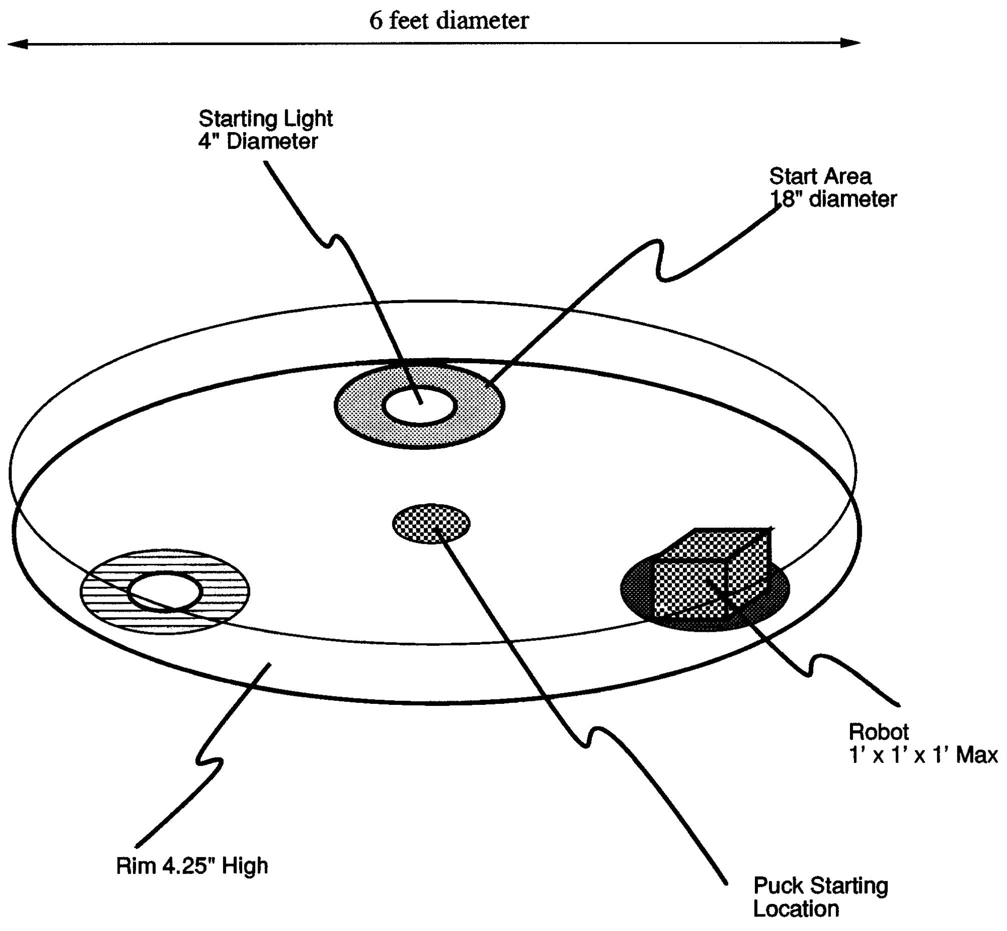
- The puck will be a hockey puck that will be on some type of ball bearings or wheels.
- The puck will have batteries on it so that it can emit infrared light. The infrared light will be one way machines can detect the puck.
- The puck's infrared beacon will be placed 1 foot above the playing surface. Best detection will be in the 1 foot plane.
- The infrared beacon will be broadcasting at a 40 KHz. This frequency will be easily detected by the Sharp IR detectors that are supplied.
- The puck may not be altered or destroyed in any way.
- The machines may not obstruct the infrared beacon from other opponents.
- The puck may not be intentionally flipped.
- If the puck inadvertently flips over, the judges will immediately upright it at the point of inversion.
- The puck may not be lifted.
- The infrared beacon must always remain in the 1 foot plane.

Period of Play

- A period will last 100 seconds. Motor and actuator power will be allowed during the first 90 seconds, and other stored energy will be allowed during the remaining 10 seconds.
- The round will be started by the judges turning on the starting lights for the beginning 5 seconds.
- The judges will place the machines on the playing field at a random angle with repect to the center. The angle at which the machines will be placed will be the same for all machines during the same period.
- The contestants will have 30 second to place their machines on the field from the time the judges call them.

ROBOPUCK

6.270 Design Project Rules and Restrictions



- Level Circular Playing Field
- Three Contestants and one "PUCK"
- Contestants arranged symetrically.
- Each Contestant Beginning at a different colored circle.
- At center of starting circle is a lighted circle used for starting the machines.
- Clear plexiglass walls to prevent machines and puck from falling off the table.
- The playing surface may not be permanently altered or destroyed.
- All evidence of an entry must be removed within 30 seconds after the end of the round.
- The puck will start in the center, equidistant from all the contestants.

Figure B-1: *Robo-Puck* table specifications

- The contestants must stand a given distance away from the playing field. Any contestant that makes an attempt to touch their machine during the period of play will automatically be disqualified from the period.
- The machines must have their own internal clock which cuts off power to the motors at the end of 90 seconds. Any machines that continue to supply actuator power after 90 seconds will be disqualified.
- The contest will be an double elimination depending on the time constraints on the contest.
- All periods will have three players except those periods due to the elimination format only two people are scheduled to a period. In this situation, a placebo constucted by one of the organizers will be the third machine in the period.
- Machines may not be allowed to destory their opponents microprocessor board.
- Machines may not try to destroy other machines broadcast or detection beacons.

Control

- All entries must be solely controlled by their onboard computer. There in no human intervention once the period begins.
- All entries must be capable of broadcasting visual light at 2, 3, and 5 Hz. This visual light beacon will be used to detect other machines. Machines failing to meet this specification will be disqualified.
- Judges will assign frequencies for visual lights to the machines in the beginning of each period.
- After the start of the contest, there can be no change to the robot's program, or configuration switches made by the contestants.
- Infrared light may not be emitted from the robots.
- Machines may not use reflective material such as metal to reflect the infrared light from the puck.
- Parts that are likely to damage the internals of machines (ie. small pieces of wire, or barb) may not be dumped on the playing surface
- Any lego that is advertently dumped on the playing surface as obstaces will become the property of the 6.270 course.
- The two 6V batteries may be connected in parallel to provide greater currents, but may not be used in any fashion to provide greater voltage.

Structure

- All entrants will be given an equal amount of Lego and Non-Lego supplies including:

  * an ample supply of LEGO, including beams, axels, gears, wheels, and other parts from the LEGO Technics line.

  * electronic hardware and plans for building a robot controller card, which can be programmed through any standard serial line.

  * actuators, including motors, and solenoids.

  * sensors, including IR dectors, touch sensors, photocells.

  * string, rubberbands, wire, solder

  * virtual material–glue, epoxy, WD40

  * The package will be valued at approximately $300.

- The visual broadcasting beacon be at least 2 inches above the highest structural piece in the verticle plane above the machine. The light must be carried by the moving part of the robot.

- Only Lego may be used as structure.

- No Legos may be glued together.

- No Legos except for the base plate may be altered in any way.

- A non-lego part may be attached to at most 5 lego parts via glue.

- Wire may only be used for electrical purposes, and not structural.

- Rubber bands may be glued to wheels or gears to increase the coefficient of friction.

- Only Lego rubber bands may be used to provide energy.

- Contestants may not alter the structure of their entry once the contest has begun, but may repair broken components between rounds if time permits.

- The dimension of the machine may not exceed an imaginary 1 foot cube at the start of each period. (an exception is the transmitting and recieving beacon) Entries may however expand once the period has begun.

- Entries must be built completely from kit parts. All other parts must be removed before the beginning of the round.

- Lubricants may be used only to reduce friction internal to the machine.

- For orienting the machines, an arrow pointing to the front of the machine must be placed at a visible location for the judges.

Scoring

- The winner(s) will be the last machine to come in contact with the puck at the end of a given period.

- Contact with the puck is as follows: The microprocessor board must touch the puck through LEGO. Therefore if the machine is touched by a piece of lego that is connected to the rest of the machine by a piece of string, it is not counted as a legal contact.

- Legal contact of the puck is contacting the puck only (touching the electronics or the beacon is not a legal contact).

- The judges will decide any discrepencies in the period.

- One point will be awarded to the losers of the period.

- Entries will compete only against other entries with the same score. ie. Machines with 1 point will compete against other machines with one point, and undefeated machines will compete against other undefeated machines.

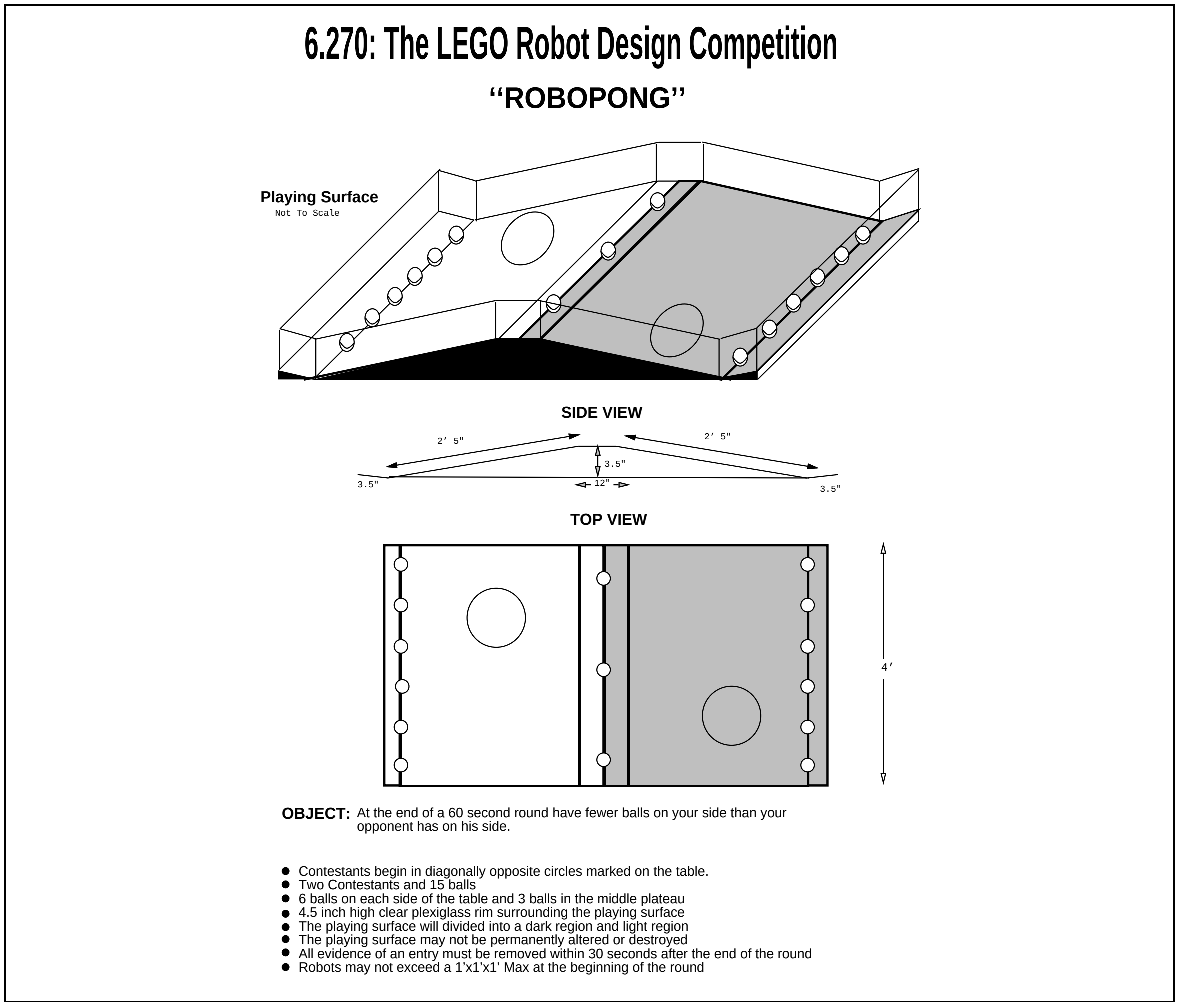
- Every round, a person with two points will be eliminated.

- In a round containing a placebo, the contact made by a placebo will not count. Therefore the last machine to make contact with the puck will be determined from the other two machines in the round.

Final arbitration of any rule disputes before the day of the contest (Feb 7) will be decided by the Contest Organizers — Fred Martin, Pankaj Oberoi, and Randy Sargent.

All machines that appear to be a safety hazards will be disqualified from the competition.

Contestants may approach the organizers listed above in privacy with questions about possible design that may be questioned under the existing rules. The designs will not be divulged to any of the other contestants.



6.270: The LEGO Robot Design Competition

"ROBOPONG"

OBJECT: At the end of a 60 second round have fewer balls on your side than your opponent has on his side.

- Contestants begin in diagonally opposite circles marked on the table.
- Two Contestants and 15 balls
- 6 balls on each side of the table and 3 balls in the middle plateau
- 4.5 inch high clear plexiglass rim surrounding the playing surface
- The playing surface will divided into a dark region and light region
- The playing surface may not be permanently altered or destroyed
- All evidence of an entry must be removed within 30 seconds after the end of the round
- Robots may not exceed a 1'x1'x1' Max at the beginning of the round

Figure B-2: *Robo-Pong* table specifications

B.3 Robo-Pong, 1991

Object

To have fewer balls on your robot's side of the table at the end of a 60 second round.

