THE OZNAKI ROBOTICS LANGUAGE OZ

AUTHOR: Harvey A. Cohen
Mathematics Department,
La Trobe University,
Bundoora, Victoria, 3083.

OZ is a computer language for children of primary school age in which the child calls upon a computer termed the WIZARD to direct the activities of a simple robot called ZONKY. The child commands the WIZARD by punching the buttons of a specialised keyboard: the layout of the keys being based on the syntax of OZ. The string of commands to the WIZARD is displayed on a CRT which serves as a message screen. Control characters sent by the WIZARD to ZONKY also appear on the message screen. The WIZARD can "remember" the name given to a string of commands - that is the child can define programs and execute them. The inclusion in OZ of modifiers akin to adjectives/adverbs enables relatively sophisticated iterative programs to be written.

Currently the WIZARD is a PDP-10 computer but a microprocessor WIZARD is under construction. The construction of ZONKY and the implementation of OZ is part of an educational development endeavour called the OZNAKI Project.

Key Words and Phrases: Message passing, Iteration, Robotics.

CR Categories: 1.50, 3.89, 4.29.

1. THE OZ SYSTEM

OZ is a computer system in which children program the activities of a rather rudimentary robot named ZONKY. The development of OZ is part of the OZNAKI Project at La Trobe University. OZNAKI is Polish for sign - and the overriding objective of the OZNAKI Project (Cohen 1976) is to design computer environments in which the abstract symbols of mathematics are given concrete embodiment.

In this section of the paper we supply some background notes and then describe the OZ automaton, ZONKY, and other hardware features of OZ. In Section 2 we present a detailed introduction to the OZ robotics language. Section 3 discusses the educational significance of OZ.

1.1 Background

The idea of a robotics language for children was first conceived by Seymour Papert and his collaborators at Cambridge, Mass. (Feurzig et al 1969, Papert 1971) who developed the LOGO language (Abelson et al 1974) an automaton called a TURTLE, graphic terminal displays, and a symbol display system on the PLATO Plasma screen (Pearlman 1974, Eastwood and Ballard 1975). LOGO at M.I.T. has, however, been largely concerned with computer graphics, the TURTLE being replaced by a triangle that moves about a CRT screen.

1.2 Components of the OZ System

OZ hardware comprises the robot ZONKY, a specialised keyboard, a CRT screen termed the Message Screen, and the computer controller termed the WIZARD. In the current implementation at La Trobe, the WIZARD is a PDP-10 computer, but during 1976 a microprocessor controller will be constructed.

Commands punched out by a child on the OZ keyboard are displayed on the message screen, along with various messages from the WIZARD to the child. Also displayed on the message screen are the ASCII control characters that direct the robot: these messages from the WIZARD to ZONKY are in a distinctive format.

A judicious choice of robot control code was made so that the messages from child user to WIZARD are invisible to ZONKY. Consequently in the DEC-10 OZ system only a single data line from the computer is required.

1.3 OZ Keyboard

A conventional teletype keyboard that generates ASCII characters may be used with OZ. However as we have been teaching very young children (from 5 years upwards) a specialised keyboard was designed. The OZ keyboard has a level surface on which is mounted a plastic overlay on which the areas to punch are printed. The overlay may be marked with a felt pen by the teacher, who may mount various overlays on which only the more basic key positions are indicated. Thus a young child may at his first lesson just use the overlay on which the numbers, the motor commands (F) (B) (R) (L), and the DO IT Button (*) are indicated.

The keys emit a clear metallic click when depressed. Physically

the keys are curved gold plated phosphor bronze strips, directly mounted on the circuit board on which is also mounted LSI encoder and UART etc. The encoder has an N roll feature, so that one key need not be released before the next one is depressed.

The arrangements of keys on the keyboard is based on the syntax of OZ and is shown in Fig.(i). Lower case letters may be used - as is pedagogically desirable - if the message screen used can display them.

