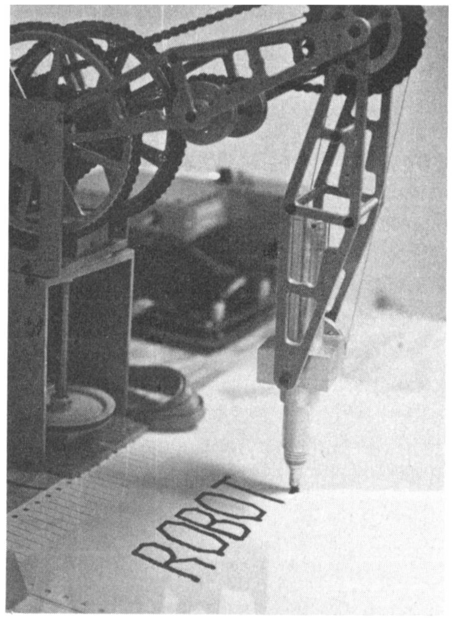
TEACHING THE RHINO XR-1 TO WRITE

Jorg Jemelka
Digital Computer Lab, Room 53
1304 West Springfield Avenue
Urbana, Illinois 61801

Sandhu Machine Design and Vertel Incorporated, recently consulted the Robotics Group at the University of Illinois at Urbana-Champaign on the development of a system, based on the Rhino XR-I, which would generate written output, translating the ASCII character set into an alphabet for execution by the Rhino. The project specified that characters were to be approximately two inches high with a 20-character line width. The development of the end effector was relegated to the facilities at Sandhu, while the control scheme would be developed by the Robotics Group.

The first several weeks of the project consisted of a feasibility study with a limited alphabet and focused primarily on enhancing the Rhino's reliability. The host computer was the North Star Horizon which was already used to control the Rhino. Existing software, written in North Star BASIC, was used to transform xyz coordinates into Rhino step sequences.



Rhino Modifications; The Rhino is powered by servomotors and provides feedback from motor-mounted encoders. A step is defined as one-sixth of a motor revolution, and the motors have a built-in 65-to-1 reduction. Further reduction is produced by gears between the motors and the joint being controlled.

Commands to the Rhino are in the form of ASCII strings communicated serially by an on-board Intel 8251 ACIA. The first character of the command string is a motor code (ASCII A..G). The second is an ASCII direction code (+ or -). The rest are the individual digits of the step count. The steps issued to a given motor can range from 0 to 121, since the feedback registers on the Rhino's Intel 8748 control processor use the high-order bit as a direction flag. Step commands larger than 127 drive the motors in the reverse direction intended, for the number of steps in excess of 127.

Goals of the project included repeatability on the order of I/lO of an inch and preventing a command sequence from being *swa//owed* by the Rhino. To ensure the reliable receipt of command sequences, the interrupt line from the Rhino's 8251 ACIA (pin 14), is fed back to pin 20 of the RS-232-C connector to serve as a clear-to-send (CTS) signal for the host computer . This signal (inverted) inhibits the host from transmitting a character while the Rhino is busy processing an earlier command character .

Positional repeatability is enhanced with the addition of a *hard home* position. The limit switches on the Rhino are relocated to provide an unambiguous initial state for the manipulator and to simplify the implementation of a single-step home routine. The limit switch for the shoulder is mounted high on the base of the Rhino, locating the shoulder 45 degrees to the rear. Moving the switch to the base, where it is activated by shoulder contact, provides the advantage that if the switch is not closed (on) the only search direction is up (ASCII '-'). A closed switch means the arm is in the home position.

The unmodified limit switches are actuated by cams mounted on the large gears which drive the various axes. A search strategy using these cams must consider that the cam may be above or below the limit switch. Should the search in either the plus or minus direction be unsuccessful (should the joint collide with the body) before the switch is closed, this "stuck" condition must be detected, and the search direction must be reversed.

The cam profile creates a "dead" zone of some 60 to 80 steps, further complicating the definition of a home position. Mounting the limit switches so that they are activated directly by the driven joint ensures that the home position can be defined to within a single encoder step.

The elbow limit switch is mounted on the shoulder and is closed when the elbow joint just makes contact with the shoulder joint. In this case, either the switch is closed and the elbow is *home,* or the elbow must be driven in the minus direction towards its home posi tion. The base limit switch is mounted so that it is closed when the base rides over it. In this case, if the switch is closed, the Rhino moves in the plus direction until the switch is open. If the switch is open, the Rhino moves in the opposite direction until the switch is closed and then reverses until the switch is open again.

There are cases in which the base strategy fails, but these require a base error in excess of 45 degrees (300 encoder steps). This amount of error is unlikely. The base home position is "edge sensitive." It provides a considerable enhancement of positional accuracy relative to the actuating cam normally provided, since it eliminates the cam dead zone and the attendant complication of searching out the plus and minus cam limits.

Writing. A provisional writing capability was first improvised by the removal of the finger assembly and its threaded open/close shaft. The hollow shaft which provides finger rotation was used to hold an ordinary ball-point refill and was spring-mounted on the hand assembly to provide approximately 25 millimeters of spring travel. The tolerances between shaft and refill were close enough to allow free movement and reasonable pen stability.