Balls

- The balls will be practice golf balls. These are similar in weight to ping pong balls, but are slightly larger and have less bounce. They will be painted.
- There will be 15 balls on the playing surface at the start of a round which will start at predetermined locations. Three balls will start at the top plateau. Each side of the table will have 6 evenly spaced balls in the player's flat area. There will be small depressions in the playing surface to hold the balls at the beginning of a round.

- The balls are inert and all have identical properties.
- Robots may gather balls “into” their body.
- The balls may not be altered or destroyed in any way.
- If a ball is removed from the playing space, then the point at which it leaves the space (crosses the rim) will determine its permanent position on the playing field. E.g.: if your robot pushes a ball off the opponent’s side of the table, it counts as being permanently on that side of the table (for that round).

Period of Play

- The powered portion of a round will last 60 seconds: After the machines are started, they will have 60 seconds to apply battery power to their actuators (defined as motors and solenoids).
- The round ends when all machines and balls come to rest.
- The round will be started by the judges turning on the starting lights, located underneath the table in the center of each robot’s starting circle, for the beginning 5 seconds.
- The contestants will place the machines on the playing field at a random angle with repect to the center per judges’ instructions. The angle at which the machines will be placed will be the same for all machines during the same round, but will vary between rounds.
- The contestants will have 30 seconds to place their machines on the field from the time the judges call them.
- The contestants must stand a given distance away from the playing field. Any contestant that makes an attempt to touch their machine during the round of play will automatically be disqualified from the round.
- The machines must have their own internal clock (software will be provided to do this) which cuts off power to the motors at the end of 60 seconds. Any machines that continue to supply actuator power after 60 seconds will be disqualified.
- The contest will be an double elimination competition held over two nights. Machines must qualify for the second night of competition, as follows:
  1. **Night 1.** All machines will play one round. If a machine loses its round against an opponent, it will run against an inert placebo. If it cannot win against the inert placebo after two tries, it will not qualify for the second night of play.
  2. **Night 2.** The main competition. Machines will play until they lose twice. Loss against opponent from the first night is included.
- All rounds will have two robot players.

- Machines are not allowed to destroy their opponent's microprocessor board.
- Machines cannot try to destroy other machines' broadcast or detection beacons.

Control

- All entries must be solely controlled by their onboard computer. There can be no human intervention once the round begins.
- All entries must be capable of broadcasting infrared (IR) light at two specified frequencies (to be determined). This IR light beacon will be used to detect other machines. Machines failing to meet this specification, or in any way modifying their transmission frequency during the round of play, will be disqualified.
- Judges will assign frequencies for IR emitters to the machines in the beginning of each round.
- After the start of the contest, there can be no change to the robot's program or configuration switches made by the contestants.
- Parts that are likely to damage the internals of machines (ie. small pieces of wire, barb, or fluids) may not be dumped on the playing surface.
- Any LEGO parts that are deliberately dumped on the playing surface (e.g., as an obstacle) will become the property of the 6.270 course.

Structure

- All entrants will be given an equal amount of LEGO and other supplies including:

  1. An ample supply of LEGO, including beams, axles, gears, wheels, and other parts from the LEGO Technics line.
  2. Electronic hardware and plans for building a robot controller board, which can be programmed through any standard serial line.
  3. Motors.
  4. Sensors, including but not limited to IR detectors, touch sensors, photocells, and level-sensing mercury switches.
  5. String, rubberbands, wire, solder, and glue.

  The package will be valued at approximately $500.

- The IR broadcasting beacon must be located at exactly one foot (12 inches) above the surface of the playing field when mounted on the robot. The beacon must be mounted on the largest (dimensionally) part of the machine that traverses across the playing field (if the robot traverses at all).

- Only LEGO parts and connectors may be used as robot structure. LEGO rubber bands are counted as LEGO parts; therefore, LEGO rubber bands *may* be used to provide structural support to your machine.

- LEGO pieces may not be glued together.

- LEGO pieces may not be altered in any way, with the following exceptions:

  1. The grey LEGO baseplate may be altered freely.
  2. LEGO pieces may be modified to facilitate the mounting of sensors and actuators.
  3. LEGO pieces may be modified to perform a function directly related to the operation of a sensor.

- A non-LEGO part may be attached to at most 5 LEGO parts via glue.

- Wire may only be used for electrical purposes, and not structural.

- Rubber bands may be glued to LEGO wheels or gears to increase the coefficient of friction.

- Only the LEGO rubber bands and thin rubber bands may be used to provide stored energy.

- Contestants may not alter the structure of their entry once the contest has begun, but may repair broken components between rounds if time permits.

- The dimension of the machine may not exceed an imaginary 1 foot cube at the start of each round. The IR transmitting and receiving beacons are exempted from this rule. Entries may however expand once the round has begun.

- Entries may not drag wires between two or more structurally separate parts of their robot, unless those wires are part of a LEGO chain link between the various parts of the robot.

- Entries must be built completely from kit parts, with the following exception: Contestants may spend up to $10 for the purchase of up to 10 electronic components used in their design. No single part may cost more than $2. Contestants must show receipts upon request.

- No lubricants may be used, at all.

- For orienting the machines, an arrow pointing to the "front" of the machine must be placed at a visible location for the judges.

Scoring

- The winner(s) will be the machine with fewer balls on its side at the end of the round.

- A clear division between the two sides will be noted by having the surface of the two sides be painted in contrasting colors. Robots will be "told" which side they are starting on by the setting a DIP switch before the round begins. Dynamically, robots will be able to determine which side they are on by using reflectivity sensors aimed toward the playing surface.

- In rounds containing a placebo, the contestants' robot must push at least one ball over to the placebo's side in order to be declared the winner of that round.

- The judges will decide any discrepancies in the contest play.

- Final arbitration of any rule disputes before the day of the contest (February 5th) will be decided by the contest organizers—Fred Martin, Pankaj Oberoi, and Randy Sargent. All machines that appear to be a safety hazards will be disqualified from the competition. Contestants may approach the organizers in privacy to consult about possible designs that may be questionable under the rules listed above. These designs will not be divulged to any of the other contestants.

B.4 Robo-Cup, 1992

This year's contest is "Robo-Cup Soccer," a game played by two autonomous robots.

Object

Have your robot place more balls into its goal within a 45 second period.

Balls

- The balls will be practice golf balls. These are plastic balls, similar in weight to ping-pong balls, but slightly larger and with less bounce.
- One point will be awarded for each ball in your robot's goal when the contest ends. It does not matter which robot caused a ball to enter your robot's goal.
- The balls will be dispensed six inches away from the wall at designated drop points.

  The balls will be dispensed when the corresponding touch sensor has been activated. The dispenser will allow one ball to be dispensed once every five seconds.
- You may place one ball into your robot before the start of the contest round.
- The balls are inert and all have identical properties.
- Robots may gather balls "into" their body.
- The balls may not be altered or destroyed in any way.

The Goal

- Your robot's goal is defined as the goal which is furthest from the robot's starting position.
- The goal is one foot wide and eight inches high with a barrier post that is one inch wide at a height of three inches above the surface.

  See Figure B-4 for an illustration of the goal.
- Your robot may not advertently place anything inside either of the goals except the balls.
- The surface inside of the goal will be sloped so that the balls will roll into the goal when they cross the goal line.

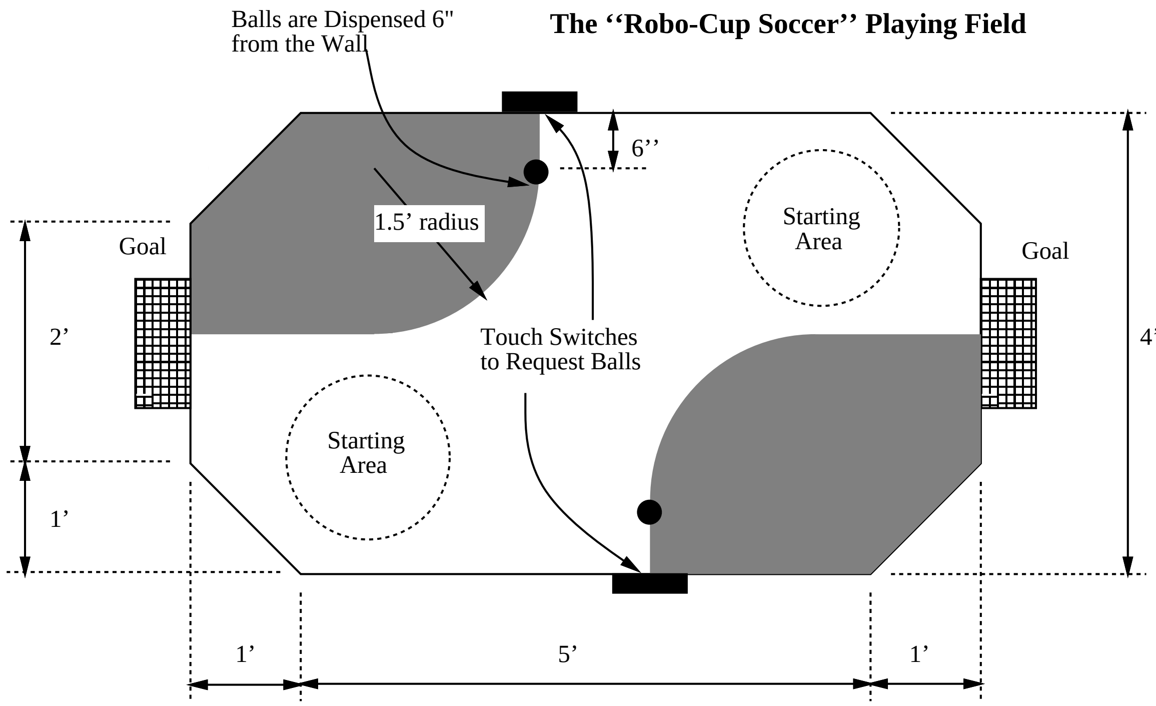

The ''Robo-Cup Soccer'' Playing Field
Balls are Dispensed 6" from the Wall
6''
1.5' radius
Starting Area
Goal
Goal
Touch Switches to Request Balls
2'
4'
Starting Area
1'
1'
5'
1'

Figure B-3: *Robo-Cup* contest playing field specification

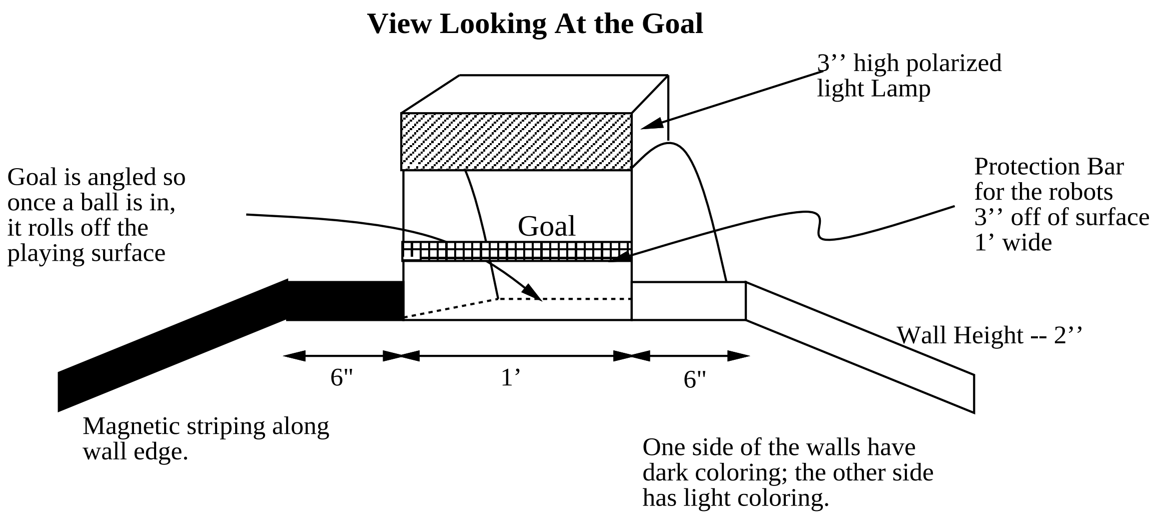

View Looking At the Goal
3'' high polarized light Lamp
Goal is angled so once a ball is in, it rolls off the playing surface
Goal
Protection Bar for the robots 3'' off of surface 1' wide
Wall Height -- 2''
6"
1'
6"
Magnetic striping along wall edge.
One side of the walls have dark coloring; the other side has light coloring.

