EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

VIRCAM Pipeline User Manual

ESO-299734

Issue 2.3.17

Date March 24, 2026

	Name	Date	Signature
Prepared:	ESO VCAM Pipeline Team	24/03/2026	
Approved:			
Released:			

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
2.3.0	01/04/2017	All	First version
2.3.1	06/04/2017	All	A. Gabasch: text edits and updates
2.3.2	06/01/2018	3	NEWS section added
2.3.4	01/05/2019	3	NEWS section updated
2.3.5	01/04/2020	3	NEWS section updated
2.3.6	20/05/2020	5.2	Color terms added
		3	NEWS section updated
2.3.8	20/05/2021	3	NEWS section updated
2.3.10	19/05/2022	3	NEWS section updated
2.3.12	15/05/2023	3	NEWS section updated
2.3.17	24/03/2026	3	NEWS section updated

This page was intentionally left blank

Contents

1 Introduction **9**

1.1 Purpose and Scope . . . 9

1.2 Acknowledgements . . . 9

1.3 Conventions on Style and Notation . . . 9

1.4 Reference documents . . . 9

2 Overview **11**

2.1 The VIRCAM Instrument . . . 11

2.2 The VIRCAM Data Reduction Pipeline . . . 11

3 What's new **13**

3.1 What's new in the latest pipeline release (2.3.17) . . . 13

3.2 What's new in pipeline release 2.3.12 . . . 13

3.3 What's new in pipeline release 2.3.10 . . . 13

3.4 What's new in pipeline release 2.3.9 . . . 13

3.5 What's new in pipeline release 2.3.8 . . . 13

3.6 What's new in pipeline release 2.3.6 . . . 13

3.7 What's new in pipeline release 2.3.5 . . . 13

3.8 What's new in pipeline release 2.3.4 . . . 14

4 Installation **15**

4.1 System Requirements . . . 15

4.2 Installing the Software . . . 15

4.3 EsoRex, Reflex, and Gasgano . . . 15

5 Data Description **17**

5.1 Raw Data . . . 17

5.2 Static Calibration Data . . . 17

5.3 Photometric and Astrometric Catalogues . . . 19

5.4 Pipeline Products . . . 20

5.4.1 Naming convention for files . . . 20

5.4.2 Images . . . 21
5.4.3 Catalogues . . . 21
5.4.4 Diagnostic tables . . . 21

6 Data Reduction **22**
6.1 Getting Started with EsoRex . . . 22
6.2 Data Organization, Classification, & Association . . . 23
6.2.1 Header Keywords . . . 23
6.2.2 Classification . . . 24
6.2.3 Organization . . . 27
6.2.4 Association . . . 27
6.3 Data Reduction Cascade . . . 28
6.4 Calibration . . . 30
6.4.1 Linearity . . . 30
6.4.2 Dark . . . 34
6.4.3 Flat Field . . . 35
6.4.4 Detector Noise [optional] . . . 38
6.5 Science Observations . . . 42
6.5.1 Creating a pawprint with vircam_science_process . . . 42
6.5.2 Creating a tile with vircam_science_postprocess . . . 44

7 Troubleshooting **49**
7.1 Log file . . . 49

8 Recipe Reference **50**
8.1 vircam_linearity_analyse . . . 50
8.1.1 Description . . . 50
8.1.2 Input frames . . . 50
8.1.3 Recipe parameters . . . 50
8.1.4 Product frames . . . 51
8.1.5 Quality control parameters . . . 51
8.2 vircam_dark_combine . . . 52

