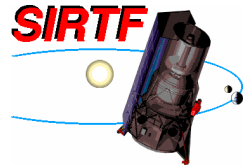


674-SO-43, Version 1.0,
SSC-PD-4059



SIRTF Science Center

Downlink Segment

Subsystem Design Specification

AOT Products Subsystem:
DELTACOMBINE

10 September 2001

California Institute of Technology
SIRTF Science Center



National Aeronautics and
Space Administration



Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

**THIS IS A PRELIMINARY DOCUMENT, the module described here may
or may not be utilized in the final pipelines as described.**

THIS PAGE IS INTENTIONALLY LEFT BLANK

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

1 Revision History

Version	Description	Date
1.0	Initial version	September 10, 2001

2 Table of Contents

1 REVISION HISTORY.....	IV
2 TABLE OF CONTENTS	V
3 LIST OF FIGURES	VII
4 LIST OF TABLES	VIII
1. INTRODUCTION.....	9
1.1. Purpose and Scope.....	9
1.2. Document Organization.....	9
1.3. Relationship to Other Documents	9
1.4. Change Procedure	10
2. OVERVIEW.....	10
2.1. DELTACOMBINE Requirements	10
2.2. Applicable Documents	11
2.3. Version History.....	11
2.3.1. Version 1.0.....	11
2.4. Liens. 11	
3. INPUT	11
3.1. DELTACOMBINE Input	11
3.1.1. DELTACOMBINE NAMELIST Input	12
3.1.2. DELTACOMBINE Command-Line Input.....	13
3.1.3. DELTACOMBINE FITS Input.....	13
4. PROCESSING.....	14
4.1. DELTACOMBINE Processing.....	14
4.2 DELTACOMBINE Processing Phases	15
4.1.1. DELTACOMBINE Initialization.....	15

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

4.1.2. Delta-File Table Inputs	16
4.1.3. Combination Algorithm.....	19
4.1.4. Output Delta-File.....	20
4.1.5. Termination.....	20
5. ALGORITHM SPECIFICS	20
6. OUTPUT	20
6.1. DELTACOMBINE Output	20
6.2. DELTACOMBINE Log-File Output	21
7. TESTING	21
8. USAGE EXAMPLES	21
9. GLOSSARY	22

List of Figures

Figure 1. DELTACOMBINE data and processing flow.....15

List of Tables

Table 1. Namelist File	12
Table 2. Command Line Options	13

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

1. Introduction

1.1. Purpose and Scope

The Subsystem Design Specification is a document that describes the basic requirements, assumptions, definitions, software-design details and necessary interfaces for each subsystem. The document will be used to trace the incremental development of each subsystem and also to allow trace-back of levied requirements; this document should have sufficient detail to allow future modification or maintenance of the software by developers other than the original developers. This document is an evolving document as changes may occur in the course of science instrument hardware design and maturity of operational procedures. This document is not intended to repeat sections or chapters from other Project documents; when appropriate, references to proper sections of primary reference documents will be made.

1.2. Document Organization

This document is organized along the major themes of Requirements; Assumptions; Operational Concept; Functional Descriptions; Functional Dependencies; Input; Output; Other S/S Interfaces; Algorithm Descriptions (when applicable); and Major Liens.

The material contained in this document represent the current understanding of the capabilities of the major SIRTf systems. Areas that require further analysis are noted by TBD (To Be Determined) or TBR (To Be Resolved). TBD indicates missing data that are not yet available. TBR indicates preliminary data that are not firmly established and are subject to change.

1.3. Relationship to Other Documents

The requirements on the operation of SIRTf flow down from the Science Requirements Document (674-SN-100) and the Facility Requirements Document (674-FE-100). The Science Operations System is governed by the SOS Requirements Document (674-SO-100). The current document is also cognizant of the requirements that appear in the Observatory Performance and Interface Control Document (674-SEIT-100) as well as the Flight Ground Interface Control Document (674-FE-101). This document is also affected by the FOS/SOS Interface Control Document (674-FE-102) that governs interfaces between the Flight Operations System and the Science Operations System. Related Software Interface Specifications (SIS) will be as indicated in Section 2.2 of this document.