Figure B-4: *Robo-Cup* contest goal specification

Period of Play

- The powered portion of a round will last 45 seconds. After the machines are started, they will have 45 seconds to apply battery power to their actuators.
- The round ends when all machines and balls come to rest.
- The round will be started by the judges turning on the starting lights, located underneath the table in the center of each robot's starting circle, for the first one second of the round.
- The contestants will place the machines on the playing field within the designated starting circles. The robots may be placed in any orientation within the starting area.
- The contestants will have 30 seconds to place their machines on the field from the time the judges call them.
- The contestants must stand a given distance away from the playing field. Any contestant who touches their machine during the round of play will automatically be disqualified their robot from the round.
- The machines must have their own internal clock (software will be provided to do this) that cuts off power to the motors at the end of 45 seconds. Any machines that continue to supply actuator power after 45 seconds will be disqualified.
- The contest will be an double elimination competition held over two nights. Machines must qualify for the second night of competition, as follows:
  - *Night 1.* All machines will play one round. If a machine loses its round against an opponent, it will run against an inert placebo. If it cannot win against the inert placebo after two tries, it will not qualify for the second night of play.
  - *Night 2.* The main competition. Machines will play until they lose twice. Loss against opponent from the first night is included.
- All rounds will have two robot players.

Control

- All entries must be solely controlled by their onboard computer. There can be no human intervention once the round begins.
- After the start of the contest, there can be no change to the robot's program or configuration switches made by the contestants.
- No parts or substances may be deliberately dumped, deposited, or otherwise left to remain on the playing surface. A machine that appears to have be designed to perform such a function will be disqualified.
- Machines are not allowed to destroy their opponent's microprocessor board.

- Machines cannot try to destroy other machines' broadcast or detection beacons.

Infrared Beacon

All robots are required to carry an infrared transmitter. This transmitters acts as a beacon so that robots can locate each other on the playing field. The following rules describe the functionality of the infrared beacon.

- All entries must carry an infrared beacon that is capable of broadcasting infrared (IR) light at modulated at either 100 Hertz or 125 Hertz with a 40,000 Hertz carrier (hardware is provided to do this).

- Machines failing to meet the infrared transmission specification, or in any way modifying or jamming their transmission frequency during the round of play will be disqualified.

- Judges will assign frequencies for IR emitters to the machines in the beginning of each round by setting the robot's DIP switch 1. If the switch is one, the robot must broadcast 100 Hertz infrared light. If the switch is zero, the robot must broadcast 125 Hertz infrared light. Software will be provided to do this.

- The IR broadcasting beacon must be located at exactly one foot (12 inches) above the surface of the playing field when mounted on the robot.

- The beacon must be located so that its center is never more than four inches (measured horizontally) from the geometric center of the microprocessor board.

- The beacon cannot be deliberately obstructed, or be designed in such a way that "accidental" obstructions are probable.

The Contest Playing Table

Figure B-3 illustrates the Robo-Cup playing field.

Polarized Light Goal Lamps

- Goals will have panels that emit polarized light at one of two orientations: a $+45$ degree and -45 degree (with respect to the vertical) polarization (see Figure B-4).

- The setting of DIP switch 1 will indicate which polarization angle is emitted by the robot's goal. If the switch is one, the robot's goal will be the one with the positive polarization. If the switch is zero, the robot's goal will the the one with negative polarization.

Wall Striping

- The two sides of the playing field will be coded with light or dark visible striping on the lower inside edge of the playing field wall.
- All wall edges will also be coded with magnetic strips.

Ball Dispensers

- The ball dispensers will drop balls at a distance of 15 inches above the designated drop points.
- Robots may obtain balls from either ball dispenser.
- Ball dispensers will dispense balls at a rate not greater than one ball per five seconds.
- Depressing the touch bumper near to the ball dispenser causes a ball to be dispensed as soon as is possible given the dispensing rate mentioned above.
- The touch bumper must be released before a subsequent ball can be dispensed.

Structure

- All kits contain exactly the same components, with the exception that some LEGO parts may be colored differently in different kits.
- Only LEGO parts and connectors may be used as robot structure. LEGO rubber bands are counted as LEGO parts; therefore, LEGO rubber bands *may* be used to provide structural support to your machine.
- LEGO pieces may not be glued together.
- LEGO pieces may not be altered in any way, with the following exceptions:

  1. The grey LEGO baseplate may be altered freely.
  2. LEGO pieces may be modified to facilitate the mounting of sensors and actuators.
  3. LEGO pieces may be modified to perform a function directly related to the operation of a sensor. An example: Holes may be drilled into a LEGO wheel to help make an optical shaft encoder.

- String may not be used for structural purposes.
- The wooden dowel may be used only as a tower to mount the infrared transmitters and any receivers.
- A non-LEGO part may be attached to at most five LEGO parts via glue.

- Cardboard, other paper products, and tape may be used for the purpose of creating optical shields for light sensors.
- Wire may only be used for electrical purposes, and not structural.
- Rubber bands may be glued to LEGO wheels or gears to increase the coefficient of friction.
- Only the LEGO rubber bands and thin rubber bands may be used to provide stored energy.
- Contestants may not alter the structure of their entry once the contest has begun, but may repair broken components between rounds if time permits.
- The dimension of the machine may not exceed an imaginary 1 foot cube at the start of each round. Only the IR transmitting and receiving beacons and the bend sensors may protrude outside this volume. Entries may however expand once the round has begun.
- Entries may not drag wires between two or more structurally separate parts of their robot.

  One portion of the robot is considered structurally separate from another if when the machine is lifted from a supporting surface and held from the other portion, the two portions are supported mainly by wire.
- No lubricants may be used.
- Cable ties may not be used for structural purposes.
- Some parts in the 6.270 kit are considered tools and may not be used on the robot. Examples are: the red plastic parts container; the small rectangular parts container; the soldering iron sponge. If there is any question about whether an object is a "kit part" or a "tool part," ask the organizers.
- Any machine that appears to be a safety hazard will be disqualified from the competition.

The $10 Electronics Rule

To encourage creativity, contestants may spend up to $10 of their own funds for the purchase of additional electronic components used in their design. Other than this rule, robots must be designed completely from standard kit parts. The following conditions apply to all non-kit-standard electronic additions:

- The following components, categories of components, or varieties of circuitry are *disallowed:*

  – Batteries of any variety.

  - Motor driver circuitry, including relays, power transistors, or any other replacements or modifications to the standard motor driver circuitry.

- No single part may cost more than \$2.
- Resistors rated less than 1 watt and capacitors valued less than 100 μF may be used freely, without accounting toward the \$10 total.
- Contestants who add *any* non-kit parts to their project must turn in a design report that includes: description of the modification, schematic of all added circuitry, and store receipts for parts purchases. This design report must be turned in to the organizers by 5:00 pm, Friday, January 31, 1992. *Any machines found with added circuitry that has not been documented in this fashion will be disqualified.*
- If a contestant wishes to use an electronic part which has been obtained through other means than retail purchase, an equivalent cost value to the part will be assigned by the organizers. Contestants must obtain this cost estimate *in writing* from the organizers and include it in the design report mentioned above.

Scoring

- Each ball entering the upper goal area will be awarded three points. Each ball entering the lower goal area will be awarded two points. The winner will be the machine with points at the end of the round.
- In rounds containing a placebo, the contestant's robot must score at least one ball into its goal in order to be declared the winner of that round.
- If no goals are scored a double loss will be awarded by the judges.
- If there is a tie at the end of the round, the win will be determined as the robot that has more balls on the half of the playing field nearer to its goal.
- A double win may be awarded at the judges' discretion.
- The judges will decide any discrepancies in the contest play.

Organizers

- Contestants may approach the organizers in privacy to consult about possible designs that may be questionable under the rules listed above. These designs will not be divulged to any of the other contestants.
- Final arbitration of any rule disputes before the day of the contest (February 3rd) will be decided by the contest organizers—Fred Martin, Pankaj Oberoi, and Randy Sargent.

Appendix C

Administrative Considerations

This appendix is included for readers who may be considering setting up their own versions of the design course presented in this dissertation. I discuss some of the concerns that we faced when we pursued the possibility of offering formal course credit to students who participated in the Robot Design project. While we wanted to offer students formal recognition of the work they were doing in the project, we did not want it to adversely affect the quality of their participation.

Additionally, the issues of kit ownership, student costs, lotteries, and participation by non-MIT members of the academic community are discussed.

C.1 Granting Academic Credit

When the Robot Design project was first getting started, participation was on a non-credit basis, as was the case with most activities in MIT's Independent Activities Period (IAP), the academic session in which the Robot Design project existed. In *Robo-Pong*, participants in the Robot Design project had the option of registering for three units of MIT credit (full term MIT classes generally are worth nine or twelve units). Beginning with *Robo-Cup*, we negotiated six units of credit. This credit was offered on pass/no-grade basis, meaning that students would either receive an ungraded mark of "pass" if they successfully completed the course, and there would be no record of their having registered if for some reason there was a problem meeting the course requirements, or if they simply changed their minds.

Simply put, the rationale for awarding credit for participation in the Robot Design project was that students were doing a lot of work and they may as well get credit for it. Two faculty members of the Electrical Engineering and Computer Science department, Professor Leonard Gould and Professor and Department Head Paul Penfield, evaluated our description the activities performed by students in the class to determine they were worthy of academic credit, decided that they were, and then worked with us to develop evaluation criteria to make sure that credit was fairly awarded.

We did not want to subject students to the performance pressures of the traditional academic term; to the contrary, we wanted to ensure that there wouldn't be any additional pressure on students to get their robots working other than that which they put on themselves. Since the credit was ungraded, the idea of "working for an A" did not exist; we simply had to determine that students had made a reasonable effort and they would get credit. Since students could bail out at any time without penalty, students would not have to make a commitment to something before they knew they were interested in it.

This matter of academic credit was an issue in which the three of us who were organizers had significant differences of opinion. In my interpretation, Pankaj Oberoi was the most "establishment oriented" of the three of us, seeing the least difficulty with offering credit, while Randy Sargent was the most "anti-establishment." While Sargent understood the value of the MIT administration recognizing the project as a credit-worthy activity, he also saw most strongly the potential for harm if students were to feel unnecessary pressure because of it. We all knew that we would have to safeguard against a student who tried to take a "free ride" with their teammates, but Sargent and I were additionally concerned with how strongly we should attach the awarding of academic credit to general participation in the project. For example, should students be *required* to take the course for credit, or should it be an option? Sargent and I would not have been supportive if we had decided on mandatory credit, but even after we agreed that credit should be an optional feature of participation, the question remained of how actively we should promote it. I thought that as long as we made clear to participants that (1) credit was optional and (2) they could remove themselves from the credit list at any time without penalty, it was appropriate to advertise that credit was available, since it would make the project's standing in the eyes of the faculty more solid. Sargent agreed to this, but I believe he continued to be concerned about more subtle ways that credit would affect students' attitudes and participation in the course.

As it turned out, the necessities of tracking students individually for purposes of ensuring that credit was awarded fairly dovetailed nicely with my interests in taking data on students' work for the purpose of this doctoral research.

In developing the requirements for credit registrants, we wanted to achieve a fair balance between the amount of information collected and the additional burden it would place on the participants. If something were to require an inordinate amount of time from the students to provide to us, then we would not make it a requirement; we wanted the additional work required to receive credit to be as little as possible.

After discussion, we agreed to the following requirements:

Weekly Written Reports. One of our concerns was having some basis for distinguishing who was doing what on a given team of robot builders. By asking students to prepare a series of three written reports, which focused on their own thinking and work on the design project, we would have an objective way of ensuring that each individual on a team was doing his or her appropriate share.

We offered students the option of keeping a daily journal log of their work, which they could photocopy and submit in lieu of a prepared report. Only a few students chose this option; we didn't want to make the journal format mandatory in that we did not believe it was a preferred working style for the vast majority of the class.

Weekly Video Reports. As an additional vehicle for collecting data, we set up a video camera in a spare room of the electronics lab and required that students make a weekly presentation in front of the video camera. The purpose here was to capture a dynamic expression of team interactions that wouldn't be possible with the written reports; all members on a given team that were registered for credit were required to participate.[1]

These two items were the primary means by which we determined that students were in fact doing work—at least for the ones that we hadn't gotten to know simply by the extent

[1]This concept was originally suggested by Mitchel Resnick.

and visibility of their general participation. We also established the following two criteria, which did not require any additional work on the students' part:

Completed Robot. We figured it was reasonable to ask that teams registered for credit complete their design project to the extent that they produced an tangible artifact which they could point to and say, “This is the robot.” We did not require that it solve the contest in order for students to receive credit.

Program Listing. As a record of their work, students were required to submit copies of the program they had written to control their robot. (One of the wonderful things about software is the simplicity by which copies of the work may be made; I wish it might be so easy to ask for so complete a recording of the robot design itself!)

Even as we developed a structure to allow students to receive academic credit for participation, we were extrememly careful to make sure their motivations for their participation weren't affected by traditional academic ambitions. For this reason we did not wish to give out letter grades; we also wanted to make it the case that a variety of styles of participation would still be encouraged under the evaluation criteria for awarding credit.

C.2 Student Costs

From the start of the project, we were concerned that financial considerations should not cause any student to hesitate to participate in our project. The first year that hardware was involved, we chose to collect $50 from each team to defray the costs of buying the parts that would comprise the robot-building kit. We estimated that this figure would be small enough, when split among the two to four team members, to be negligible in any student's financial planning. That year Microsoft Corp. was the patron sponsor of our project, providing over $1000 of cash support.