The control strategy consists of two parts. The first part moves the motors from home to the first character coordinate, driving the motors the number of steps required at full speed without consideration for coordinated joint movement. This same uncoordinated approach moves the manipulator from the end point of one character to the starting point of the next. A second strategy is employed while actually tracing out the individual letters. To produce coordinated joint movements, the servomotors are driven one step at a time. The use of the individual step sensing capabilities of the encoders obviates the need for speed control in coordinating the various joints and reduces the control problem to a form suitable for solution using digital difference analyzer (DDA) algorithms.

DDA algorithms are commonly encountered in computer graphics when the slope of a line is to be plotted. An example is plotting points on a screen when only integer coordinates are attainable and the y-coordinate may have a fractional component for a given x. In this case, if a line is to be traced from some start xy to some end xy, the base, elbow, and shoulder motors all may play a part in accomplishing the move. The motor with the largest number of steps is used as the *master* and is stepped continuously. The remaining motors are stepped if their total step count, when subtracted from the total step count of the master motor, drops below zero. This subtraction is performed once for every step of the master motor.

When the zero threshold is crossed for any motor, the master motor's step count is added to the negative quantity remaining, the motor is stepped, and the procedure continues until the master motor has no steps remaining. To ensure that the master motor is continuously stepped, the initial quantity from which each motor's step count is subsequently subtracted is one half the master motor's step count. This procedure provides a zero crossover with each subtraction of the master motor's step count for the master motor and a correct time interval for stepping the remaining motors.

The control algorithm is implemented in Z80 assembly language. Since fine control is needed for only small moves, an 8-bit step count is used, providing a maximum move of 255 steps for any motor in any segment. Care must be taken not to allow the step count to build up in the master motor's command register. A build-up of unexecuted commands results in imprecise coordination of the Rhino's motors. Conversely, reading the master motor's control register and waiting until each step has been executed slows the motors down visibly and results in a speed versus accuracy tradeoff.

To effectively use this "incremental servoing" technique, the characters are divided into segments. A whole segment is defined to be the maximum height (two inches) of the Rhino alphabet. Fractional segments are constants defined as (H)alf, (Q)uarter, and (E)ighth. A whole segment is a 50-millimeter line, with the fractional segments being 25, 13, and 7-millimeter lines respectively. To trace out a letter:

.The manipulator moves from home position to a starting xy position with a z-coordinate approximately 20 millimeters above the table.

- A flag is set to indicate the incremental servoing mode is in effect.
- The manipulator moves down to a z-coordinate that corresponds to a 1/*2* inch spring loading of the pen.
- Line segments are drawn until the character is completed.
- The incremental servoing flag is reset.
- The pen is moved to a z coordinate approximately 20 millimeters above the table.
- The manipulator moves to the next character position.

Once the feasibility of tracing reasonable characters was demonstrated, the Robotics Group developed the complete alphabet and numerals, as well as space, dash, and line feed commands.

As the program neared completion, Sandhu Machine Design delivered a pen holder designed for broad-stroke felt pens. This holder is precision-machined from aluminum stock, with a total sprung travel of two inches. The tolerances were close enough that a hole was drilled to vent pressure buildup. The Robotics Group experimented by varying the size of the opening to effect spring damping. The combination of felt pen and spring damping helped to mitigate the staircase effect that was evident in earlier efforts using the ballpoint refill and the standard hand.

Software Character Definition. Each character is a BASIC subroutine that consists of a sequence of xyz coordinates. As each coordinate is reached, the pen traces a line segment. By way of illustration, the subroutine that draws the letter S is shown below.

```
530 Xl=Nl+E \ Yl=N2-H \ Zl=U \ GOSUB 70 \Zl=D \ GOSUB 70 \ F5=1 531 Xl=Xl-
E\Yl=Yl+E\GOSUB 70\Yl=Yl-t-Q\OOSUB 70\ Xl=Xl+E \Yl=Yl+E 532 GOSUB 70 \ Xl=Xl-t-Q \
GOSUB 70 \ Xl=Xl+E \Yl=Yl-E \GOSUB 70 533 Yl=YI-Q\GOSUB 70\Yl=Yl-E\X1=Xl+E\OOSUB
70\X1=Xl-t-Q OOSUB 70 534 Xl=Xl+E\Yl=Yl+E\GOSUB 70\Yl=Yl-t-Q\GOSUB 70\ Xl=Xl-
E\Yl=Yl+E 535 OOSUB 70 \ F5=O \ Zl=U \ GOSUB 70 \ N2=N2-H-E \ Yl=N2
536 Xl=Nl +H \ OOSUB 70 \ GOTO 335 \ REM 'S'
```

The variables Nl and N2 are offsets for the x- and y-axes from the base center. N2 is updated at the end of each letter subroutine to reflect the center of the next letter field's x center, and the start of the next letter field's y-coordinate. The z-axis can be either up, U, or down, D.

On entry into the S routine, the pen is set at the top left of the letter S and is set down. The F5 flag is set to indicate that incremental servoing is in effect. The remainder of the letter is traced. The F5 flag is reset and the pen is raised in line 535. The subroutine call to line 70 performs the coordinate to motor step transformations and calls the assembly-language drivers according to the flag setting.

Pen movement is from left to right. The start x is 170 millimeters out from the base, and start y is 240 millimeters to the left of the base. Two successive text lines may be written. Best results are attained if the recommended line width of 20 characters is not exceeded. The program also centers each line of text with respect to the center of the Rhino's base.

The final phase of the project is expected to be the connection of the Rhino to a Vertel magnetic card reader .