1.4. Change Procedure

This document is a level 4 document according to the SIRTf Project Documentation Plan (674-FE-103). Changes to this document after approval require the approval of the SOS Change Board (TBD). The process for change control is described in the SOS Configuration Management Plan.

2. Overview

DELTACOMBINE reads data from two sets of delta-files and applies an optimisation algorithm to combine the delta-values from each to write a new improved delta-file. The input files are in IPAC table format. There are two user-specified options for the combining: the delta-values can be combined by either taking an (inverse variance) weighted mean, or, by selecting the lowest uncertainty corresponding to the delta-values from the two input tables. The input delta-files are initially generated by the “DELTAPOINT” and “DELTAFTCORR” software modules. These represent respectively the delta-file generated from “pointing header keywords” and the delta-file computed from “point-source correlation”. DELTACOMBINE is written in standard C.

2.1. DELTACOMBINE Requirements

DELTACOMBINE is initiated by a startup script under the control of the pipeline executive and does its required functions for a given DCE image or pre-processed DCE image; this involves performing the following tasks.

- A.) Retrieve the command line parameters passed by the start up script and use them to run the program.
- B.) Read in as input two delta-files in IPAC table format.
- C.) Produce as primary output a new delta-file.
- D.) Provide exit codes to the pipeline executive and also provides logon and logoff messages identifying the version number and write any error messages to the standard output devices.
- E.) Produce a processing summary.

2.2. Applicable Documents

The following documents are relevant to the DELTACOMBINE program of the AOT PRODUCTS Subsystems.

- A.) The SOS Requirements Document
- B.) The SOS Downlink Requirements Document
- C.) The SOS Downlink Software Development Guidelines

2.3. Version History

2.3.1. Version 1.0

Initial version created on September 10, 2001.

2.4. Liens

No major liens have been identified.

3. Input

3.1. DELTACOMBINE Input

DELTACOMBINE takes all of its input from either the command line or namelist file, which is set up by the startup script that is controlled by the pipeline executive or standalone. If the namelist is not specified, then all required inputs are expected from the command line. If both namelist and command-line inputs are specified, then the command-line inputs override the namelist values. Prior to reading namelist and/or command-line parameters, default values for the relevant parameters are assigned.

3.1.1. DELTACOMBINE NAMELIST Input

DELTACOMBINE reads the NAMELIST file whose name is passed to it by start-up script. The name of the NAMELIST is DELTACMBIN. The parameters that can be defined in the NAMELIST are listed in Table 1.

Namelist variable	Description	Dim.	Type	Units	Default
Delta_File_from_pointing	Required filename of delta-file in IPAC table format derived from pointing keywords	1024	C	-	Null
Delta_File_from_pnt_source_cor rel	Required filename of delta-file in IPAC table format derived from point-source correlation	1024	C	-	Null
Data_Out_Filename	Required filename of new delta-file.	1024	C	-	Null
Log_Filename	Optional output log filename	1024	C	-	stdout
Combine_Flag	Option for combining delta-files: 1 = weighted mean, 2 = lowest uncertainty.	1	I*1		1
Ancillary_File_Path	Pathname where supporting source files are installed.	1024	C	-	./ (current directory)

Table 1. Namelist file

The following is an example of the contents of a DELTACMBIN NAMELIST file that might be used, where the values specified are not necessarily realistic.

```
&DELTACMBIN
```

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

```
Comment = 'Generic namelist file for deltacombine, default values.',  
Ancillary_File_Path = '../deltacombine_v1',  
Delta_File_from_pointing = './testing/pointing.tbl',  
Delta_File_from_pnt_source_correl = './testing/pointcorr.tbl',  
Comment = '1 = Weighted mean, 2 = Select lowest uncertainty, Default=1',  
Combine_Flag = 1,  
Data_Out_Filename = './testing/combined.tbl',  
Log_Filename = 'stdout',  
&END
```

3.1.2. DELTACOMBINE Command-Line Input

Alternatively, all inputs can be specified via command line, in which case, a namelist file is not needed. Or, inputs can be provided with a hybrid of both namelist and command-line mechanisms, with the latter overriding the former. Table 2 lists the available command-line options associated with their namelist-variable counterparts, as well as other options for specifying the namelist-file name and making the standard output more verbose.