In the subsequent year we attempted to raise cash and part donations from numerous corporations in our commitment to keep student costs to a minimum. We also solicited substantial cash support from MIT's Electrical Engineering and Computer Science (EECS) department. As it turned out, Motorola Semiconductor donated expensive components that were instrumental in the 1990 project year, Microsoft continued as a substantial cash sponsor, and MIT became the major underwriter of the project costs.

Over the subsequent years of the contest we were fortunate enough have the continued commitment of our initial sponsors while recruiting about a dozen others (who contributed primarily through part donations). We also had the support of Professor Gould in our commitment to keep student costs down; the kit fee has remained at $50 per team since the start of the project.

C.3 Kit Ownership

The fact that students kept their kit at the end of the course had a positive motivational effect. One of MIT's other project-oriented courses, *Digital Design Laboratory*, requires that students disassemble their projects essentially immediately after they have finally demonstrated them to work. After spending six weeks on a project and a final intensive period of several days, it is a painful event to have to rip the wires out of one's project in a furious burst just after the project's completed. By allowing students to keep their robots, we encouraged them to work further on them as time allowed. Each year a number of students got their robots in functioning order for a re-match contest that we held a few months later at Boston's Computer Museum.

The low entry fee combined with the kit ownership policy caused an unforeseen complication, however, as the degree of subsidy increased each year. By the 1991 project year, we were providing students with a robot-building kit that had a retail value of about $500. The question of who keeps the robot or how to split up the kit after the class became nettlesome in a number of cases.

Sometimes these disputes took on a sad although comical dimension, resembling bitter child custody battles that ensue when marriages fail. One student pays for the whole kit, but becomes less involved in the project than his partner. Both feel they have a valid claim to keeping the kit afterward. The student who paid the course entry fee believes he bought the kit; the other argues that the entry fee is only a fraction of the kit value, so it should not be the determining factor. After witnessing a number of these situations occur, we made a

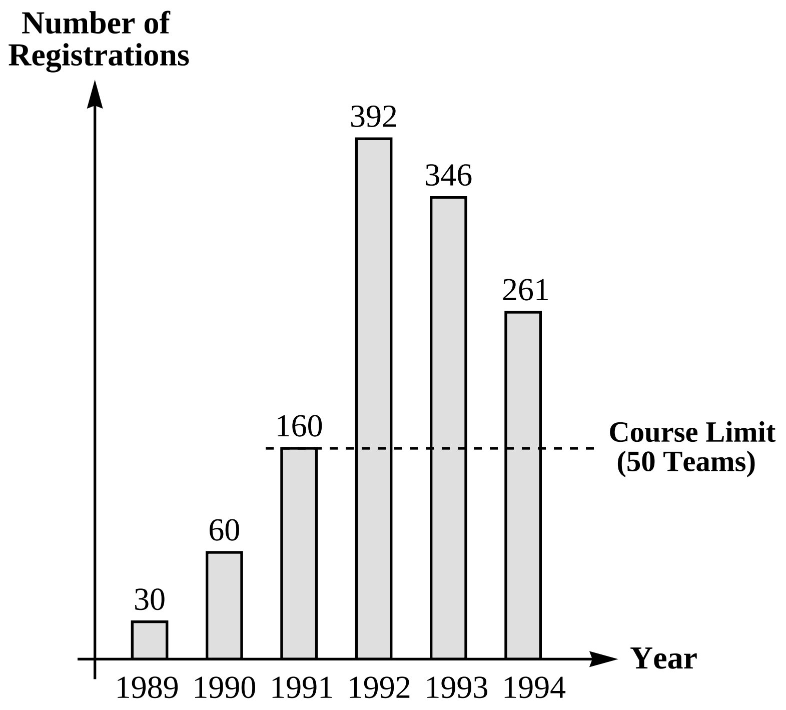


Figure C-1: Number of registrants by year, 1989 through 1994

point of having students talk the issue through before the class got underway, and come to some kind of agreement about how the matter would be resolved beforehand. This seemed to ameliorate the problem.

C.4 Lotteries

Another problematic matter arose from the success of the class, beginning in the 1991 contest year, when more students signed up for the class than resources would permit. For several years, the registration for the class was on an exponential growth curve: 30 students in 1989, 80 in 1990, 160 in 1991, and 392 in 1992 (see Figure C-1). (Since the peak in 1992, the registration count fell to 346 in 1993 and 261 in 1994.) For the first three of these years, we expanded the class to accept all interested persons. But after the 1991 year, we knew that we did not have the resources to support more than fifty teams of students. It wasn't a question of monetary resources, but that of personnel to run the class; we felt stretched very thin by the 160 student class.

After 1991, when we were able to accept all of the registered students, we knew we would have to come up with a mechanism for dealing with oversubscription. We determined that a simple lottery was the most appropriate way to deal with the matter. This would avoid having to decide on individuals on a case-by-case basis and subject our project to the unseemly appearance of letting people into the class because of personal relationships.

In the years of 1992 and 1993, all regstrants had equal weight in the lottery. The organizers in 1994 decided that it would be appropriate to give students who had lost previous lotteries a higher chance of getting in, and a complex weighting system was established in which a person's chance of getting into the class depended on the weighted average of previous lottery losses across a team entry. Still, there were cases in which persons who were graduating seniors and had lost prior lotteries were denied admission. It is clear that the lottery system is not ideal, though there continues to be general agreement amongst the organizers that it is better than trying to resolve cases individually.

C.5 Non-MIT Participation

By the 1993 contest year, knowledge and excitement about the Robot Design project had spread to students at Harvard University and Wellesley College. A number of students from each of these institutions inquired as to whether they might register for the class.

The organizers at the time made the decision that non-MIT students could register, but they would only be accepted into the class if there was not enough demand from the MIT community. Effectively, this was equivalent to denying them admission; there was and still is no foreseeable end to the oversubscription problem.

I was no longer an active organizer of the class at this time and had little effect on the discussion of this matter. While I can see the reasoning behind the organizers' decision, I believe it was parochial. It is painful to deny any person admission to the class, and it would seem unfair to allow a non-MIT student admission when so many are denied it. Harvard and Wellesley, however, are institutions that enjoy a particularly close relationship with MIT. Students from all three institutions take classes at the others. It would have been more farsighted to allocate one to two team slots for students from these schools.

Appendix D

Technology Development

This appendix chronicles the development of the technology for the Robot Design project, providing additional technical details and conceptual motivations behind the three stages of technology that were developed: the Remote Controller (1989), the Assembly Language Controller (1990), and the C Language Controller (1991 to present).

D.1 The Remote Controller

In the first year that we used real electronics in the course, parts cost and development time were extreme constraints. We were aiming for materials for ten to twelve teams of participants that should cost between one and two thousand dollars total (we had gotten a commitment of $1000 as a sponsoring donation from the Microsoft Corporation, and we anticipated raising additional money through MIT sources and registration fees from students taking the class). We had just about one month to design a system and procure ten to twelve sets of materials for the preregistered participants.

We first agreed on what was a necessary core of a hands-on robotics experience. We determined that it would have to include sensors, motors, and programming. So our minimal system would have to allow project participants to build a machine that incorporated motor control, sensor input, and programming to tie it together.

Randy Sargent hit upon the idea of using an inexpensive communications chip (a universal asynchronous receiver/transmitter, or UART). Using the UART chip, we designed a hand-wired board that added a couple of motor driver chips I was familar with, having used them in the Programmable Brick. Our solution thus sacrificed on-board computation, allowing us to forgo implementing a full-blown microprocessor design and use the computational power of desktop machines instead. This approach afforded us a solution that was feasible within the extreme cost and time constraints that were present.

D.1.1 Hardware

Since little could be done without it, the first task we gave the participants of our project was the one of building their own copy of the controller board. Each board had to be individually hand-wired on blank perfboard panels; we created a chip layout diagram and wire list so that each team (we organized the participants into teams of one to three persons) would build the same circuit. Many of the participants had never soldered before, making the board assembly process fairly difficult. Our troubles in assisting the students were compounded when we realized that about 20% of the blank perfboards were numbered with slight differences from the one we had used to formulate the wirelist.

Figure D-1 shows a photograph of the completed board, along with a couple of accessory boards. The D-shaped connector along one edge of the board provided the connection to the off-board computer and power supply. The board was in principle capable of driving up to four motors independently, but the power supplies that we purchased were underpowered for dealing with the motors we had acquired, making sufficient power available for driving only two motors simultaneously.

For sensors, the board was able to accept data from up to eight digital ("on" or "off") sensors. The primary sensor to be used in the contest would be a rolling ball switch device that would allow robots to detect their inclination on the playing surface. This sensor was easy to interface to the board, though other unforeseen complications arose with its use (vibrations made it difficult to obtain readings from the device).

Though it was not strictly necessary in solving the contest challenge, we hoped that the participants in the class would want to use analog light sensors. We anticipated two possible uses for these. The first was in detecting the light bulb that was embedded in the peak of the "mountain"—a convenient way for the robot to determine that it had reached

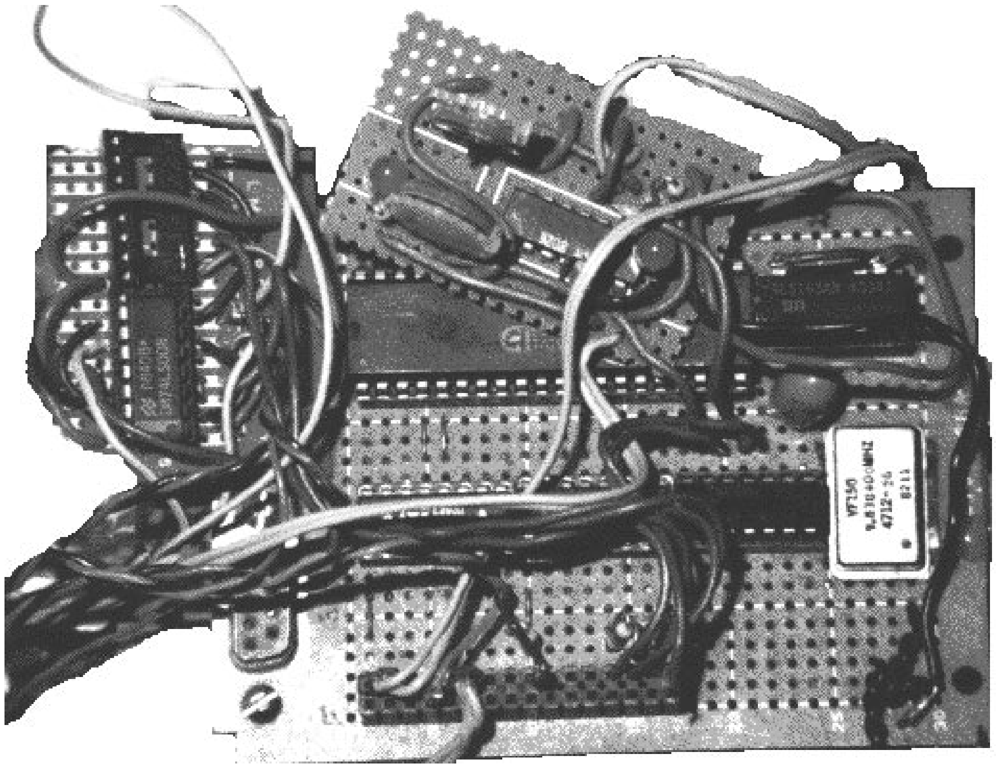


Figure D-1: The 1989 Remote Controller Board

the top of the hill. The other possible use was in detecting the two opponent robots: it was mandated that robots carry illuminated flashlight bulbs, so that other robots might be able to detect their light and thereby their presence, and then react in some useful way.

To facilitate the use of analog light sensors, we provided the participants with the parts and a diagram for building a circuit that would convert the analog signal provided by the light sensors into a digital pulse train that could be accepted by the interface board. The use of this circuit would require special software to time the length of the pulse, which would correspond to the value of the analog sensor (see Figure D-2).

It turned out that not one group of students used the analog sensing capability, though several did experiment with it. Getting basic functionality from their robots (i.e., a robot that could climb the hill) was so fraught with problems that there was no time to add the light sensors to their robots. It was sufficiently difficult that there was no point in doing so.

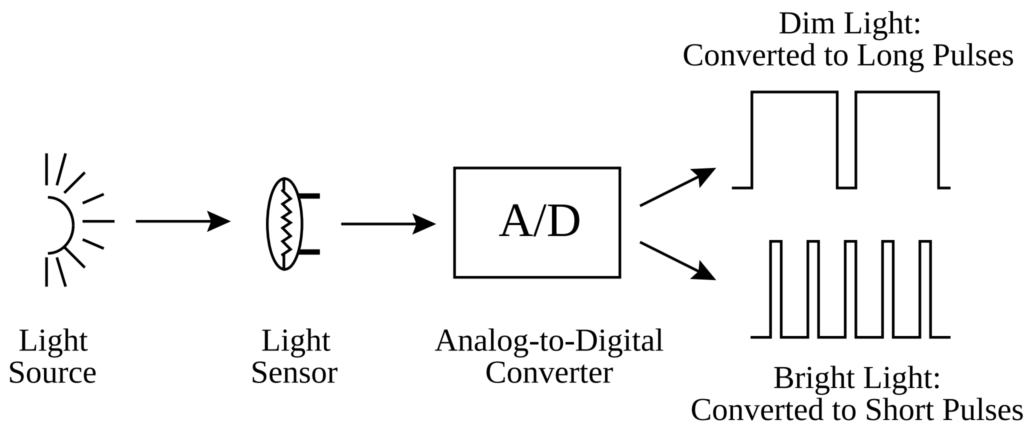


Figure D-2: Converting analog light reading to digital pulse stream

D.1.2 Software

The programming environment that went along with the Serial Line Controller Board consisted of a diverse set of options. We planned on obtaining permission to use campus-wide Project Athena workstations for programming, but had failed to do so.[1] We were thus forced to cobble together a collection of personal computers, consisting mostly of IBM-PC compatibles in various configurations—some with hard disks, some without—and a few Macintosh computers which were provided and used by students who personally owned those machines.