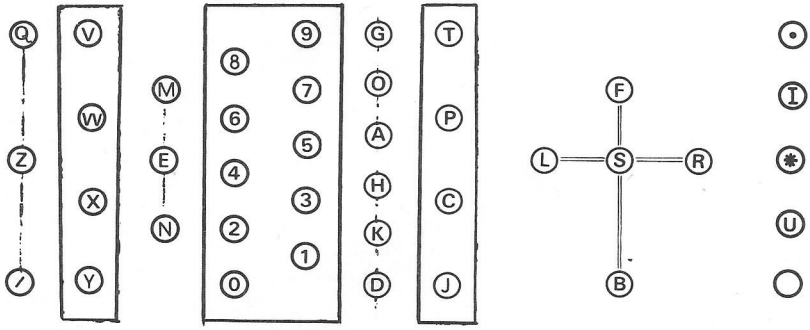


Fig.(i) The OZ Keyboard.

1.4 The Robot ZONKY

This robot crawls about on a level paper surface and while moving or stationary may switch on its lights and toot. A felt tipped pen can be mounted centrally so that the path can be marked on the paper.

The basic mechanical requirements of the robot are that it can move backwards and forwards in a straight line and can turn on the spot about the marker pen. This is achieved if the robot has two parallel driving wheels, driven independently by two reversible motors, and if any other wheels needed for stability can swivel freely.

The original robot, called ZONKY, was based on a $15 plastic toy tank (which had one DC motor driving each track via 60 to 1 gearboxes). The tank was loaded with lead shot - not for its guns - but to minimise drift on turns. Our new model ZONKY will present a less military appearance

Electronically ZONKY is very much at the current state of the art. The all C-MOS controller on board decodes ASCII control characters and switches on the corresponding power circuits. The controller also counts engine revolutions.

ZONKY is gaily painted and looks like any toy. His front is obvious and is marked with a large F. The symbols R and L are marked on wings mounted forward of the centre so that irrespective of which way the robot is facing the direction of turn - whether Right or Left - is obvious.

2. AN OZ PRIMER

The OZ language is composed of strings of symbols (OZNAKI in Polish) punched one by one at the special OZ keyboard of Fig.(i). It is impossible to give a formal description of OZ as the symbols lead not just to mathematical operations but effect the real-time behaviour of an automaton. One wonders if a formalised presentation is worthwhile in any case - five year olds learn the basics of OZ in a few lessons. Certainly a glossary of OZ commands is nearly unintelligible. What we have done in this section is to present the details of the OZ language in the manner of a primer.

The keyboard symbols are regarded by the (child) user as standing for words in the OZ language. And in fact OZ is not so far different from English. In a formal discussion of OZ one might classify these words as numbers, modifiers, nouns, and verbs - modifiers being akin to adjectives. To appreciate the syntax of OZ, just remember that of English! One thinks of these words in OZ as being directed at the computer controller of the OZ system, which is playfully referred to as the WIZARD OF OZ. However there is in OZ one peculiarity about these commands directed at the WIZARD: no matter how many words one says to him/her/it the WIZARD waits for the word (*) before doing anything. Thus roughly (*) is DO IT.

2.1 Symbols and syntax

The symbols used in OZ comprise all 26 letters of the alphabet (either upper case or lower case), the numbers 0 to 9, the dot ".", the slash symbol "/", the space " ", and carriage return which we denote by "*". These symbols are clustered together on the OZ keyboard into the following groups:

Numbers: (0) ... (9)

Modifiers: (G) (D) (H) (A) (O) (K)

Motor and Sit Commands: (F) (B) (R) (L) (S)

Music Commands: (T) (P) (C) (J)

Memory and Print Commands: (Q) (Z) (/)

Variable Commands: (V) (W) (X) (Y)

The layout of the groups on the keyboard is consistent with the order of composing the paradigm OZ statement:

(Z) (X) (2) (H) (A) (F) (3) (T) (*)

Note that modifiers and numbers precede the single command that they modify to yield what might be called a "composite" command.

