REVISION RECORD			
Rev	Description	Date	Author
A	Initial Release	06-Jan-2004	E. Wolfe

Contents

Tables

1. References

1) Gell, D. "Vector Requirements", SPRL File 055-4020.

2) Wolfe, E., "Vector Design and Maintenance", SPRL File 055-4274.

3) Gell, D. "Vector File Format", SPRL File 055-3933.

4) Gell, D. ,"File Naming Convention", SPRL File 055-3545.

5) Gell, D. "Profile File Format", SPRL File 055-3532.

2. Introduction

The purpose of this document is to assist the user in the running of the vector program, describe how to discern if the run was successful, and provide some measures that can be taken in the event of failure.

2.1 Purpose

The vector program is intended to operate on the data in a TIDI profile (level 2) file (see reference (5)) and produce a TIDI vector (level 3) file (see reference (3)) containing the profile data mapped onto a regularly spaced track angle grid.

2.2 Intended Audience

Those users responsible for generating TIDI data products. Some familiarity with the TIDI data flow scheme and a basic knowledge of Unix commands is needed.

3. Operation

For daily data production, the perl script runVector.pl is available for invoking vector. It can be given a date string in the form of yyyydoy instead of an input file name. It is able to write the output file to a location local to the machine being run on to avoid a problem with extremely slow write speeds over the current NFS connection, and then move it to the standard production directory. It also sends a PAN (product availability notice) upon successful completion of vector. The pan destination can be set to a test site. By default, it will print the full input/output filenames and the pan command, and then exit without doing anything. It also checks the exit status of vector, and logs a message on failure.

3.1 Required Files

The text "yyyy" below refers to the year in which the TIDI data was produced:

3.1.1 profile (.PRF) file

The input file containing profiles to be placed on a regularly spaced track angle grid. Located in /data/tidi/profile/yyyy

3.1.2 orbit-attitude (.aon) file

An input file containing "as flown" orbit information, for inclination and right ascension of the ascending node. Located in /data/timed/pvat/yyyy.

3.1.3 constant parameter (.CPF) file

An input file containing instrument model parameters and other constant values used throughout the data processing system. Vector uses values for π, Earth radius, and others. Located in /tidi/tidi_software/lib.

3.1.4 vector (.VEC) file

This is the output file containing profiles mapped to regularly spaced track angle grid. See reference (3) for full list of products. This will be created (or replaced) in /data/tidi/vector/yyyy.

3.2 Vector Execution Command and Switches

To execute vector, one may use the runVector.pl production script, or issue the vector command directly on the command line. The runVector.pl script does not allow the specification of all of the control options that vector accepts. The script also posts all messages to the stdout device. To use vector most generally, capturing all messages in a log file, you use the following command appropriate to the shell in use.

vector –i somefile.PRF [other switches and arguments] > logfile.log 2>&1 (from sh)

vector –i somefile.PRF [other switches and arguments] >& logfile.log (from csh)

The input filename must be specified, all of the other arguments are optional. Table 1 below lists the available options and their effect. Options are case-sensitive, must be preceded by a hyphen, "-", and followed by a space if they take an argument or there are other options.

Table 1: Vector Command Line Options

Option	*Argument*	*Effect*
h	none	display available options to stdout, then exit
i	input filename	a profile file is required for vector to begin
o	output filename	specifies the name, path, or full path and name
p	none	enable writing to production area
k	none	permits overwriting existing output file
t	none	show full filenames then exit
st	<hh:mm:ss>	time to start processing in current day
et	<hh:mm:ss>	time to stop processing in current day
sp	degrees	set output map spacing. default: 8.0 deg
d	integer power of 2	cause various debug printing messages
a	kilometers	specify an altitude (km) when -d is set: default: 90.

Notes:

The input file must have <.PRF> extension. If there is no path information given (does not contain a </>), the standard TIDI path /data/tidi/profile/year/ will be formed from the filename.

If the output filename option is not given, or has a trailing </>, the output filename will be formed according to the TIDI conventions specified in reference 3. The full path and name can be specified – this is useful in a debugging or validation context.