We were unable to provide an officially supported programming language either. Students used whatever language with which they were most comfortable with or to which they could most easily gain access. This turned out to be various versions of the C, Pascal, or BASIC programming languages.

Regardless of the language used, we recommended an approach to programming whereby a main "event loop" would be used. Inside this loop, the control program would sample the state of the sensors and choose a desired action. Through repetitive evaluation of the loop, the robot's behavior would be implemented.

This approach was adequate. The primary challenge that year was simply to get

[1]Project Athena administrators were concerned that damage might result from our hardware being connected to the machines' serial ports. Two years later, we were able to persuade them otherwise, which turned out to be crucial in our plan to provide a consistent software environment for future students.

something working at all. Sophisticated robotic control ideas were not important; basic functionality was. This largely consisted of successfully debugging the controller board and getting communications between the desktop PC and the controller to function properly.

One of the issues we encountered in this first class/contest experience, which would return in future versions, was the interrelatedness of contest and hardware. Our desire and intent was to create and provide interesting materials for the students, but we saw the need to make them easy to use and provide motivation for their use. The failure of the students to use the option for analog circuitry was a case of the technology simply being too difficult to use as well as being insufficiently motivated.

D.2 The Assembly Language Controller

Among the difficulties in using the hand-wired remote controller, the greatest was the fact that it had to be tethered to the desktop computer. Our primary desire in upgrading the hardware used in the project was to create environment in which robots could have on-board control. This was the single most important improvement to the materials that could enhance the excitement of the project and capture the students' imaginations. By providing an autonomous controller, not only would the level of technology would be raised, but the students would become involved in more interesting robotic and design issues.

The design of a controller board for our purposes was made feasible by the availability of a new, highly integrated microprocessor chip that incorporated into a single integrated circuit the functionality of what typically required several chips. This chip, the Motorola 6811, was literally a "single chip computer" that included important features for robotic control applications, such as direct analog input, serial line communications hardware, and an internal programmable memory.[2] Whereas the Programmable Brick design used seven integrated circuits and two printed circuit boards, we realized that we could easily design a far more simple controller using the Motorola 6811 that would be within the budgetary

[2]We owe thanks to Henry Q. Minksy of the MIT Artificial Intelligence Laboratory, who helped us get started using the Motorola 6811 by providing us with a prototype board of his own design, development software, and other technical support.

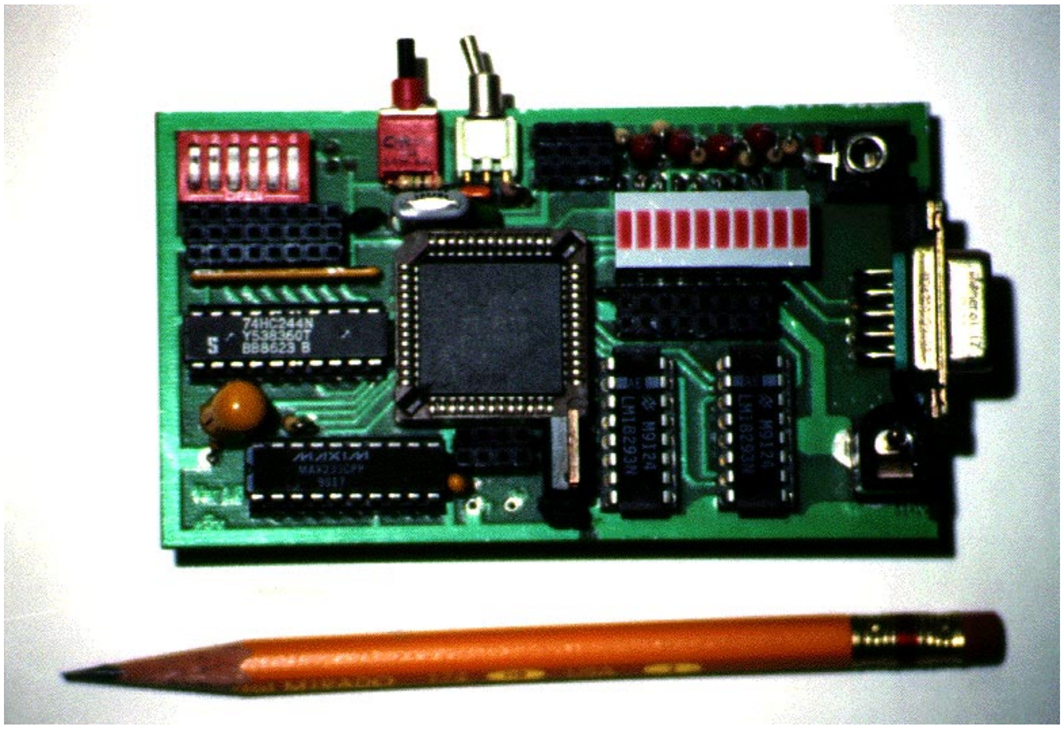


Figure D-3: The 1990 Assembly Language-based Controller Board (actual size)

scope of the Robot Design class.

D.2.1 Hardware

We attempted to make a controller board that would be as easy to use as possible. Here the experience of having worked on the Programmable Brick controller was valuable.

The LEGO/Logo controller interface had status lamps on the motor outputs; when a motor was being driven, the corresponding status lamp would light. The Programmable Brick did not have these. Having worked with children using both systems, it was apparent that the status indicators were a positive factor in helping users understand the relationship between computer commands (e.g., turning on a motor) and tangible results (e.g., seeing the motor turn on). Also, the status indicators were invaluable when debugging: if the user believed a motor should be turning, a glance at the output lamps would tell whether or not the computer "thought" the motor should be on. It made it easy to trace problems that were due to faulty wiring versus other causes.

So we designed motor status lights into the new board. There were a few other features that helped make the board easy to use, including a serial connector port compatible with the IBM personal computer standard, and sensor connectors that allowed both digital and analog sensors to use the same connector style. Figure D-3 shows the controller board we designed for *Robo-Puck*, the 1990 version of the Robot Design project.

D.2.2 Software

For programming the 6811 chip, we obtained the standard software products used by embedded control applications engineers. Using these software tools was a tedious process, even for an experienced programmer. First, the target program—the program to be run on the 6811 microprocessor—would be composed on a text editor. This program was written in assembly language, a code that uses word-like mnemonics to represent the primitive instructions of the computer chip. This program was then translated into a binary object code form with the use of an assembler program; syntax errors and other coding mistakes would generate errors at this point that would need to be resolved. After this sort of error was fixed, the resulting object code file could be downloaded to the microprocessor (using a downloader program). Finally, the target program could be run on the robot.

There were a couple of serious troubles with this sort of system. The first problem was that the system provided little interactivity between the computer and the programmer. For example, there is no way for the computer program to display results for the programmer to examine. The only "results" of program execution that are readily available are the state of the motors, which are often the thing being debugged! Second, it is notoriously easy to write an assembly language program that fails absolutely and without warning, due to the nature of the language itself. When such a program crashes, it is difficult to tell where in the code stream the problem lies.

Traditional assembly language development environments address these problems with the use of what is known as a *breakpoint monitor*. This system allows the programmer to single-step through the program execution and set breakpoints where the program is halted so that the programmer can interact with the microprocessor to examine execution status and display other results. Our controller board, however, did not have enough memory to support the traditional breakpoint monitor. So we were stuck with the primitive assembler and downloader because there was not time to develop a better environment. We hoped that the students who participated in our workshop would be able to be successful using the system despite its inherent user unfriendliness.

It was nevertheless apparent that some method for easily interacting with the robot's hardware (i.e., motors and sensors) would be valuable. We realized that a limited sort of monitor program, that focused on providing interaction with the motor and sensor features, could be supported by the capabilities of the controller board.

Following up on the idea, we created a monitor program named *RM-11* (Robot Monitor for the 6811). The program allowed users to type simple keyword commands to turn motors on and off and display sensor values. It was feasible in terms of the memory limitations because it did not support the typical monitor functions of single-stepping and breakpointing. Instead, it was customized for the particular hardware of the controller board and provided a straightforward interface for interacting with the special features of our board.

The assembly language programming environment proved to be quite problematic. Most of the workshop participants didn't get to the programming stage of their projects until there were only a few days before the contest. Many of them had not seen their machine's motors actuated by the controller board by this time, and hence their design ideas were largely untested at a point when there was insufficient time to make substantial revisions.

The conceptual overhead involved in doing the most minimal programming task (e.g., proof-of-concept code that turned on a motor based on a sensor reading) was significant: even a trivial task required the understanding of a variety of mundane programming details. The result was that students didn't attempt programming until all other aspects of their robot design had been completed, at least as far as they believed.

Not only was the overhead in learning to use the assembly language environment considerable, but the interactivity in doing programming was minimal. This led most teams to take top-down strategies toward the programming project. They prepared their program conceptually or on paper, wrote the entire program code, and tried to debug it.

This style of programming was quite problematic from a technical standpoint of getting the thing working. Because of the nature of assembly language programming, errors tend to be of the crash-and-burn variety: the program fails without warning, without providing notice as to where it crashed, how or why. In the case of the robotic hardware being programmed, the problem was made even worse because of the variety and intermingling of possible failure modes, including electrical problems with the computer hardware, software errors, and sensor-related failures.

D.3 The C Language Controller

In analyzing the shortcomings of the assembly language controller system, we perceived two conceptual areas in which it could be changed:

Observability. The assembly language system was difficult to work with and debug because the tools available for observing the state of the system were minimal. In conventional assembly language development environments, the breakpoint monitor is used to observe progress of program execution. This tool is itself difficult to work with and limited, but even it was not available.

In order to make a system that was more user-friendly, that would encourage users to explore, play with, and understand the technology, it would need to provide many more ways of observing the internal state of the robot, including both sensor values and program execution status.

Level of Abstraction. The assembly language system did not insulate users from the low-level details of 6811 programming; in fact, it forced them to deal with various details in order to accomplish their programming. This situation could be seen as either a valuable pedagogical feature, encouraging students to learn about the technology in a thorough way, or an unnecessary conceptual burden on the users, who are simply trying to build interesting robots.

We decided that it would be advantageous to give students a higher level interface to their robot design work, shielding them from details of 6811 programming but offering them the possibility to express more complex ideas. This was a difficult decision, as we felt that the assembly language programming experience could be quite valuable for students, but it was in keeping with the overall philosophy of the project. In giving students a predesigned controller board and predesigned sensors, we were abstracting them away from low-level details of digital and analog electronics; if we provided them with a higher level software interface, it would be continuing an established trend. The value of the giving students a higher level software environment would be evaluated by seeing the sorts of problems in which they became engaged, in comparison to the sorts of problems they encountered in the earlier assembly language environment.

The question of observability was less controversial. It was evident that the assembly language system was quite poor in this regard and providing better tools here would not have any negative consequences. To the contrary, having a system that was more observable would give students a greater sense of power and control over the materials, allowing them to create more complex systems and have greater understanding of their work.

Between the 1990 and 1991 years of the class, Sargent and I performed the development work required to bring about a more sophisticated set of robot development hardware and software. In doing so, we developed a system that had a strong resemblance to the original LEGO/Logo Programmable Brick. The development effort had two parts: the creation of a more powerful controller board and the design of a custom programming environment.

D.3.1 Hardware

As a foundation, we designed a new controller board with significant increases in capability and versatility over the assembly language board design. In order of importance, the changes we made were:

More Memory. The most important change was an increase in on-board memory. The assembly language controller was limited to 2K bytes (2048 bytes) of memory for user programs. The new board had 32K bytes of memory. Not only did the increase allow students to build longer, more complex programs, but it was required to support the new programming environment we created.