In teaching very young children (less than five years) keyboard covers can be used that progressively reveal more of the keyboard groups.

2.2 The Motor Commands

The basic commands directing the movements of the robot ZONKY are

(F) = *FORWARD*, (B) = *BACK*, (R) = *RIGHT* and (L) = *LEFT*.

(STEP). The forward (or backward) steps taken are in the direction (or opposite the direction) of the current heading of ZONKY. Right and Left refers to clockwise or anti-clockwise turns (by about 20° per step) on the spot, the direction of turn being marked on ZONKY's wings for the benefit of the child. These OZ "nouns" are invariably used in children's programs in conjunction with the modifiers

(H) = *HONKING* and (A) = *ALIGHT*.

If a student pushes the buttons (in sequence written)

(3)	(F)	(*)
THREE	*FORWARD*	*DO IT*

then the OZ robot only moves forward in three distinct steps (there is a time lapse of 1 second after each forward "step"). The distance moved by ZONKY as indicated by the pen trace mark drawn is three times the distance travelled when ZONKY obeys the instructions

either	(1)	(F)	(*)	or	(F)	(*)
	ONE	*FORWARD*	*DO IT*		*FORWARD*	*DO IT*

Similarly following the command string

(9)	(A)	(R)	(*)
NINE	*ALIGHT*	*RIGHT*	*DO IT*

ZONKY will make nine distinct turns with lights on - with one second gaps between each light flash. Motor commands sent by the computer WIZARD to ZONKY appear on the message along with the commands directed to the WIZARD. Herewith is the printout when the above commands are given.

```
!F
@@@@@@@@@@

F
@@@@@@@@@@

9AR
IIIIIIIIII IIIIIIIIII IIIIIIIIII IIIIIIIIII IIIIIIIIII IIIIIIIIII
IIIIIIIIII IIIIIIIIII IIIIIIIII
```

Suppose ZONKY is now given the order

(3)	(H)	(A)	(F)	(*)
THREE	*HONKING*	*ALIGHT*	*FORWARD*	*DO IT*

then in addition to the three-fold activities executed following (3)(F)(*), in this case the lights are on and the siren is sounding during the forward step. As in the previous instance, the one second time gap between actions makes it easy to perceive the three separate light flashes and the three separate "honks".

Note that there are ten (10) characters output corresponding to each command. Older children use the decimal feature, discussed elsewhere, as instanced by the following print-outs on the message screen:

```
3HAF
CCCCCCCCCC CCCCCCCCCC CCCCCCCCCC

3.4HAF
CCCCCCCCCC CCCCCCCCCC CCCCCCCCCC CCCC

5HL
FFFFFFFFFF FFFFFFFFFF FFFFFFFFFF FFFFFFFFFF FFFFFFFFFF

1.7L
DDDDDDDDDD DDDDDDD

1.6B4HB.2AB
LLLLLLLLLL LLLLLL NNNNNNNNNN NNNNNNNNNN NNNNNNNNNN NNNNNNNNNN MM
```

2.3 The Sit Commands

The OZ command (S) = *SIT* directs the robot to stay still for a definite sit period of approximately one second. (Compare the command (P) = *PAUSE* of 2.6.) Corresponding to each pause unit just one control character - to be followed by a space - is transmitted by the WIZARD to ZONKY. The decimal part of numbers is ignored. The *SIT* command is modifiable just like the motor commands with (H) and (A).