8.2.1 Description . . . 52
8.2.2 Input frames . . . 52
8.2.3 Recipe parameters . . . 52
8.2.4 Product frames . . . 53
8.2.5 Quality control parameters . . . 54
8.3 vircam_twilight_flat_combine . . . 54
8.3.1 Description . . . 54
8.3.2 Input frames . . . 54
8.3.3 Recipe parameters . . . 56
8.3.4 Product frames . . . 57
8.3.5 Quality control parameters . . . 58
8.4 vircam_detector_noise . . . 58
8.4.1 Description . . . 58
8.4.2 Input frames . . . 58
8.4.3 Recipe parameters . . . 58
8.4.4 Product frames . . . 59
8.4.5 Quality control parameters . . . 59
8.5 vircam_science_process . . . 59
8.5.1 Description . . . 59
8.5.2 Input frames . . . 59
8.5.3 Recipe parameters . . . 60
8.5.4 Product frames . . . 64
8.5.5 Quality control parameters . . . 65
8.6 vircam_science_postprocess . . . 66
8.6.1 Description . . . 66
8.6.2 Input frames . . . 66
8.6.3 Recipe parameters . . . 67
8.6.4 Product frames . . . 69
8.6.5 Quality control parameters . . . 70

9 Recipe Algorithms **72**

9.1 Stacking 72
9.2 Confidence maps 72
9.3 Sky Estimation 72
9.4 Object detection/catalogue generation 74
9.4.1 Background analysis and object detection 74
9.4.2 Object parameterisation 75
9.4.3 Morphological classification 76
9.5 Nebulising 77
9.6 Destriping 77
9.7 Grouting 79

A File Formats **80**
A.1 Static Calibration Files 80
A.2 Astrometric and Photometric Catalogue Files 81
A.3 Recipe Product Catalogues 83

1 Introduction

1.1 Purpose and Scope

The VIRCAM pipeline is a subsystem of the *VLT Data Flow System* (DFS). It is used in two operational environments, for the ESO *Data Flow Operations* (DFO), and for the *Paranal Science Operations* (PSO). It is used to provide a quick-look assessment of data, to generate master calibration data, to reduce of scientific exposures, and to assessed data quality. Additionally, the VIRCAM pipeline recipes are made public to the user community, to allow a more personalised processing of the data from the instrument. The purpose of this document is to describe a typical VIRCAM data reduction sequence with the VIRCAM pipeline.

This manual is a complete description of the data reduction recipes implemented by the the VIRCAM pipeline, reflecting the status of the VIRCAM pipeline version 2.3.17.

Note that this pipeline is **not** a replacement for the operational QC pipeline that has been in place for several years (as described in [6]). This pipeline is partly based on that QC pipeline and provides additional recipes for users to create calibrated science data products.

1.2 Acknowledgements

The VIRCAM pipeline is based on the VISTA Data Reduction Software developed by Cambridge Astronomy Survey Unit at the University of Cambridge.

This release has benefitted from the feedback provided by ESO staff. In particular we would like to thank M. Neeser, L. Coccato, A. Gabasch, and C.E.G. Dabo for testing the pipeline and their suggestions for improvements.

1.3 Conventions on Style and Notation

This document uses the following conventions for font styles:

- **bold**: commands or other input to be typed as shown
- *italics*: placeholder of input to be replaced by real text as appropriate for particular use
- `teletype`: FITS keywords, program names, file paths, and terminal output, etc.

A generic shell prompt is used as a prefix to commands or other user input.

To improve readability, hierarchical FITS keyword names appear in ‘dot-notation’. This means that the prefix `"HIERARCH ESO"` is omitted and the spaces separating the keyword name in the FITS header are replaced by a single dot, e.g. `DET.DIT` refers to the keyword `"HIERARCH ESO DET DIT"`.

1.4 Reference documents

[1] R. M. Cutri, M. F. Skrutskie, S. van Dyk, C. A. Beichman, J. M. Carpenter, T. Chester, L. Cambresy, T. Evans, J. Fowler, J. Gizis, E. Howard, J. Huchra, T. Jarrett, E. L. Kopan, J. D. Kirkpatrick, R. M. Light, K. A. Marsh, H. McCallon, S. Schneider, R. Stiening, M. Sykes, M. Weinberg, W. A. Wheaton, S. Wheelock, and N. Zacarias. VizieR Online Data Catalog: 2MASS All-Sky Catalog of Point Sources (Cutri+ 2003). *VizieR Online Data Catalog*, 2246:0, June 2003. 20, 43