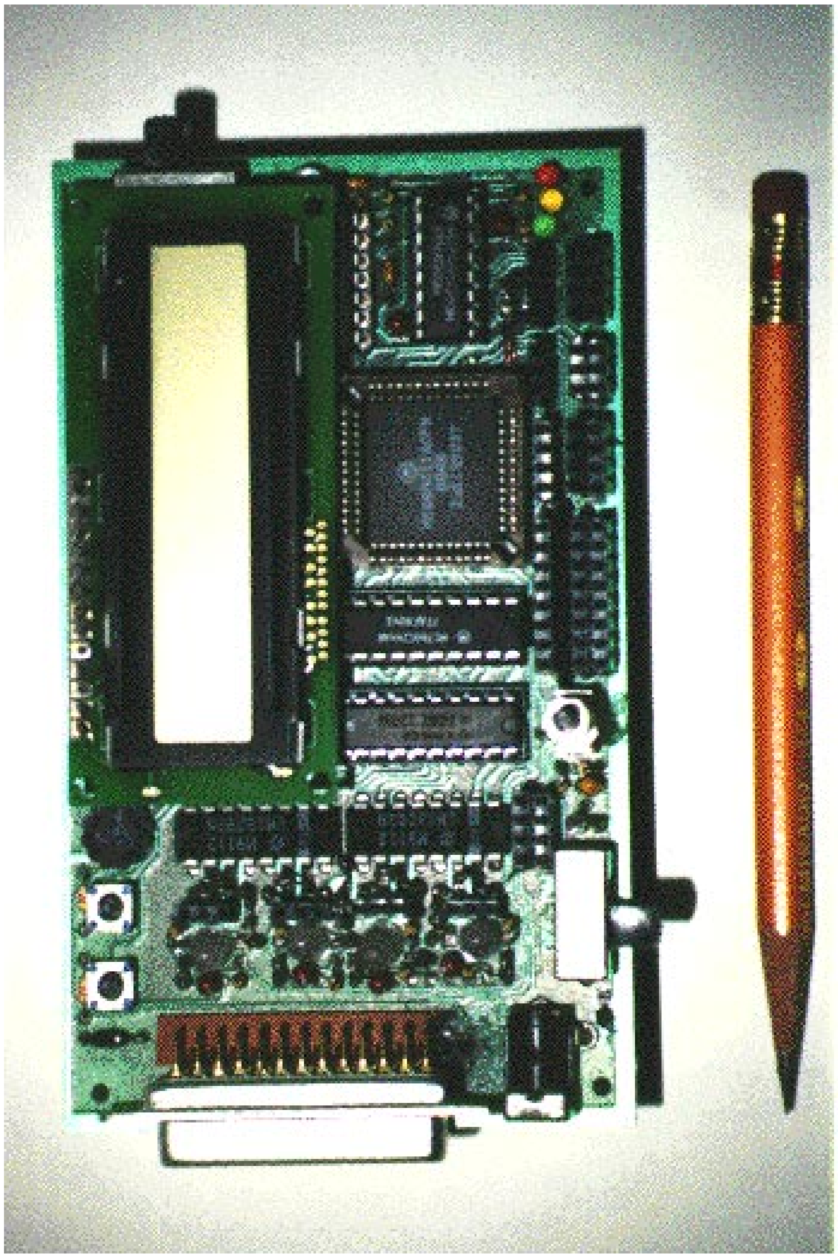


Figure D-4: The 1991 C-language-based Robot Controller Board (actual size)

As with the assembly language board, the memory was *non-volatile*, meaning that the robots would retain their program when switched off. We had found this feature quite valuable; it allowed students to program their robots and test them for extended periods away from a development computer, without needing to constantly reload the program from the computer. This was the feature that was noticeably lacking from the LEGO Programmable Brick.

Expandability. The new board was designed for expandability, allowing students to control more motors, receive data from more sensors, and have a general purpose prototyping space for developing their own circuits. A daughter board that plugged on top of the main board provided these additional features.

LCD Character Display Panel With the assembly language board, there was no manner for the hardware to provide feedback to the students about its internal state. That is to say, the students could program the board to control their robot's motors, but there was no easy way to provide any status information about the state of program execution (other than by observing the state of the motors, which was often not terribly informative).

The new board supported a 15-character LCD display panel. The software we developed included a programming statement to write data to the display (both text strings and numeric output). This vastly changed the "observability" of program execution: it became easy to add a print statement to a program and thereby monitor its internal status.

Also, it became much easier to experiment with the operation of sensors. Students could write a one-line program to repeatedly print the value of a given sensor to the display. The robot no longer needed to be connected to the desktop development computer in order to display results—students could disconnect the robot from the computer, bring it to the contest playing table or other location, and manipulate the robot or conditions in its environment and directly observe changes to sensors' values.

Electronic Beeper An electronic "piezo" beeper was provided with simple routines for making beeps of varying pitch and duration. The assembly language board had included a beeper, but it was difficult to operate from software and was not utilized by students generally. With the new system, we saw an explosion of "musical robots" that used sound output for both entertainment and informational purposes.

User Buttons and Knob. To facilitate interactions with the robot's hardware, we put two pushbuttons and an adjustable knob on the board. This allows users to create menu-based programs using the LCD display. Choices could be selected with the buttons, values could be entered using the knob, and results could be displayed on the LCD for immediate viewing. This allowed for a new sort of interaction with the robot when it was detached from the development computer.

Infrared Capability. Each robot was given the ability to broadcast modulated infrared light. This feature is discussed later in this appendix (Section D.4).

D.3.2 Software

Sargent spearheaded the design and implementation of a custom programming language for use with our new board; I assisted as a sparring partner in the conceptual design of the system and as an implementor of certain subsections of it.

We had to determine which programming language we would implement. We knew that we wanted to implement a language based on the procedural metaphor because this was the most common, proven useful, and generally accepted form of programming. The actual language on which to base our system was not immediately obvious.

We implemented a trial version of the programming environment based on the Logo language. The implementation supported procedures, recursion, and integer variables. We saw Logo as desirable because of its learnability and the simplicity of the Logo syntax.

Other than Logo, the C language, another popular procedural language, was a front-runner in our considerations. We considered the following factors in deciding which language to use:

Data Types and Structures. The C language provides for a richer set of data types than does Logo. For the full version of the language, we wanted to support floating point numbers and arrays. The traditional syntax for the Logo language does not provide for different data types. While Logo's primary data structure (the recursive list) is a superset of the functionality of the C-language array data structure, for implementation reasons it was impractical to support the Logo list data type, leaving us with the choice of non-standard representations in the Logo option.

Control Structures. C provides more types of control structures in its base language specification than does Logo. While various control structures could easily be added to Logo, they would be idiosyncratic modifications rather than standard features.

Reputation. The Logo language is thought of as being a baby language, suited primarily for use by elementary school children. This common perception of Logo is unfortunate and indeed incorrect—Logo is actually an elegant and powerful computer programming language—but the idea of Logo as a kid's language, in the belittling sense of "kids," is widely held.

On the other hand, the C language is generally perceived as being a serious, professional development language, having some learnability problems but otherwise being an indispensible tool for computer professionals.

Learnability. The syntax of C is more obscure than that of Logo; this is definitely a drawback for a novice to the language. However, the two most objectionable properties of traditional C environments—the lack of interactivity in program development and the ease with which it is possible to write a C program that will crash—were factors that we would deal with separately from the choice of language grammar and data types.

We expected that many MIT students would already know how to program in C, while those who did not would not object to having to learn it.

For a combination of these reasons, we chose to use create a version of the C language for our project. However, rather than being constrained by the traditional C model, we implemented some features that would make our system ideal for learners, including *interactivity* and *uncrashability*. (These two features are characteristic of Logo systems but not C systems.) In addition, we added *multitasking* capability (standard to neither C nor Logo systems). So in a sense, the system we created combined the best features of both languages: the ease of use of a Logo system with the expressive power and versatility of the C language.

Interactivity

The biggest problem with the software environment of the assembly language system was its batch-mode metaphor: first you would write a program, then assemble it, then download it, and then see if it worked. In contrast, the Logo Programmable Brick system provided a Command Center, in which users could type commands which would be executed immediately after being typed.

We gave the our new system a feature quite like the Programmable Brick's Command Center—an interactive command line interface. By typing functional calls and compound statements at the command line, users were in interactive control with their robot, and could directly command motors settings and examine sensor values, as well as being able to download and execute procedures.

Uncrashability

One of the difficulties in working in both traditional C and assembly language, as compared to an interpreted language like Logo, is the ease with which it is possible to write programs that fail completely, causing the computer to latch up and becoming unresponsive (i.e., to crash). When compounded with a hardware environment that is unstable, debugging becomes nearly intractable because one cannot be sure if hardware troubles, software mistakes, or both are the cause of a failure. This was perhaps the most frustrating aspect of working with the assembly language board system. If a software environment could be provided that did not allow total failures, then crashes could immediately be attributed to hardware causes, simplifying debugging tremendously.

Our goal for providing an uncrashable environment should not be interpreted to mean that programming errors could never occur, but rather that when they did, the system should respond in a controlled way, informing the user that the program has encountered some sort of error condition. In contrast, typical assembly and some C coding errors result in the computer's latching up, which gives no particular information about what part of the program caused the error to occur.

We designed specific hardware and software features into our system to inform the user about the state of program execution. The main portion of the work was done in the software, which protected users against types of errors that would traditionally cause program crashes. For example, in the conventional specification of arrays in the C language, the program is free to make array references outside of the region for which the array is declared. Put other way, it is the responsibility of the programmer to make sure that the program does not inadvertently violate array bounds. Failure to respect array bounds can result in immediate "segmentation violation" errors on some computer operating systems, or more insidious, delayed failure modes on others.

Our system was designed to report this sort of error to the user immediately, with an error code designed to assist in debugging. The result of our design was that our system would never simply crash unless a hardware condition were to blame; any type of software failure would result in an error code being reported to the user.

To assist users in determining the execution status of their programs, we designed a "system heartbeat" monitor on the LCD screen. When the hardware was functioning normally, one character of the LCD screen would blink between two different states. If the blinking stopped, then the system has crashed.

Multitasking

As a side effect of the overall architecture we used for the Interactive C language system, it was relatively simple to add multitasking capability to the system. The system allowed up to about ten procedures to be interpreted in a time-sliced fashion, giving the illusion that they all are executing simultaneously.

This feature was not motivated from a usability point of view, but rather as a tool to give students a richer way to express their control ideas.

D.3.3 Results

Interactive C and our new hardware system provided a quantum leap in the complexity of robots built by students, and the sophistication of ideas they developed in doing so. It became possible for students to approach the design of their robot through bottom-up rather than top-down means; the implications of this change are discussed in Chapter 5.

The Interactive C software was implemented on the campus network of Unix machines (the Athena network). The existence of an ample supply of these computers, as well as a built-in distribution channel for the software (the campus network) greatly facilitated the use of the software. The campus computer clusters became centers of creative robot development activity; also, since Athena workstations were installed in many dormitories and fraternities, students would work from their living spaces.

The technology has been improved incrementally since the first year of the Interactive C system, but not in substantive ways. Additions have included a larger LCD display, more analog input channels, and outputs to drive a servo motor.

D.4 Sensor Development

From the start of our work on the Robot Design project, we knew that key to creating interesting environments for robots, and hence for the students who were creating them, was the development of versatile, functional sensor devices. Without sensors, a robot is just a machine; robots need sensors to deduce what is happening in their world and to be able to react to changing situations.

A whole industry is built around the development of sensors for use in industrial settings, such as process monitoring. The research robotics community also has developed its own set of well-understood sensor devices. But few of these sensors would be suitable for our project, because of the cost of obtaining and complexity of employing these devices. We needed to devise sensors that would be inexpensive and simple to use, yet provide a valuable function in the students' robot projects.

Several devices were fairly obvious choices. For example, pushbutton or other types of switches can act as tactile (contact) sensors. The cadmium sulfide cell is another commonly available and easy to use device that is used to sense levels of light. The mercury switch is an example of a device that seemed simple at first glance, but turned out to have unexpected complications when put into service. Just as vibrations caused problems with the rolling-ball sensor (see Section 3.3), vibrations of the robot as it moved about caused the bead of mercury to bounce within the glass tube, leading to unreliable readings.