A complete command string would be

(6)	(H)	(A)	(S)	(*)
SIX	*HONKING*	*ALIGHT*	*SITS*	*DO IT*

directing the robot to stay stationary whilst six distinct honks are heard accompanied by light flashes. As for motor commands, there is a one-second gap between each observable action. To direct ZONKY to just make seven toots (honks) requires the command

(7)	(H)	(S)	(*)
SEVEN	*HONKING*	*SITS*	*DO IT*

whilst seven distinct light flashes are emitted following the command string

(7)	(A)	(S)	(*)
SEVEN	*ALIGHT*	*SITS*	*DO IT*

Here are the messages appearing on the message screen corresponding to the above *SITS* at the beginning of a session of OZ.

```
WELCOME TO THE WONDERFUL WORLD OF OZ
THE WIZARD WILL DO WHAT YOU WISH

6HAS
S S S S S S

7HS
R R R R R R R

7AS
Q Q Q Q Q Q Q
```

2.4 The Memory Commands

The use of procedures is a striking aspect of the child's use of the OZ language. In teaching a child, I focus on the idea that the WIZARD knows just four names Ⓥ,Ⓦ Ⓧ Ⓨ which although peculiar, are really names like Jack and Jill. And the child can tell the WIZARD to remember that one of these names is the name of something (that the child itself has defined). For example, if the child, guided by the keyboard layout, punches out

Ⓩ	Ⓧ	③	Ⓗ	Ⓐ	Ⓡ	②	Ⓕ	✱
REMEMBER	*X (IS)*	*THREE*	*HONKING*	*ALIGHT*	*RIGHT*	*TWO*	*FORWARD*	*DO IT*

Then the WIZARD will respond in LARGE TALK (header type letters) that he now knows what Ⓧ"is". On the message screen the output is:

```
ZX3HAR2F

X   X          333   H   H   AAA   RRRR    222   FFFFF
X   X        3   3   H   H  A   A  R   R  2   2  F
 X X   =====      3   H   H  A   A  R   R      2  F
  X            3    HHHHH  A   A  RRRR      2    FFFF
 X X   =====      3   H   H  AAAAA  R R      2     F
X   X          3   3   H   H  A   A  R  R    2      F
X   X           333    H   H  A   A  R   R  22222   F
```

The "variable" commands, Ⓥ Ⓦ Ⓧ Ⓨ are used with numbers just like motor commands. For example, to ask the WIZARD to make ZONKY perform the command called Ⓧ four times, one punches in

④ Ⓧ ✱

On the message screen will then appear the following messages.

```
4X
KKKKKKKKKK KKKKKKKKKK KKKKKKKKKK ●●●●●●●●●● ●●●●●●●●●● KKKKKKKKKK
KKKKKKKKKK KKKKKKKKKK ●●●●●●●●●● ●●●●●●●●●● KKKKKKKKKK KKKKKKKKKK
KKKKKKKKKK ●●●●●●●●●● ●●●●●●●●●● KKKKKKKKKK KKKKKKKKKK KKKKKKKKKK
●●●●●●●●●● ●●●●●●●●●●
```

If ZONKY has his pen down, he will draw 4 sides of a regular (but not necessarily closed) polygon.

Of course just as one can call Jack and Jill the "Horrible Two" so one can tell the WIZARD such commands as the following:

(Z)	(V)	(9)	(H)	(S)	(*)
(Z)	(W)	(9)	(V)	(*)	
(Z)	(Y)	(9)	(W)	(*)	
(Z)	(X)	(9)	(Y)	(*)	

(X) as here defined is a solution to the problem of making ZONKY give as many honks (Honking Sits) as possible with just one command. (Fortunately there is a way of aborting commands!)

When the WIZARD is told what a name stands for, he forgets any earlier use of that name (i.e. the new procedure definition overwrites the old). One can always ask the WIZARD what a particular name stands for as per:

(Q)	(X)	(*)
WHAT'S	*X*	*DO IT*

On the message screen the WIZARD replies in LARGE TALK

```
QX

X   X          999    W   W
X   X        9   9    W   W
 X X   =====  9   9    W   W
  X           9999     W   W
 X X   =====      9    W W W
X   X             9    WW WW
X   X          999     W   W
```

2.5 The Music Commands

In OZ there are three commands suitable for "music" programs: Ⓣ = *TOOT*, Ⓟ = *PAUSE*, Ⓒ = *CLANG*. In-built into the system is also Ⓙ = *JINGLE BELLS*.

