SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD AND APPLETON LABORATORY
COMPUTING DIVISION

Starlink Project
Starlink User Note 17.1

09.06.81

The IDL Language

'IDL' stands for 'Interactive Data Language'. It is a commercially produced piece of software for analysing data consisting of arrays of any number of dimensions. It was purchased from Research Systems, Inc., of Denver, Colorado, mainly for a group of users working on data from the SMM satellite. It is not Starlink supported software and this information is provided for users who may wish to attempt to use it on their own initiative without support. The best source of advice for new users is other IDL users. Problems should be reported to Jeff Payne at RAL (Abingdon 21900 X6404), or to Mike Lawden (X6234). We are unable to modify IDL or correct any bugs. All problems will be passed back to the author in the USA.

The language is described in a manual produced by Research Systems, Inc.. A copy of the latest version of this manual is stored on line in file SYS$SYSDISK:[STARPACK.IDL]IDL.LIS, however, this does not contain the diagrams and pictures in the original manual.

Data input to the system must be in the form of DEC Files-11 format files of data arrays. The user must get his data into this form before attempting to use IDL. One method of doing this is to use program IDLP1 described in SUN/18. Information on how to input data is given in section 'OPENR' on page 54 and section 'ASSOC' on page 26 of the latest manual.

There is a test image in file [STARPACK.IDL]IMAGE.DAT consisting of a 20x20 array of 2-byte signed integers. This can be used in the following example test of the IDL system.

Having logged in, set the terminal in upper case mode by typing the command

```
        SET TERM/UPPER
```

IDL will not work properly if you type in IDL commands in lower case. Now start up the IDL system by typing the command

```
IDL
```

This command is a global symbol that is defined when you login. The IDL system will display the following information:

```
IDL     STARLINK V2.1 (C) 1981, RESEARCH SYSTEMS, INC.
300     DISC BLOCKS
700     BYTES OF DYNAMIC MEMORY.
16484   BYTES AVAILABLE FOR LARGEST VARIABLE.


IDL>
```

IDL will prompt you for commands with the prompt "IDL>". This prompt will be omitted from now on. The first thing to do is to open the test file for input. This is done by the IDL command:

```
OPENR,1,'SYS$SYSDISK:[STARPACK.IDL]IMAGE.DAT'
```

The second field, '1', is a 'logical unit number' which identifies the file within IDL. The structure of the data must then be specified by the command

```
ARRAY=INTARR(20)
```

Lines of the array can now be read using the 'FORRD' command. Thus the command

```
FORRD,1,ARRAY
```

will read the first line of the array stored in file SYS$SYSDISK:[STARPACK.IDL]IMAGE.DAT. Note that the ASSOC command, which is the normal I/O mechanism in IDL does not work with data produced by program IDLP1. The FORRD command must be used instead since this command is specifically designed to read files produced by the FORTRAN unformatted binary WRITE statement whereas ASSOC makes use of an internal IDL format. Now that the first line has been read, it can be displayed on an ordinary terminal by the command

```
FOR I=0,19 DO PRINT,ARRAY(I)
```

This command should cause the number sequence 0, 1, ... 19 to appear on your terminal. Notice that array elements are indexed from 0 upwards rather than from 1 upwards, so the first element is ARRAY(0). Notice also that you must have a comma between 'PRINT' and 'ARRAY'. The next line of the array can be read in by the command

```
FORRD,1,ARRAY
```

and the largest element in this line can be displayed by the command

```
    I=MAX(ARRAY) & PRINT,I
```

In this line, two commands have been written, separated by the character '&'. IDL should display the answer '20' in reply. You can now return from the IDL system to your original environment by typing

```
    (CNTRL)/Z
```

The example above is simply an elementary test to demonstrate that IDL is working. It can be carried out from any interactive terminal. However, the real power of IDL is that it enables you to program algorithms which operate directly on arrays and to display results graphically (for which you need a graphics terminal). In order to learn to use IDL properly you must study the IDL manual.

The following is a simple example of how IDL may be used to create a vector containing a sampled sine curve and plot it on the VDU screen.

```
    A=FLTARR(128)
    FOR I=0,127 DO A(I)=SIN(I/15.0)
    PLOT,A
```

The first statement issued causes identifier 'A' to be defined as a one-dimensional floating point array with 128 elements preset to zero. The next statement assigns the elements of A to sample values from a sine curve. The last statement causes the values held in array 'A' to be plotted on the VDU screen. Either a Tektronix 4010 or a terminal which emulates 4010 mode (graphics Lear Siegler or Sigma GOC set up appropriately) must be used if the graphics capability of IDL is to be exploited.

The version of IDL installed on the Chilton VAX is a PDP-11 version running in compatibility mode. The biggest limitation of this version is that the largest data array that can be manipulated is one of 16484 bytes. This allows manipulation of 128x128 byte arrays or 64x64 floating point arrays. Thus, IDL is not suitable for processing large images such as 768x768 byte IUE images or Landsat images. A VAX native mode version is currently in preparation but is not expected to be available for purchase for about a year.

The latest version (2.1) installed on Starlink computers contains the following updates on the version originally installed.

1. The SPAWN command

    This command should work when version 2.3 of VMS is released, probably in autumn '81.

2. Procedures

   The maximum number of user procedures allowed has been increased to 48.

3. The CURSOR command

   This has an optional third parameter to allow the user to set the number of characters expected on pressing carriage return. This is useful on SIGMA terminals.

4. !HI & !LOW

   These are now displayed as lines when !FLIP=0

5. EOF function has been corrected.

6. BLOCK STATEMENTS

   A block statement is simply one or more statements started by a BEGIN identifier and ended by an END identifier. As an aid to the proper nesting of compound statements the "END" terminating the block may be followed by the type of the statement in which the block resides.

   The following forms of END are optional:

ENDELS	-	IF statements, ELSE clause
ENDFOR	-	FOR statement
ENDIF	-	IF statement, THEN clause
ENDREP	-	REPEAT statement
ENDWHI	-	WHILE statement

7. CONVOL function

   Convolves a kernel with an array, returning the result as the value of the function.

8. HELPSY statement

   Prints the names and values of all the system variables on the terminal.

Some IDL users have requested an increase in:

a. The size of each program unit.

b. The number of files open simultaneously.

Increasing either of these would impose severe limitations on the size of the largest variable.

REFERENCES:

1. "IDL - Interactive Data Language". This is the standard reference source for the IDL system and is available from Research Systems Inc., 2021 Albion Street, Denver, CO 80207, USA.

2. SUN/18 "IDLP1 - Convert Starlink image frames to IDL format"