3.1.3. DELTACOMBINE IPAC Table Input

DELTACOMBINE uses the “tbl” IPAC table library routines to read the formatted input delta files. The routines used are tbl_open, tbl_keyinfo, tbl_column, tbl_colval and tbl_close.

Command-line option	Variable
-n	Namelist_Filename
-i1	Delta_File_from_pointing
-i2	Delta_File_from_pnt_source_correl
-o	Data_Out_Filename

-l	Log_FileName
-c	Combine_Flag
-a	Ancillary_File_Path
-v (verbose switch)	-
-vv (super-verbose switch)	-

Table 2. Command-line options

4. Processing

4.1. DELTACOMBINE Processing

DELTACOMBINE begins processing by writing its name and version number to standard output (verbose mode only), and then it initializes relevant variables with default values, and checks that the required namelist parameters and/or command-line parameters were passed to it. If this condition is not true, then it writes a message stating which parameters are missing, recommends a look at this document, and terminates by issuing an appropriate exit code to the pipeline executive; otherwise it proceeds as follows.

If an error occurs during processing, then an error message is written to standard output, a termination-status code is written to the log file, and an exit code to the pipeline executive issued.

After processing, the program name and version number, namelist filename (if used), input, and output filenames, values of other input parameters, date and time, processing time, and a termination-status code are written a log file.

4.2 DELTACOMBINE Processing Phases

DELTACOMBINE operates in five phases: initialization, delta-file table inputs, combining algorithm, results output and termination. This processing level is depicted in Figure 1.

4.1.1. DELTACOMBINE Initialization

DELTACOMBINE initializes itself by performing the following tasks.

- A.) A message is printed to STDOUT (verbose mode only), which includes the program name and version number.
- B.) If specified on the command line, the NAMELIST file is opened and read. If any errors are encountered, a message is printed, and execution aborts.
- C.) The remaining command-line inputs are read and checked for correct data range, consistency, etc. If any errors are encountered, a message is printed, and execution aborts.