The corresponding control characters which effect the "music" effects are not visible so that for these commands the WIZARD's messages to ZONKY are invisible.

Ⓒ causes the terminal bell to operate. Thus the command

Ⓖ	Ⓒ	⊛
SIX	*CLANGS*	*DO IT*

causes six distinct clangs. The command string

Ⓖ	Ⓣ	⊛
SEVEN	*TOOT*	*DO IT*

causes the ZONKY horn and lights to switch on for 7 times the toot duration unit. The command string

Ⓘ	Ⓟ	⊛
NINE	*PAUSES*	*DO IT*

has no observable effect unless sandwiched in some command string. For 9 times the pause duration unit (the pause duration unit is the toot duration unit) ZONKY does nothing.

Here is a typical simple music program.

```
ZX3T2P3T2P5T4P
```

```
X   X          333    TTTTT    222    PPPP     333    TTTTT
X   X        3   3      T     2   2   P   P  3   3     T
 X X   =====      3      T         2   P   P      3     T
  X              3       T        2   PPPP        3    T
 X X   =====      3      T       2    P              3   T
X   X          3   3     T      2     P         3   3   T
X   X           333      T     22222  P          333    T

 222    PPPP    55555   TTTTT   4   4   PPPP
2   2   P   P   5         T     4   4   P   P
    2   P   P   555       T     4   4   P   P
   2    PPPP        5     T     44444   PPPP
  2     P               5  T         4   P
 2      P           5   5  T         4   P
22222   P            555   T         4   P
```

Note its simplicity - yet this program when executed is recognisably the "Jingle Bells" first line of the Christmas Carol. It could be improved by using (C) clang commands. A more advanced program is built into the system as a model program, that for (J).

```
QJ
J=/J/I/N/G/L/E/ /B/E/L/L/S/ /!/ 3T2P3T2P5T4P
```

In this version the words JINGLE BELLS! are output before the music. [The command (/) causes the character following to be output by the terminal.] A more elegant program would spell the words through the music program - synchronising "speech" with "sound".

2.6 Growing and Diminishing

In OZ there are four modifiers

(H) (A) (G) (D)

HONKING *ALIGHT* *GROWING* *DIMINISHING*

We have previously mentioned the (H) and (A) modifiers in describing Motor Commands in Section 2.2. To recapitulate, for the four motor commands, each command leads to WIZARD messages to ZONKY comprising control words of exactly 10 identical characters. When the command is modified by (H) or (A), these control words are still 10 characters long, but the characters involved are different: ZONKY goes exactly the same distance or angle, but performs some additional function.

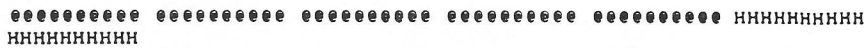
Now, when a motor command is modified by (G) or (D), the effect is to alter the number of characters in each control word, correspondingly altering ZONKY's step size. The effect is well shown in the following example (which draws a converging spiral).

Note that each time the command (X) is called, the number of characters in each (R) step increases by one, while the number in each (F) step decreases by one. The first time (X) is called, these steps have their usual length.

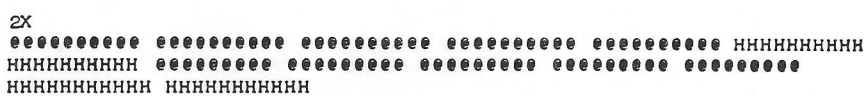
```
ZX5DF2GR
```

```
X   X          55555  DDDD   FFFFF   222    GGGG  RRRR
X   X          5      D   D  F      2   2  G     R   R
 X X    =====  555    D   D  F          2  G     R   R
  X            5     D   D  FFFF       2   G     RFRR
 X X    =====      5  D   D  F         2    G GGG  R R
X   X          5   5  D   D  F        2     G   G  R  R
X   X           555   DDDD   F       22222   GGG   R   R
```