[2] ESO. *VIRCAM Reflex Tutorial*, 2.3.17edition, 2015. VIS-MAN-ESO-XXXXX-XXXX. 11, 16

[3] ESO/SDD/DFS, *http://www.eso.org/cpl/esorex.html*. *ESOREX home page*. 22

[4] ESO/SDD/DFS, *http://www.eso.org/gasgano/*. *Gasgano User's Manual*. VLT-PRO-ESO-19000-1932. 11

[5] M. J. Irwin. Automatic analysis of crowded fields. *MNRAS*, 214:575–604, June 1985. 74

[6] Lewis J., Bunclark P., and Hodgkin S. *VISTA Data Reduction Library Design*. ESO, 1.10 edition, 2008. VIS-SPE-IOA-20000-0010. 9

[7] D. J. Schlegel, D. P. Finkbeiner, and M. Davis. Maps of Dust Infrared Emission for Use in Estimation of Reddening and Cosmic Microwave Background Radiation Foregrounds. *Astrophysical Journal*, 500:525–553, June 1998. 19, 80

[8] Forchì V. *Reflex User's Manual*. ESO/SDD/DFS, 0.7 edition, 2012. VLT-MAN-ESO-19000-5037. 11, 16

2 Overview

2.1 The VIRCAM Instrument

The VISTA InfraRed Camera (VIRCAM) is a cryogenic, wide-field infrared imager installed on the 4-m class VISTA telescope. It comprises 16 Raytheon VIRGO 2048x2048 pix detectors. With a pixel size $\approx 0.34''$, each chip covers a $\approx 11'$ x $11'$ field of view. A typical observing sequence fills in the gaps between detectors to produce a tile covering $\approx 1.64\text{deg}^2$. Images can be taken in one of five broadband filters (Z, Y, J, H, K_S) or two narrowband filters (NB_{980}, NB_{118}).

Please refer to the instrument web page at `https://www.eso.org/sci/facilities/paranal/instruments/vircam/doc.html` for detailed information about the instrument.

2.2 The VIRCAM Data Reduction Pipeline

The VIRCAM data reduction pipeline is part of a suite of tools provided by the Pipeline Systems Group (PSG) of the Science Operation Software Department at ESO. In collaboration with instrument consortia, the PSD implements data reduction pipelines for three main purposes:

Master calibration product creation: pipelines are used to produce master calibration products (e.g., combined bias frames, master flats).

Data quality control: pipelines are used to produce the quantitative information necessary to monitor instrument performance.

Science product creation: using pipeline-generated master calibration products, science products are produced for the supported instrument modes (e.g., calibrated images; stack of jittered images). The VIRCAM pipeline aims to provide scientific grade data products.

Instrument pipelines consist of a set of data processing modules that can be executed in one of four ways: 1) from the automatic data management tools available on Paranal, 2) from the command line (using *EsoRex*), 3) from *Reflex*, or 4) from *Gasgano*. The two latter applications are included in the pipeline distribution (see [8] and [4]). These applications can also be downloaded separately from `http://www.eso.org/sci/software/cpl/esorex.html`, `http://www.eso.org/sci/software/reflex/`, and `http://www.eso.org/sci/software/gasgano.html`. An illustrated guide to processing VIRCAM data with Reflex is provided in [2].

Note that this pipeline cannot be used to process VIRCAM data taken in "microstepped" mode. Only normal, jittered images can be processed.

Also note that the pipeline **is not designed to be used to process science images that span more than one Observation Block**. Attempting to process data that span multiple OBs may yield unexpected and/or less than ideal results, particularly with respect to sky corrections.

The VIRCAM pipeline comprises 6 recipes: 4 recipes for calibration and 2 recipes for processing science images.