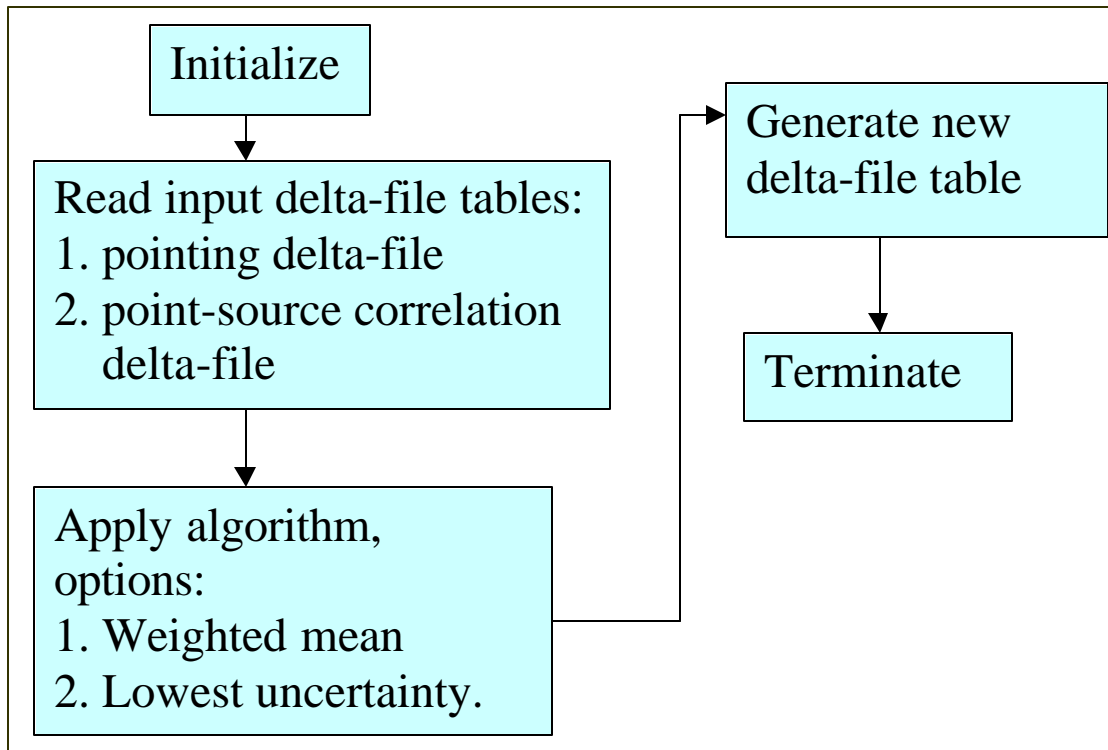


Figure 1. DELTACOMBINE data and processing flow

4.1.2. Delta-File Table Inputs

The two delta-file tables (namelist parameters: `Delta_File_from_pointing` and `Delta_File_from_pnt_source_correl`) are read as standard IPAC tables using the “tbl” IPAC table library and stored in memory. An example input table is shown on the following page.

```

\character Delta_File_Program = Output from deltafp, version 2.00
\character Creation_Date_Time = Tue Jul 24 18:44:53 2001
\character Input_Image_List = ./testing/deltafp_images.list3
\character Reference_Image = ./testing/sws4322a00205.fits
\integer Number_of_Frames = 9
\real RefFrCRVAL1 = 82.218150
\real RefFrCRVAL2 = 35.836323
\real RefFrCRPIX1 = 64.5
\real RefFrCRPIX2 = 64.5
\real RefFrCROTA2 = -0.092466
\real RefFrCDELT1 = -0.004323
\real RefFrCDELT2 = 0.004297
\real Input_Rotation_Offset = 0.000000
|Index |Filename |Xshift |Yshift |Rotation |err_Xshift |err_Yshift |err_Rot
|Xcenter |Ycenter | | | | | | |
|i |c |r |r |r |r |r |r
64.5 64.5 1 sws4322a00201.fits -2.488 -5.856 0.00030 0.136 0.137 0.17375
64.5 64.5 2 sws4322a00202.fits -5.731 -11.004 0.00325 0.136 0.137 0.17375
64.5 64.5 3 sws4322a00203.fits -2.399 -1.482 0.01365 0.136 0.137 0.17375
64.5 64.5 4 sws4322a00204.fits -2.559 16.615 0.00924 0.136 0.137 0.17375
64.5 64.5 5 sws4322a00205.fits 0.000 0.000 0.00000 0.000 0.000 0.00000
64.5 64.5 6 sws4322a00206.fits -24.193 4.841 0.01471 0.136 0.137 0.17375
64.5 64.5 7 sws4322a00207.fits 3.275 -5.591 0.00829 0.136 0.137 0.17375
.. -

```

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

THIS IS A PRELIMINARY DOCUMENT, the module described here may or may not be utilized in the final pipelines as described.

4.1.3. Combination Algorithm

Each delta-file is initially (and separately) derived from a different algorithm as implemented by the “deltafpoint” and “deltafptcorr” software modules. The latter software is more robust, however, depending on pointing accuracy, the former could outperform the latter. Furthermore, it is not guaranteed that “deltafptcorr” will produce a set of delta-values (translational and rotational offsets) for every image in a coadd or mosaic since it requires the existence of overlapping point sources between images. On the other hand, “deltafpoint” will always compute delta-values for every image provided pointing information exists. Therefore, we need a scheme to combine these two delta-files generated by different methods to compute an overall “robust” delta-file.