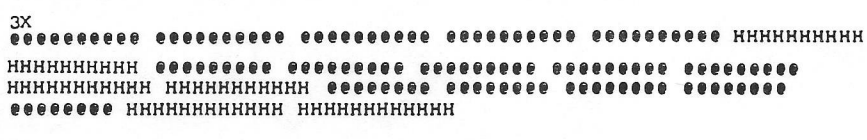
X



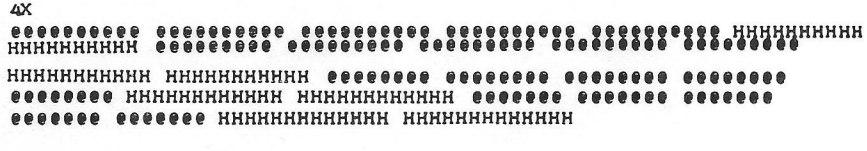
2X



3X



4X



It is worth comparing the above OZ spiral program with the far more complicated "squiral" programs of LOGO (Papert, 1971 a).

The *GROWING* and *DIMINISHING* modifiers also influence the number of characters output by the print command (/). See Section 2.7, where the Xmas tree program provides an amusing use of the modifiers.

A command modified by both (G) and (D) is unaffected, whereas if it is modified twice, i.e. by (G)(G) say, then it increases (or decreases) at twice the rate for a single modifier.

2.7 The Print Command

The command (/) = *PRINT* causes the following character in the command string to be echoed on the message screen. Thus following the command string

(Z) (X) (/) (O) (/) (L) (/) (I) (/) (V) (/) (I) (/) (A) (*)

the command

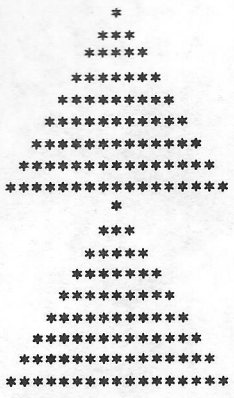
(4) (X) (*)

will lead to "OLIVIA" being written four times on the message screen. The character output is ignored by the robot so there is no other effect.

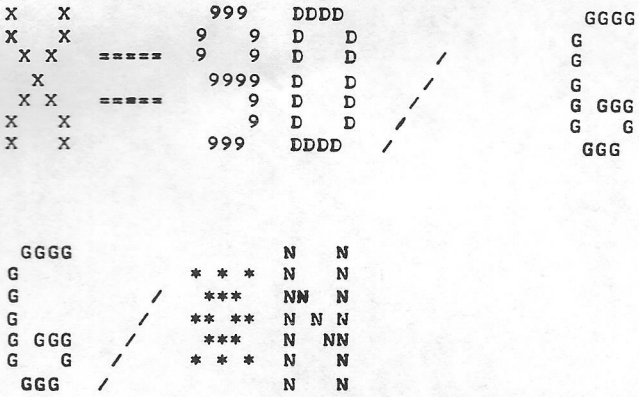
Used often with (/) is the command (N), which leads to a *NEW LINE* in the WIZARD's message.

An amusing example of the use of print commands is the Xmas program specified in the following printout of the message screen:

N9X9X



QX



2.8 LOGIN/LOGOUT

The command QQ is the super question - the question that asks the WIZARD to ask a question.

The WIZARD replies asking who he working for. If the name punched in is new to the WIZARD, a cheery welcome is given. If the name punched in reply is the name of someone who had earlier instructed the WIZARD to super remember (see below) what he had called V, W, X, and Y, then the WIZARD responds in a cheery way. Here are typical print-outs for the two cases:

NEW CHUM:

```
QQ
HELLO THERE.
WHAT IS YOUR NAME
 CINER
HOW DO YOU DO CINER
I AM THE WIZARD OF OZ
I LOVE TO MEET NEW PEOPLE LIKE YOU CINER
```

OLD CHUM:

```
QQ
HELLO THERE.
WHAT IS YOUR NAME
 HARVEY
WELCOME BACK HARVEY
THE WIZARD REMEMBERS WHAT YOU CALLED V,W,X, AND Y
```

Now during a session of OZ a child will write various programs using the remember feature. In other words, he will use the labels (V) (W) (X) and (Y) as names for various command strings which he has himself written. The child wants these same programs to be remembered long term - to be available at the next or a later session. To do so, he calls upon the WIZARD to Super-remember by punching in (Z) (Z) (*) .

