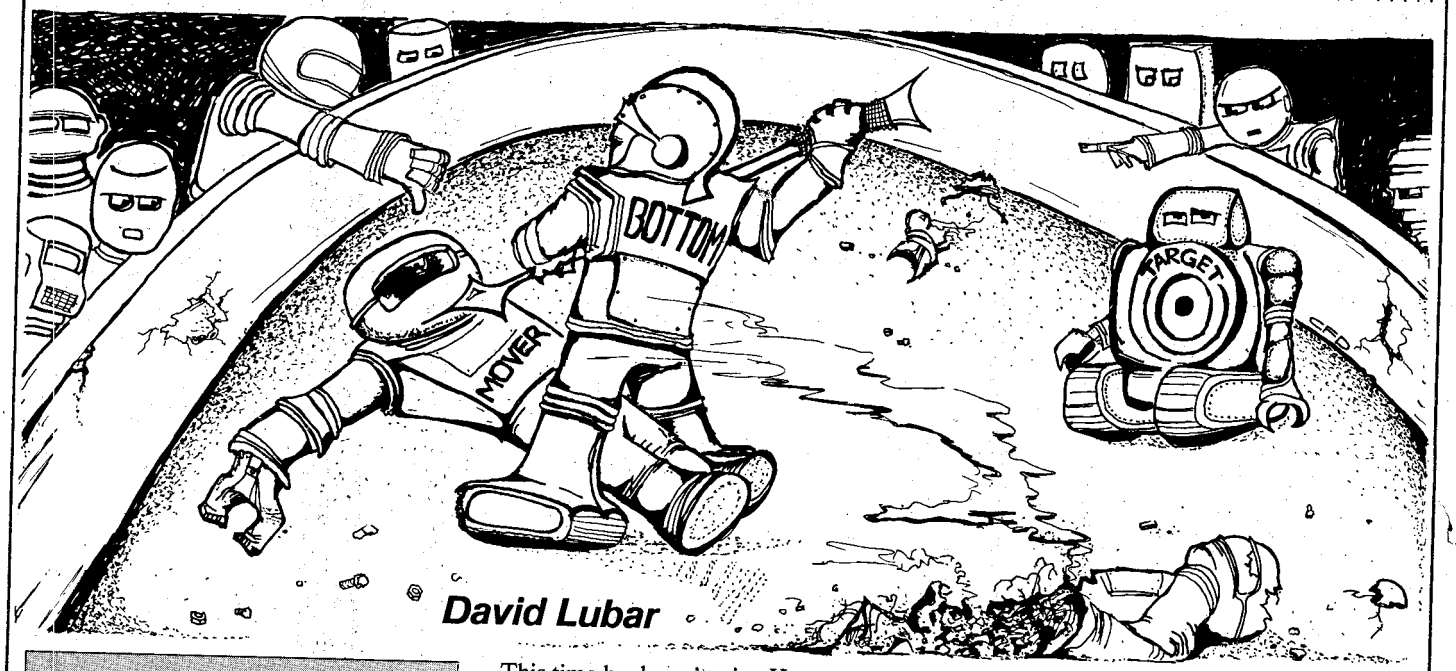
creative computing software evaluation

ROBOTWAR



David Lubar

creative computing

SOFTWARE PROFILE

Name: Robotwar

Type: Game

System: 48K Apple II, Applesoft, Disk Drive

Format: Disk

Language: Machine language

Summary: Complete game system for developing gladiator robots

Price: $39.95

Manufacturer: Muse Software
330 N. Charles St.
Baltimore, MD 21201

His name is Bottom. His battle plan has carried him through scores of contests in the arena. Bottom always seeks the south wall of the arena, hugs it and moves back and forth while aiming his radar to the north. But Bottom is blind to his flanks. During the battle, Mover slips down, locks on his target, and fires. Bottom keeps coming. Mover shoots again. They collide and both are damaged. But the shots have taken their toll. Bottom succumbs. Mover senses his damage and goes to a new location. He spots another target. But this one moves before Mover fires. The shell flies past the target. Mover scans clockwise with his radar. He finds the target again. This time he doesn't miss. He stands alone in the arena, the enemies are gone.