If the production option is not specified, the output file will be written to the <current directory>/year, unless the output file option contains path information.

The –k (kill) option is used when it is not desired to produce a file with an incremented revision number, useful when there was a program failure or in a debugging context. This has effect whether or not the production option was specified.

The –st and –et (start and end times for this day in 24 hour hh:mm:ss format, for example 13:10:45) are useful during development and debugging or tracking down the reason for a program failure – it limits the amount of messaging and the time it takes to run vector. It is not necessary to specify both. These are converted to mission time and record collection does not begin until the start time is met or exceeded, but the end time will not abort processing of a set of records already collected. For example, for a start time of 13:00:00 and an end time of 13:00:10, processing will collect all records after 13:00:00 until the scan table changes, then no more will be processed if the end time is exceeded.

The –d (debug) switch is used to control which messages in addition to the usual informational messages are written. The value given should be the sum of the powers of 2 given in `Table 2` below.

Table 2: Values for the Debug Option

Value	*Type of Message or variable name*
1	indexing values
2	track angles & map track angles
4	u and v
8	t_doppler
16	t_rot
32	ver2
64	back2

3.3 runVector.pl Command Line and Options

The command line to invoke the runVector.pl script is

/tidi/tidi_software/productionCtrl/runVector.pl –doy yyyydoy -nodebug

or

/tidi/tidi_software/productionCtrl/runVector.pl –i inputfile -nodebug

Like vector, the input filename or the day of year must be specified. Table 3 Below lists other available options. Options must be preceded by a single or double hyphen, "-", followed by a space unless it is the last thing on the command line, but are case-insensitive.

Table 3: runVector.pl Options

Option	*Argument*	*Effect*
help	none	show available options, then exit
(no)debug	none	show full in/out filenames, pan command, then exit
doy	yyyydoy	year, day of year to process
infile	input filename	full path optional

Table 3: runVector.pl Options

Option	*Argument*	*Effect*
opath	output filename	full path and name or path only ending with /
production	none	force writing to /data/tidi/vector/yyyy
kill	none	replace existing output file
(no)pan	none	send the PAN
ftpdest	string	ftp destination for the PAN

Notes: doy and infile switches are alternate ways of indicating what profile file to process. If both are given, infile is used. The doy switch is useful for running vector from a driver script with an integer loop counter when, for example it is necessary to process more than one file per day. It also runs findTididata to use the highest revision profile file available.

This script assumes the output should be written in the production area and always sends the –p switch to vector to cause writing there. This can be avoided by specifying an opath argument that contains path information. Note that the production switch here has a different meaning than in vector: here it means do not use a temporary local disk for the output file. Until the underlying problem with slow writing to the disk array currently in use (tidi06) is solved, use of this switch will dramatically increase the run time of any program that writes there. Once the problem is resolved, use of this switch will allow writing directly to the disk array.

The pan switch need not be given, as PANs are sent unless –nopan is specified, however it may be desirable for scripts to set this just to document that sending of a PAN is desired. The ftpdest also is not needed, the sendpan script sends to APL by default. Issue the command /tidi/tidi_software/productionCntrl/sendpan.pl -h for test destination choices and other options for sending PANs.

4. Determining Success or Failure

Vector sets an exit status with a value from 0 to 4, according to the standard TIDI convention. Errors and informational messages are written to stdout, which should be captured in a file and examined for unusual behavior. Occasionally, the program may simply crash and the operating system will issue a cryptic Unix message to stderr. If vector is run from a production script, this message will be captured in the log file for that script.

4.1 Errors and Recovery

Usually, an error message will appear in the logfile and give some indication of what went wrong, such as an empty input file or one that does not have all of the expected variables. Rerun vector when a profile file has been generated. File protection mode errors can occur when the user does not have privilege to read or write a directory. Rerun vector when proper privilege is established. Insufficient disk space can also cause failure: This will usually require someone with system privilege to correct.

If the problem cannot be corrected by examination of the log messages and the output data, the judicious selection of debugging switches should be able to isolate it. This can be a tedious process, and may require the expertise of a programmer familiar with the TIDI data processing scheme to locate and correct.