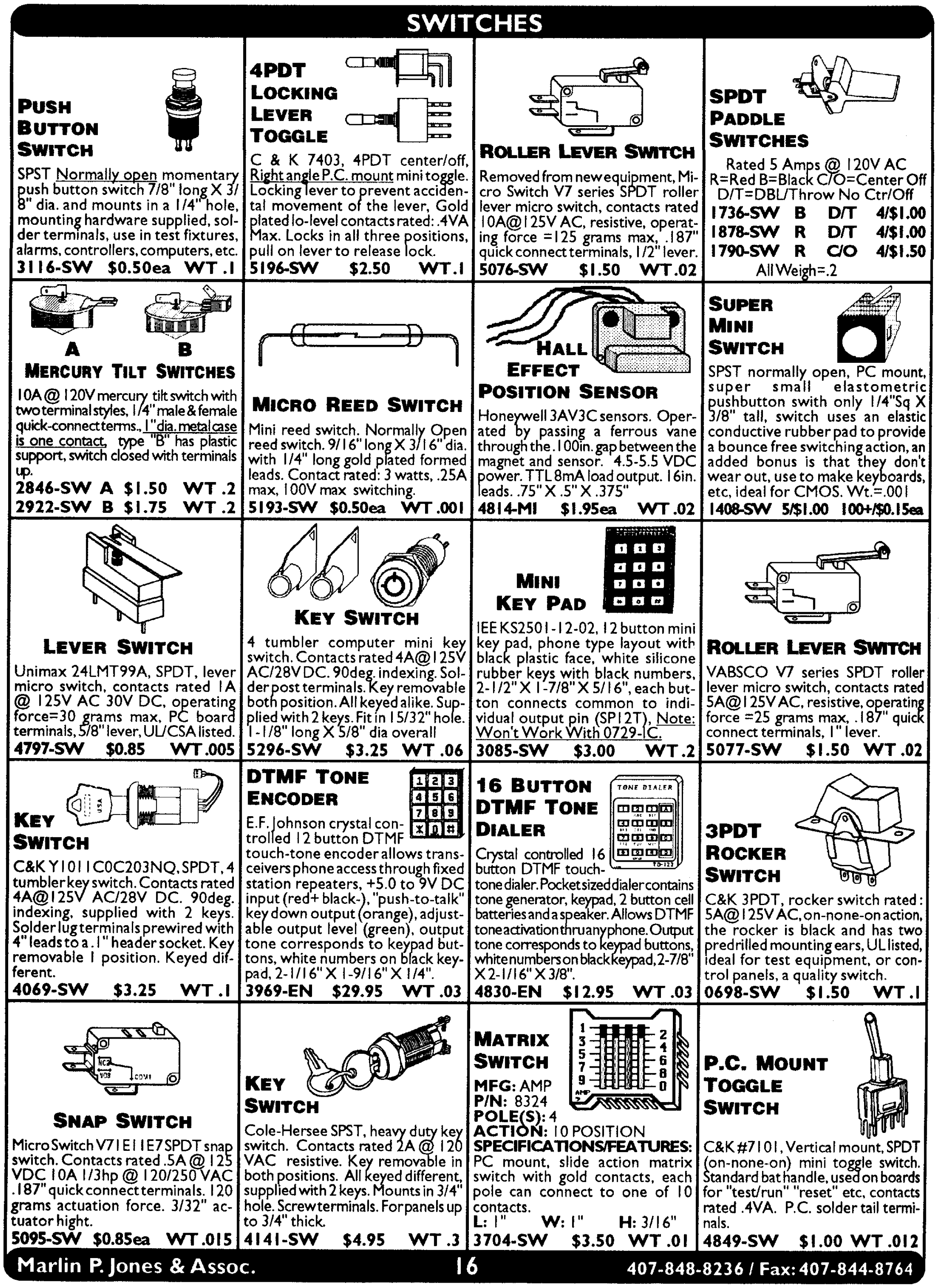
The development of such sensors was an iterative and additive process; each year we would discover, develop, deploy, and evaluate new devices. If they were successful, we would keep them for the next year. If they were not successful, we would either gain the experience needed to make them of value for the subsequent year or discard the errant device and try something different.

This process was often driven by serendipitous discovery as we perused the pages of catalogs from the electronics surplus industry. These companies stocked a hit-and-miss collection of parts at prices typically one-half or one-third of retail prices, and sometimes as low as one-tenth of a commercial distributor's price. The primary drawback of using parts from the surplus industry was not their quality—parts were generally in new condition—but the fact that stock could vary widely from year to year, making it necessary to search for desirable parts on an on-going basis.

The surplus world turned out to be a boon for our project. For example, consider the selection of switches from the catalog page of a typical surplus company, *Marlin P. Jones & Associates*, shown in Figure D-5. Any number of these devices would serve wonderfully as touch sensors for a robot. In fact, we used the "Super Mini Switch" shown in the Figure, catalog number 1408–SW, for this purpose for several consecutive years. A switch comparable to the Super Mini, which *MPJA* sold for 15 cents each in quantities of one hundred or more, would cost about $5 each from a retail dealer.

The design of each of the contest activities was based on the availability of sensors that would be instrumental in solving it. For example, two of the contests used inclined playing fields (*King* and *Robo-Pong*) and we supplied specific sensors to detect inclination to the students. Other contests (e.g., *Robo-Cup* and *Robo-Pong*) relied on sensing surface reflectance, and sensors were provided for this capability.

The remainder of this section details the development, deployment, and evaluation of



SWITCHES

PUSH BUTTON SWITCH
SPST Normally open momentary push button switch 7/8" long X 3/8" dia. and mounts in a 1/4" hole, mounting hardware supplied, solder terminals, use in test fixtures, alarms, controllers, computers, etc.
3116-SW $0.50ea WT .1

4PDT LOCKING LEVER TOGGLE
C & K 7403, 4PDT center/off, Right angle P.C. mount mini toggle. Locking lever to prevent accidental movement of the lever, Gold plated lo-level contacts rated: .4VA Max. Locks in all three positions, pull on lever to release lock.
5196-SW $2.50 WT.1

ROLLER LEVER SWITCH
Removed from new equipment, Micro Switch V7 series SPDT roller lever micro switch, contacts rated 10A@125V AC, resistive, operating force =125 grams max, .187" quick connect terminals, 1/2" lever.
5076-SW $1.50 WT.02

SPDT PADDLE SWITCHES
Rated 5 Amps @ 120V AC
R=Red B=Black C/O=Center Off
D/T=DBL/Throw No Ctr/Off
1736-SW B D/T 4/$1.00
1878-SW R D/T 4/$1.00
1790-SW R C/O 4/$1.50
All Weigh=.2

A B
MERCURY TILT SWITCHES
10A@120V mercury tilt switch with two terminal styles, 1/4" male & female quick-connect terms., 1" dia. metal case is one contact, type "B" has plastic support, switch closed with terminals up.
2846-SW A $1.50 WT .2
2922-SW B $1.75 WT .2

MICRO REED SWITCH
Mini reed switch. Normally Open reed switch. 9/16" long X 3/16" dia. with 1/4" long gold plated formed leads. Contact rated: 3 watts, .25A max, 100V max switching.
5193-SW $0.50ea WT .001

HALL EFFECT POSITION SENSOR
Honeywell 3AV3C sensors. Operated by passing a ferrous vane through the .100in. gap between the magnet and sensor. 4.5-5.5 VDC power. TTL 8mA load output. 16in. leads. .75" X .5" X .375"
4814-MI $1.95ea WT .02

SUPER MINI SWITCH
SPST normally open, PC mount, super small elastometric pushbutton swith only 1/4"Sq X 3/8" tall, switch uses an elastic conductive rubber pad to provide a bounce free switching action, an added bonus is that they don't wear out, use to make keyboards, etc, ideal for CMOS. Wt.=.001
1408-SW 5/$1.00 100+/$0.15ea

LEVER SWITCH
Unimax 24LMT99A, SPDT, lever micro switch, contacts rated 1A @ 125V AC 30V DC, operating force=30 grams max, PC board terminals, 5/8" lever, UL/CSA listed.
4797-SW $0.85 WT.005

KEY SWITCH
4 tumbler computer mini key switch. Contacts rated 4A@125V AC/28V DC. 90deg. indexing. Solder post terminals. Key removable both position. All keyed alike. Supplied with 2 keys. Fit in 15/32" hole. 1-1/8" long X 5/8" dia overall
5296-SW $3.25 WT .06

MINI KEY PAD
IEE KS2501-12-02, 12 button mini key pad, phone type layout with black plastic face, white silicone rubber keys with black numbers, 2-1/2" X 1-7/8" X 5/16", each button connects common to individual output pin (SP12T). Note: Won't Work With 0729-IC.
3085-SW $3.00 WT.2

ROLLER LEVER SWITCH
VABSCO V7 series SPDT roller lever micro switch, contacts rated 5A@125V AC, resistive, operating force =25 grams max, .187" quick connect terminals, 1" lever.
5077-SW $1.50 WT .02

KEY SWITCH
C&K Y1011C0C203NQ, SPDT, 4 tumbler key switch. Contacts rated 4A@125V AC/28V DC. 90deg. indexing, supplied with 2 keys. Solder lug terminals prewired with 4" leads to a .1" header socket. Key removable 1 position. Keyed different.
4069-SW $3.25 WT .1

DTMF TONE ENCODER
E.F. Johnson crystal controlled 12 button DTMF touch-tone encoder allows transceivers phone access through fixed station repeaters, +5.0 to 9V DC input (red+ black-), "push-to-talk" key down output (orange), adjustable output level (green), output tone corresponds to keypad buttons, white numbers on black keypad, 2-1/16" X 1-9/16" X 1/4".
3969-EN $29.95 WT .03

16 BUTTON DTMF TONE DIALER
Crystal controlled 16 button DTMF touch-tone dialer. Pocket sized dialer contains tone generator, keypad, 2 button cell batteries and a speaker. Allows DTMF tone activation thru any phone. Output tone corresponds to keypad buttons, white numbers on black keypad, 2-7/8" X 2-1/16" X 3/8".
4830-EN $12.95 WT .03

3PDT ROCKER SWITCH
C&K 3PDT, rocker switch rated: 5A@125V AC, on-none-on action, the rocker is black and has two predrilled mounting ears, UL listed, ideal for test equipment, or control panels, a quality switch.
0698-SW $1.50 WT .1

SNAP SWITCH
Micro Switch V71E11E7 SPDT snap switch. Contacts rated .5A @ 125 VDC 10A 1/3hp @ 120/250 VAC .187" quick connect terminals. 120 grams actuation force. 3/32" actuator hight.
5095-SW $0.85ea WT .015

KEY SWITCH
Cole-Hersee SPST, heavy duty key switch. Contacts rated 2A @ 120 VAC resistive. Key removable in both positions. All keyed different, supplied with 2 keys. Mounts in 3/4" hole. Screw terminals. For panels up to 3/4" thick.
4141-SW $4.95 WT .3

MATRIX SWITCH
MFG: AMP
P/N: 8324
POLE(S): 4
ACTION: 10 POSITION
SPECIFICATIONS/FEATURES: PC mount, slide action matrix switch with gold contacts, each pole can connect to one of 10 contacts.
L: 1" **W:** 1" **H:** 3/16"
3704-SW $3.50 WT .01

P.C. MOUNT TOGGLE SWITCH
C&K #7101, Vertical mount, SPDT (on-none-on) mini toggle switch. Standard bat handle, used on boards for "test/run" "reset" etc, contacts rated .4VA. P.C. solder tail terminals.
4849-SW $1.00 WT .012

Marlin P. Jones & Assoc. 16 407-848-8236 / Fax: 407-844-8764

Figure D-5: Page from the electronic surplus catalog of *Marlin P. Jones & Associates* (used with permission) a sensor technology created to allow robots to “see” each other. This story illustrates the difficulties in designing a new sensor and creating a role for it in the students’ projects to enrich their learning experience.

D.4.1 Robot Detection

After the *Robo-Puck* contest, we had gained enough experience with the infrared sensor technology to know that it was interesting (see discussion in Section 3.3.1). Having worked out the obscure noise problem, the infrared sensor was an extremely effective way to determine the location of an infrared emitting object (location being defined as an angular coordinate from the position of the sensor toward the object). Surely there would be an interesting way to develop and use the technology in subsequent contests.

One of our thoughts was that it would be a way to provide for multiple game objects, which would in turn allow the development of more complex game-playing strategies. Since the infrared broadcasting and detection worked well for *Robo-Puck*, why not create a game with multiple puck- or ball-like objects, each emitting infrared for the benefit of the robots?

We discussed this idea for future contests and it became clear that there were two significant problems with the concept. The first problem was a technical one. The infrared light from two (or more) objects would interfere destructively when reaching the sensor, meaning that neither (or none) of the objects would be detected. This is to say that if a sensor were trained on light emitted from more than one infrared beacon at a time, it would see no light at all.

The other problem was a more practical difficulty. We were afraid of the job of building a large number of reliable infrared transmitting game objects. We had had enough trouble getting just a few working pucks for the *Robo-Puck* contest, and we had seen the difficulty in helping the group of class participants debug each of their standardized controller boards. The job of designing and manufacturing a collection of transmitters (each with its own battery power—should they use disposable or rechargeable batteries?, etc.) was not one we were willing to stake a contest on.

The alternative of using infrared to detect a single game object had already been tried, so we came up with another idea: building infrared transmitters into the robots, so that robots could "see" each other on the playing field. This idea seemed like an alternative that was practically feasible, and opened up opportunities for students to develop robot behaviors that would be based on the ability to detect the opponent robot.

We developed a method whereby a robot could selectively broadcast infrared light on one of two different frequencies while simultaneously looking for the presence of light on the other frequency. Hence one robot could broadcast light on frequency "A" while detecting light on frequency "B," and the other robot would do the converse. This solution meant that a given robot wouldn't become confused by detecting its own emissions. (It was still necessary to deploy the sensors in a way that shielded a robot's light emissions from its own sensors due to the destructive interference problem mentioned earlier).

In the role of robot-to-robot detection, the infrared sensor technology became an added feature of the robot's sensing capability rather than a central one, as it had been in the *Robo-Puck* contest. Only by seeing it in use would we be able to evaluate its value.

D.4.2 Deploying the System

Implementing the infrared light detection system necessitated additions to the set of rules that students were required to follow in preparing their competitive contest robot. If the detection system was to be useful, all robots would be *required* to broadcast a standardized frequency and "amount" of infrared light. If a robot were to use an infrared-based strategy to find its opponent, it would be severely disadvantaged if the opponent were to cheat.

We therefore installed appropriately strong language in the contest rules requiring robots to broadcast infrared light at the correct frequencies, and provided the necessary hardware and software for them to do so. We also warned the students in no uncertain terms that cheating would not be tolerated and deliberate offenders would be unceremoniously ejected from competitive play.