Bottom and Mover are two inhabitants of the universe of *Robotwar*. They have been programmed to find and destroy all who enter the arena. Were this all that occurred, Robotwar from Muse would be just another game. But there is more. Robotwar is a full system that allows the user to program and test his own robots. Among other things, this makes Robotwar an excellent educational tool.

Robot Talk

Bottom and his disk mates are programmed in a special robot assembly language. In essence, each robot acts as a small central processing unit. Actions are carried out by sending values to specific registers within the robot. For instance, the RADAR register activates the radar of the robot. A value from 0 to 355 can be placed in the register, causing the robot to send a beam at the appropriate angle. At this point the RADAR register will contain a new value indicating the result of the radar beam. The absolute value of this number tells the distance to the spotted object. The sign of the number tells the type of object. Walls return a positive value; other robots return a negative value. The robots also contain an AIM register which is used for pointing a gun, a SHOT register which fires a projectile, SPEEDX and SPEEDY registers to control movement, a DAMAGE register which indicates the percent of injury sustained, a RANDOM register for generating random numbers, and X and Y registers to indicate position. There are twenty-four storage registers and an INDEX register. The storage registers are for holding values. The index allows the programmer to address registers indirectly. Each register has a number. Thus, if SPEEDX is number seven, you can place a seven in the INDEX register and any access to INDEX will actually access SPEEDX. This is a powerful capability. For instance, by putting a base number plus a random value into INDEX, you can randomly access either SPEEDX or SPEEDY.

O.K., there is a set of registers. What can you do with them? Well, the robot language also contains a set of commands. A value can be sent to a register with the TO command. A TO B takes the value in A and places it in B. Arithmetic operations are allowed, such as A<5 TO SPEEDX. The flexibility of the robots is increased with GOTO and GOSUB, and their "intelligence" amplified with IF statements. Putting it all together, a simple scan and shoot routine would be:

```
LOOP
AIM+5 TO AIM
AIM TO RADAR
IF RADAR<0 THEN GOSUB SHOOT
GOTO LOOP
SHOOT
0-RADAR TO SHOT
ENDSUB
```

In this example, LOOP is a label. The next line increases the aim of the gun. If the radar reading is less than zero, a robot has been spotted. In this case, the SHOOT subroutine is called. Placing the distance of the robot (0-RADAR) into SHOT causes a shell to go in the direction of AIM and explode at the instructed distance. END-SUB causes the program to return from the subroutine and continue execution at the line following the call.

The above, of course, is just one possible program segment. There are many ways to use radar, and many strategies. This is part of the value of Robotwar. Owners of the disk are enticed into creating better, more effective robots. If two robots employ the same strategy, the one using more efficient code will probably triumph. The user begins to ask questions such as "What if radar oscillated instead of sweeping?" or "How large an increment can be used without too much risk of scanning past a target?" These "what ifs" can be put to the test. In the arena, program flaws become painfully obvious, but there is a less painful way to test robots. The disk contains a test bench which executes the code and displays register contents. Radar spottings and damage can be simulated. In essence, the test bench provides full debugging features, including single step and trace.

Even* Target, *whose strategy is to sit in place and do nothing, has been known to win once in a while.