There are two possibilities:

(a) The child has previously punched QQ.

```
ZZ

O.K. HARVEY
I WONT FORGET WHAT YOUR V,W,X,Y STAND FOR .....WIZARD
```

(b) If the child punches (Z) (Z) (*) , but is now known to the WIZARD, because he hadn't previously punched (Q) (Q) (*) , the WIZARD first asks "WHAT IS YOUR NAME" and waits for a reply before logging out.

After logout, the system still runs, and the last defined (V)(W)(X)(Y) are still there. But the WIZARD no longer knows the name of the operator.

2.9 The Abort Feature

A string of commands in OZ is always terminated by the *DO IT* command. Once the *DO IT* button is pressed, the commands in the string are carried out in sequence, and the control characters sent by the WIZARD to ZONKY appear one by one on the message screen, with time gaps between the appearance of control characters of up to one second. If during this process the *DO IT* button is punched, the control characters appear in a continuous stream, and the robot ceases to perform.

2.10 Etcetera

Numbers and modifiers may be jumbled together in any order ahead of the command that they modify.

Integers are limited to be less than ten. If this rule is overlooked only the number furthest to the right is actually valid. Thus (8)(9) is interpreted as (9). Likewise for fixed point numbers, (6)(7)(·)(5)(4) is read as (7)(·)(4). If no number is present the kindly WIZARD assumes (1) was intended.

There is a limit of 20 to the number of executions of a recursively defined command. And one is gently reproofed by the WIZARD if one attempts to call a new command string by other than V, W, X or Y.

On the modifier column of the OZ keyboard are two unimplemented modifiers, (O) and (K). These two symbols, as well as (M), (E), (U), (I) and the blank symbol (◯) are all ignored by the WIZARD, unless preceded by the print command (/).

3. EDUCATIONAL ASPECTS

This paper is necessarily a first introduction to OZ. We offer here only the briefest of pointers to the way OZ may be used in the primary and junior secondary classroom to inculcate basic mathematical ideas. However incidentally OZ serves to introduce the child to the world of computers. So what does a child learn about computing when exposed to OZ?

3.1 "Little Men".

In order to explain to children the ideas of recursive calls to procedures in LOGO, Papert et al (Feurzeg et al 1969) introduced the notion of "little men" as being responsible for each call of a procedure, the flow of control in a program being pictured in terms of the passing of messages from one "little man" to another.

A group at XEROX Pao Alto Research Center (Alan Kay 1974) have recently developed a rather elaborate robotics/graphic system called SMALLTALK. In SMALLTALK the idea of message passing is paramount, and in writing a program one explicitly defines "little men" in terms of what sort of messages they receive and what the "little man" should do with messages. In SMALLTALK the "little men" are oganized into classes following the ideas present in SIMULA (Birtwistle et al 1974).

3.2 Actors.

Carl Hewitt (1975) and his co-workers at M.I.T. have adopted the "little men" idea to explicate the semantics of computer programming. Hewitt terms the "little man" an ACTOR. An ACTOR is precisely defined as an entity with an internal state, which can receive and transmit messages. Now all programming features (and data structures) may be interpreted in terms of ACTORS; however making explicit use of ACTOR building blocks means that one has control over significant program features otherwise hidden (in a compiler or evaluator). This consideration has lead Hewitt to the design of the artificial intelligence language PLANNER 73.

The same ACTOR concept was earlier and independently developed at Hewlett-Packard Electronic Research Laboratory (C. Clare 1973) for purposes of the design of microprocessor logic devices: what Hewitt calls an ACTOR Clare terms "The General State Machine Module".