Nevertheless, we underestimated the cost of voluntary compliance with our plan, both as a responsibility of the students' and our own. In order to ensure fairness, we needed a way to test robots' infrared transmission hardware, both in advance of the contest (to help the students be sure that it was functioning properly) and during the contest (to make sure it didn't fail, either deliberately or inadvertently).

The testing problem was difficult because infrared light is invisible; hence, some sort of specialized equipment was needed to detect its presence. Also, we had to ensure that not only was the hardware functioning, but that the infrared being transmitted was at its nominal full brightness. Each infrared beacon used a total of eight individual transmitting elements; we had to ensure that all eight of them were functioning properly.

In the *Robo-Pong* year, the first year we used the infrared robot detection system, we obtained infrared viewing cameras to look at the robots' infrared transmitters. Using this equipment, we were able to determine that the transmitters were on and appropriately bright, but we were unable to measure the frequency of transmission. This solution was inadequate because the infrared viewers were scarce and students themselves had no easy way of telling if their own circuit was functioning properly.

For the subsequent year, we modified the transmitter device by placing a visible LED lamp in series with each invisible infrared transmitter. This way, one could readily tell which of the eight transmitter LEDs were operating: if a visible LED was illuminated, it was nearly assured that the corresponding infrared one was working too. In order to check for proper frequency, we built a circuit using the same detection hardware as the opponent robot would use and tested for proper transmission using our own detectors.

By the end of the *Robo-Cup* contest, the second year of the infrared robot detection system, it was thus reasonably straightforward to test the students' hardware.

D.4.3 Evaluating the System

Each year only a few student teams fielded final designs which used the infrared robot detection in a central way. There are a number of reasons why this sensor technology did not become a major factor in the students' designs, though most students did experiment with the infrared system.

From the students' perspective while developing their robots' strategy, the infrared detection system was only of significant value if they were designing an aggression-based robot. If they focused on their robot's own task, it usually meant ignoring the presence of the opponent robot and hence its infrared light transmissions. As discussed in Section 3.1, we were diligent in the design of our contests to create games where brute force aggression would not be rewarded, especially after the oversight in the design of the *Robo-Puck* contest. Further, beginning with the *Robo-Cup* contest, students were required to demonstrate that their robot could beat a “placebo” (i.e., non-functioning) robot in order to qualify for the main contest night. (The placebo did transmit infrared light as a normal opponent would.)

Most teams planned aspects of their strategy to deal with the opponent robot, but these ideas were not the cornerstone of their robot’s strategy. As time ran out, the infrared-based strategy became a disposable “extra” that was indeed disposed. This was evidenced by the vast majority of robots which had infrared sensors mounted and installed, but ultimately not used in the final competitive program version.

For the teams that did complete a design based on infrared detection of the opponent, it was typically critical to their success that the opponent robot was indeed transmitting infrared light. During contest rounds, these teams had a vested interest in making sure their opponents’ tranmission was valid, and had no qualms about being vocal about this. (Could there be a correlation between a student’s own personality and his or her choice of an aggressive design for a robot?)

During the contests a number of teams were disqualified because their robots were not transmitting infrared properly. It was understandable that students who had themselves given up on using infrared detection for their own robot’s strategy would be neglectful in ensuring that their own robot was transmitting infrared properly. Nevertheless, we really had no choice other than to strictly enforce the transmission rule. It was generally apparent that a lack of transmission or faulty transmission was due to oversight and not malice, but it was little consolation for students who would forfeit rounds (sometimes well-played ones) on such a technicality.

One particular round during the *Robo-Pong* contest epitomizes the unexpected difficulty in using the infrared detection system. In this round, a well-designed attack robot went up against a typical opponent. The attack robot, named *50/50*, used a scanning infrared sensor that panned like a radar dish, seeking the opponent’s signal. When the sensor detected the opponent, the robot’s body would spin underneath it to face the direction of the sensor, and the robot would ram its opponent.

50/50 had done well in previous rounds, but this time it missed the opponent entirely, and persistently drove into a boundary wall at the edge of the playing field. The robot lost the round as the opponent succeeded in its own task, oblivious of the hazard it had been spared.

Shortly after the round had ended, the robot's designers came up to me in a very agitated state. Other rounds had already begun and it was a very hectic time. The designers claimed that their robot had indeed locked on to the opposing robot's infrared signal—one that was *reflecting off a judge's pair of pants!* They pointed to the accused judge, who was wearing a clean pair of white pants, which are ideal infrared reflectors. He was standing quite near to the edge of the table; he happened to be the judge who was responsible for checking robot's infrared transmissions, and he needed to be up close to do it. The students called for a replay of the round they had just lost, citing this interference from the judge.

I had no way of knowing for certain whether the story I was hearing was true: I had not paid enough attention to the performance round in question. While the story was technically plausible, our predisposition during the contest was to reject complaints; there were too many things going wrong if we paid attention to them all. So I rejected the students' call for a rerun of the round, over their strenuous objections. It was only in later reviewing a contest videotape that I now believe their claim was valid. Technically, it certainly could have happened; the video record suggests that it indeed did. In subsequent years, to minimize the reflection problem, we established a rule that contest judges had to wear black pants. Students continue to use the infrared system; in recent years, there have been several robots that successfully used the system to detect and compensate for the actions of the opponent robot.

The story of the development of each particular sensor types can serve a microcosm for explaining the fashion in which the whole Robot Design project has been developed: We had a sense of what would be an interesting and viable technology; we experimented and developed the technology into workable form; we built it into the design space of the class by employing it in the contest specification; and then we evaluated the result of its use.

Sensor development is the most active portion of on-going technology development in the years of the Robot Design project since those discussed here. The students who have taken over the administration and development of the project continue to explore sensor technologies for ones that would be viable and instructive to add to the course. The infrared robot detection system remains in place; in recent years, students have employed it to detect their own robot's orientation at the start of a round. By mounting four infrared sensors in ninety degree quadrants, they can detect the direction to the opponent at the start of the round, which yields an indication of their own robot's randomized orientation.

Bibliography

Abelson, H., & Sussman, G. J. (1985). *Structure and Interpretation of Computer Programs.* The MIT Press, Cambridge, Massachusetts.

Ackermann, E. (1991). The agency model of transactions: Toward an understanding of children's theory of control. In Montangero, J., & Tryphon, A. (Eds.), *Psychologie Génétique et Sciences Cognitives*, chap. 4, pp. 63–75. Fondation Archives Jean Piaget, Genève.

Anderson, P. H. (1991). Amateur radio in an electrical engineering program. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 338–345. I.E.E.E. and A.S.E.E.

Arneke, D. (1990). Sports news: Robots play hockey. *Game Player's Sports for Kids*, *1*(5).

Asimow, M. (1962). *Introduction to Design*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Banios, E. W. (1991). Teaching engineering practices. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 161–168. I.E.E.E. and A.S.E.E.

Breger, L. (1989). A *creative* design class on piezo- and pyro-electricity. In *Proceedings of the Frontiers in Education conference*, pp. 167–169. I.E.E.E. and A.S.E.E.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, *RA*(2), 14–23.

Bucciarelli, L. L. (1988). Engineering design process. In Dubinskas, F. A. (Ed.), *Making Time: Ethnographies of High-Technology Organizations*, chap. 3, pp. 92–122. Temple University Press, Philadelphia.

Chandler, D. L. (1992). Robotics soccer gives ideas a fighting chance. February 10, 1992. *The Boston Globe*.

Connell, J. H. (1988). A behavior-based arm controller. A.I. Memo 1025, MIT Artificial Intelligence Laboratory.

Ferguson, E. S. (1992). *Engineering and the Mind's Eye*. The MIT Press, Cambridge, Massachusetts.

Flowers, W. C. (1987). On engineering students' creativity and academia. In *ASEE Annual Conference Proceedings*.

Freedman, D. H. (1990). Robo-hockey. *Discover*, *11*(5).

Grinther, L. (1954). Interim report of the committee on evaluation of engineering education. *Journal of Engineering Education*, *45*, 40–66.

Grinther, L. (1955). [Final] report of the committee on evaluation of engineering education. *Journal of Engineering Education*, *46*, 25–60.

Jolda, J. G., Barber, P. F., & Lane, W. D. (1991). Teaching electrical engineering to non-science majors at West Point. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 682–686. I.E.E.E. and A.S.E.E.

Jones, J. B. (1991). Design at the frontiers of engineering education. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 107–111. I.E.E.E. and A.S.E.E.

Klein, R. E. (1991). The bicycle project: A vehicle to relevancy and motivation. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 47–52. I.E.E.E. and A.S.E.E.

Maes, P. (1990). A bottom-up mechanism for behavior selection in an artificial creature. In *Proceedings of the conference Adaptive Behavior: From Animals to Animats*.

Magleby, S. P., Sorensen, C. D., & Todd, R. H. (1991). Integrated product and process design: A capstone course in mechanical and manufacturing engineering. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 469–474. I.E.E.E. and A.S.E.E.

Martin, F. (1992). Building robots to learn design and engineering. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education Conference* Vanderbilt University in Nashville, Tennessee.

Martin, F. (1993). The MIT robot design toolkit. In Estes, N., & Thomas, M. (Eds.), *Proceedings of the Tenth International Conference on Technology and Education* The University of Texas at Austin.

Martin, F., & Sargent, R. (1992). Learning engineering through robotic design. In Estes, N., & Thomas, M. (Eds.), *Proceedings of the Ninth International Conference on Technology and Education*, pp. 1191–1193 The University of Texas at Austin.

Martin, F. G. (1988). Children, cybernetics, and programmable turtles. Master's thesis, The Massachusetts Institute of Technology, M.I.T. Media Laboratory, 20 Ames Street Room E15–315, Cambridge, MA 02139.

Martin, F. G. (1992). The 6.270 Robot Builder's Guide. Epistemology and Learning Manual 1, MIT Media Laboratory, 20 Ames Street Room E15–315, Cambridge, MA 02139. Epistemology and Learning Publications.

Martin, F. G. (1993). The Mini Board 2.0 Technical Reference. Epistemology and Learning Manual 2, MIT Media Laboratory, 20 Ames Street Room E15–315, Cambridge, MA 02139. Epistemology and Learning Publications.

Meeden, L. A., McGraw, G., & Blank, D. (1993). Emergence of control and planning in an autonomous vehicle. In *Proceedings of the Fifteenth Annual Conference of the*

Cognitive Science Society, pp. 735–740 Hillsdale, New Jersey. Lawrence Erlbaum Associates.

M.I.T. (1961). Report on engineering design. *Journal of Engineering Education*, *51*(8), 645–660.

Neumann, P. G. (1993). Inside risks: Modeling and simulation. *Communications of the ACM*, *36*(4), 124.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, Inc.

Papert, S. (1986). Constructionism: A new opportunity for elementary science education.. Proposal to the National Science Foundation. MIT Media Laboratory.

Papert, S. (1991). New images of programming: In search of an educationally powerful concept of technological fluency. Proposal submitted to the National Science Foundation from the Massachusetts Institute of Technology, The Media Laboratory, Epistemology and Learning Group.

Papert, S., & Solomon, C. (1971). Twenty things to do with a computer. Logo Memo 3, Massachusetts Institute of Technology, 20 Ames Street Room E15–315, Cambridge, MA 02139.

Prados, J. W. (1992). Can ABET change its spots?. *ASEE PRISM*, *2*(2), 17.

Rashid, M. H. (1991). Integration of design in electronics courses. In Grayson, L. P. (Ed.), *Proceedings of the Frontiers in Education conference*, pp. 63–66. I.E.E.E. and A.S.E.E.

Resnick, M. (1990). Xylophones, hamsters, and fireworks: the role of diversity in constructionist activities. Epistemology and Learning Memo 9, MIT Media Laboratory, 20 Ames Street Room E15–315, Cambridge, MA 02139.

Resnick, M., & Ocko, S. (1990). LEGO/Logo: Learning through and about design. Epistemology and Learning Memo 8, MIT Media Laboratory, 20 Ames Street Room E15–315, Cambridge, MA 02139.

Roberts, E., & Berlin, S. (1987). Computers and the strategic defense initiative. In Bellin, D., & Chapman, G. (Eds.), *Computers in Battle: Will They Work?*, pp. 149–170. Harcourt Brace Jovanovich, Orlando, Florida.

Sargent, R., & Resnick, M. (1993). Twenty things to do with a programmable brick. Internal memo, Epistemology and Learning Group, MIT Media Laboratory, Cambridge, MA 02139.

Schön, D. A. (1982). *The Reflective Practitioner: How Professionals Think in Action.* Basic Books, Inc.

Simon, H. A. (1969). *The Sciences of the Artificial* (first edition). The M.I.T. Press, Cambridge, Massachusetts.

Troxel, D. E. (1968). A digital systems project laboratory. *The IEEE Transactions on Education*, *E-11*(1), 41–43.

Turkle, S., & Papert, S. (1990). Epistemological pluralism: Styles and voices within the computer culture. *Signs*, *16*(1).

Vincenti, W. G. (1990). *What Engineers Know and How They Know It: Analytical Studies from Aeronautical History*. John Hopkins Studies in the History of Technology. The John Hopkins University Press, Baltimore and London.

Whitney, D. E. (1990). Designing the design process. *Research in Engineering Design*, *2*, 3–13.