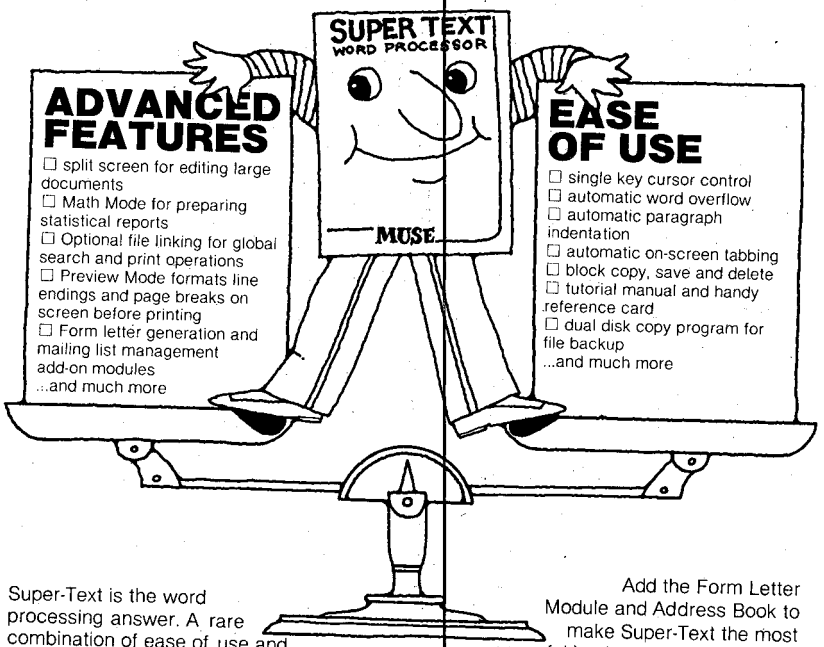
Let's step through some features of the disk. The menu begins with an option to put robots into the arena. If this is selected, the user sees a roster of available robots. From two to five can be selected. Once the choice is made, the battle begins. The robots appear as square, hexagonal, or octagonal shapes, each of which is distinctive. The side of the arena contains a display showing each shape, the name of the robot, percent of damage, and score. Every time a robot is destroyed, the survivors gain a point. Radar is shown by a white flash, the aim of the gun is indicated with a line inside the robot. Projectiles are small dots. Each robot in turn carries out an order. Since this is a sequential process, the program speeds up as robots are eliminated. With two robots, the action is quite fast.

Instead of a single battle, the user can schedule a tournament. Here, a running score is kept, allowing robots to be tested for long-term performance. It is possible for an inferior robot to win occasionally, but unlikely that such a robot will win a tournament. Even Target, whose strategy is to sit in place and do nothing, has been known to win once in a while.

The editor for writing robot code is a version of Supertext, and is easy to use. Since labels are allowed, the code doesn't require line numbers. The language is simple enough for beginners, but sophisticated enough to allow for a wide range of robots. Once the code is written, the source and object files can be saved to the master disk, or to a specially formatted backup. Disks can be initialized from the program, which is good since the master disk has limited free space. A robot on a backup disk must be transferred to the master before entering a tournament. This is an easy process which is fully explained in the documentation.

The manual contains full instructions for programming robots, including many examples of different routines, and clear descriptions of registers and robot language. The only weakness in the system is the use of several different units of measurement. The arena is 400 meters across. The X and Y locations of the robot are given as values from 0 to 255, while the speed of the robots is given in decimeters. Aside from this, *Robotwar* is both an excellent game and an excellent learning tool. □

SUPER-TEXT™



ADVANCED FEATURES

☐ split screen for editing large documents
☐ Math Mode for preparing statistical reports
☐ Optional file linking for global search and print operations
☐ Preview Mode formats line endings and page breaks on screen before printing
☐ Form letter generation and mailing list management add-on modules
...and much more

EASE OF USE

☐ single key cursor control
☐ automatic word overflow
☐ automatic paragraph indentation
☐ automatic on-screen tabbing
☐ block copy, save and delete
☐ tutorial manual and handy reference card
☐ dual disk copy program for file backup
...and much more

Super-Text is the word processing answer. A rare combination of ease of use and advanced features. Super-Text turns an Apple computer into a powerful word processing system with capabilities unmatched by many dedicated WP systems costing thousands of dollars more.

Add the Form Letter Module and Address Book to make Super-Text the most powerful business correspondence system. Use Super-Text to create a letter, then print a personalized copy for everyone on your mailing list. Super-Text is $150, the Form Letter Module $100 and the Address Book $50 at computer stores everywhere.

From the leader in quality software...

For the Apple II or Apple II Plus (48K)
Apple is a trademark of Apple Computer Corp.

MUSE SOFTWARE™
330 N. Charles St.
Baltimore, MD 21201
(301) 659-7212

CIRCLE 225 ON READER SERVICE CARD