3.3 OZ Semantics.

On the message screen of the OZ system appear messages to and from the child (user) to the (computer) WIZARD, together with the control code messages from WIZARD to ZONKY. Thus message passing is a focus of attention in OZ. The WIZARD is an ACTOR who performs in public!

On the smaller scale in OZ there are four "variable commands" - which suggestively precede the column (M) (E) (N) on the OZ keyboard. In the absence of the modifiers (G) and (D) these commands are highly modular procedures. However in interpreting the command (X) defined by (Z) (X) (G) (F) (*) one must first read (3) (X) as (X) (X) (X) Then each time the "little man" (X) is woken up he takes note - i.e. changes his internal state. The Little Men of OZ have memories - as does the WIZARD himself!

3.4 Conclusion and Speculation.

We know from the work of Piaget (1947) that a child acquires topological ideas before metrical ideas, that topological ideas are in fact more intuitive, yet it took mathematics three thousand years to "advance" from Euclidean geometry to topology. And likewise in computing languages - in their very short history. It seems that only now as we attempt to write programming languages for children are we learning to describe the truly basic and elemental ideas of computational mathematics.

Despite the elementary character of OZ the concrete representation of message passing together with the modular features demonstrate that OZ actually embodies the basics of computing in accord with the best current thinking.

Most efforts to teach school children computer languages have involved teaching subsets of the more old fashioned programming languages. In contrast OZ, although designed as a language for "mere" children, probably has much more in common with the languages and computing styles of tomorrow.

ACKNOWLEDGMENTS

I would like to thank Seymour Papert for the hospitality of the LOGO group at M.I.T. during 1974, which provided that stimulus which lead to OZ. The OZ robot ZONKY described herein was designed in conjunction with Mr. Andrew Downing. The implementation of OZ on La Trobe's DEC-10 computer was written largely in Algol-60 together with various DEC-10 Macros: Mr. Ian Griffiths aided this work by reporting to me his research on the undocumented features of the DECUS Algol compiler.

REFERENCES

Abelson, H., Goodman, N., and Rudolf, L. (1974): "LOGO Manual" LOGO Memo No.7, Artificial Intelligence Laboratory, Massachussetts Institute of Technology.

Birtwistle et al (1974): "SIMULA Begin", Petrocelli Books, New York.

Clare, C.R. (1973): "Designing Logic Systems Using State Machines", McGraw Hill Inc., San Francisco.

Cohen, H.A. (1976): "The OZNAKI Project: Mini-Robots for Mini-Mathematicians" OZ Working Paper 6, La Trobe University, Bundoora, Victoria.

Dienes, Z.P. (1964): "Mathematics in the Primary School", Macmillan, Melbourne.

Eastwood, L.F. and Ballard, R.J. (1975): "The Plato IV CAI System", J. Educational Technology Systems, Vol.3, p.267.

Feurzeig, W., Papert, S., et al (1969): "Programming Languages as a Conceptual Framework for Teaching Mathematics", Report No.1899, Bolt Beraneck and Newman Inc., Cambridge, Mass.

Hewitt, C. (1975): "How to Use What You Know", p.189, Advance Papers of the Fourth International Joint Conference on Artificial Intelligence, Tbilisi, U.S.S.R.

Hewitt, C. and Smith, B. (1975): "Towards a Programming Apprentice", IEEE Journal of Software Engineering, March.

Kay, A. (1974): "SMALLTALK, A communication medium for children of all ages", Xerox Pao Alto Research Center, Pao Alto, California.

Papert, S. (1971): "Teaching Children to be Mathematicians Versus Teaching About Mathematics", LOGO Memo No.4, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Perlman, R. (1974): "TORTIS: Toddler's Own Recursive Turtle Interpreter System", LOGO Memo No.9, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Piaget, J. (1970): "Genetic Epistemology", Norton Publishing, New York.