The first method (specified by namelist parameter: `Combine_Flag = 1` and is the default method) consists of computing an inverse-variance weighted mean between sets of delta-values from each input table. If the quantities X_1 and X_2 represent either of the three delta-values: θ (rotation), X_T (translation in X) and Y_T (translation in Y) from each delta-file with corresponding uncertainties $\sigma(X_1)$ and $\sigma(X_2)$, then their mean is defined:

$$\langle X \rangle = \frac{w_1 X_1 + w_2 X_2}{w_1 + w_2},$$

$$\text{with error } \mathbf{s}\langle X \rangle = \frac{1}{\sqrt{w_1 + w_2}}$$

$$\text{where } w_1 = \frac{1}{\mathbf{s}^2(X_1)}, \quad w_2 = \frac{1}{\mathbf{s}^2(X_2)}.$$

The second method (`Combine_Flag = 2`) consists of selecting the set of delta-values between the two files which have the lowest uncertainties. These delta-values are then written to the output delta-file.

Furthermore, if any entries in the “Delta_File_from_pnt_source_correl” file are missing due to lack of common point sources between images, then the values from the “Delta_File_from_pointing” file are used to replace them in the final output delta-file.

4.1.4. Output Delta-File

The delta-values from the two input delta-files are reduced to a single set of values for each image using the above algorithm and written to a new delta-file in IPAC table format (see example in section 4.1.2).

4.1.5. Termination

Summary output is appended to the log file (the log file is created if previously non-existent), which includes diagnostic reports for the Q/A Subsystem and the appropriate exit code issued to be picked up by the pipeline executive. A detailed list of log file contents is given in Section 6.1.2.

5. Algorithm Specifics

- A. DELTACOMBINE strictly assumes that each input delta-file is in the same format as that shown by the example given in section 4.1.2.
- B. The list of FITS image names in each input delta-file need not be in the same order. However, if no matches are found between the two files, the program will abort with an error message sent to standard output indicating that the tables are inconsistent.
- C. The maximum number of image names from each delta-file that can be read and stored in memory is currently set at 3000. This is specified by the parameter "MAX_NUM_IMAGES" in the include file: deltacombine.h.

6. Output

6.1. DELTACOMBINE Output

DELTACOMBINE is capable of generating the following output:

- A.) Standard-output processing and status messages.

B.) A table in IPAC format containing the new delta-values and a header similar to that of the input delta-files.

C.) A log file containing processing statistics, status messages and ancillary information.

All DELTACOMBINE disk output is written to the pathnames that are specified with the output filenames in the command-line or namelist inputs.

6.2. DELTACOMBINE Log-File Output

The information stored in the log file at the output of this program includes: program name and version number, values of all namelist and/or command-line inputs, a message indicating the type of calculation performed, status code, processing time, date and time, and a message indicating program termination.

7. Testing

DELTACOMBINE has been successfully unit-tested as a stand-alone program for a variety of different input cases. The tests were designed to check DELTACOMBINE robustness and capability of generating corrected results.

Here is a summary of the unit tests that were conducted:

1. Tested DELTACOMBINE on simulated delta-files containing over 2000 images.
2. Executed DELTACOMBINE with inputs read from and output written to directories different from where the program was run. Both namelist and command-line input mechanisms were exercised.
3. Executed DELTACOMBINE for all combinations of input parameters, in order to test that they function properly.

8. Usage Examples

Using a namelist file with the verbose (-v) output saved to a file "out.log":

```
DELTACOMBINE -n deltacombine.nl -v | & tee out.log
```

Without using a namelist file and using the “Weighted Mean” combine option (-c 1):

```
DELTACOMBINE -i1 table1.tbl -i2 table2.tbl -o new.tbl -a  
./ancpath -c 1 -v
```

9. Glossary

DCE	Data Collection Event
DN	Data Number
IOC	In-Orbit Checkout
SDS	Subsystem Design Specification
SIS	Software Interface Specification
TBD	To Be Determined
TBR	To Be Resolved