The four calibration recipes are:

vircam_linearity_analyse: create a bad pixel map and a table with values of the non-linearity for each detector by analysing a set of darks and lamp flats.

vircam_dark_combine: create a master dark image by combining raw dark images. The recipe can also quantitatively compare the master dark to a user-specified 'reference' dark.

vircam_twilight_flat_combine: create a master flat image and confidence map by combining raw twilight images. The recipe can also quantitatively compare the master flat to a user-specified 'reference' flat.

vircam_detector_noise: calculate the detector readout noise and gain from a set of raw dark and twilight flat images.

The two science recipes are:

vircam_science_process: create calibrated images and object catalogues by combining a number of jittered science images on a chip-by-chip basis, i.e. a "pawprint" or "stack".

vircam_science_postprocess: create one calibrated image and object catalogue from a set of jittered science images by combining images from all chips, i.e. a "tile". If requested, the recipe will "nebulise" the image when creating the object catalogue, i.e. remove the effect of strongly varying diffuse background emission.

More details on the pipeline and how to use it can be found later in this manual in §6.

3 What's new

3.1 What's new in the latest pipeline release (2.3.17)

- Pipeline updated to the latest CPL release (v7.3)

3.2 What's new in pipeline release 2.3.12

- Pipeline updated to the latest CPL release (v7.3)

3.3 What's new in pipeline release 2.3.10

- Improvements & bugfixes in the CASU library
- Clarify the GlobalReusePreviousExecution param on the Reflex workflow canvas

3.4 What's new in pipeline release 2.3.9

- Several improvements/fixes to the Reflex workflows
- Products use RADESYS instead of RADECSYS, and do not contain HDRVER
- Documentation now reflects new support channel & FTP server protocol (https), broken links fixed
- Pipeline updated to the latest CPL release (v7.2)
- Mac OS Catalina support added

3.5 What's new in pipeline release 2.3.8

- Pipeline updated to the latest CPL release (v7.1.4)

3.6 What's new in pipeline release 2.3.6

- Tutorial restructured to make use of common Reflex sections
- Color term details added to manual
- Reflex A&A citation and link added to workflow canvas

3.7 What's new in pipeline release 2.3.5

- Reflex python scripts have been ported to python 3 keeping the backward compatibility with python 2.

3.8 What's new in pipeline release 2.3.4

- The pipeline has been built on many different platforms with different compilers and the warnings reported by those compiler were fixed. In a few occasions, the calculation of the absolute value of a number was done by using the wrong c-function (`fabs()` vs. `abs()`). This has been fixed. As a consequence, some values in the catalogues will show subtle differences.

4 Installation

4.1 System Requirements

Due to the large number of files with short exposures, data reduction for infrared imaging may use a significant amount of memory. This can be exacerbated when the images are combined into a tile that covers a large part of the sky. The exact amount of memory required to run a recipe depends on a number of factors, e.g. the sky subtraction scheme, number of files, and sky coverage. Below is a table with the minimum resident memory needed to process data as a function of recipe and number of input science. This table assumes that all recipe parameters are set to their default and a small maximum jitter offset between science frames. The execution time of `vircam_science_process` can be shortened by choosing 'fast' stacking, but this will increase the memory requirements (see description of `stk_fast` and `stk_nfst` parameters in §9.1).

Recipe	# science frames	Min. RAM
vircam_science_process	12	5.5 GB
vircam_science_postprocess	6 (pawprints)	18.4 GB

Table 4.1: Minimum memory requirements for selected VIRCAM recipes

4.2 Installing the Software

The VIRCAM pipeline is distributed as a standard pipeline kit package and can be obtained from the ESO web pages at http://www.eso.org/sci/software/pipelines. In addition to the VIRCAM pipeline, the distributed package contains all dependencies needed for the installation, the tools to run the recipes, this manual, and the installer utility for the kit.

Using the installer, the VIRCAM pipeline can be installed on recent versions of any major Linux distribution, as well as Mac OS X.

To install the VIRCAM pipeline, unpack the kit in a temporary location, go to the top level directory of the unpacked distribution package and execute the installer script as shown in the following example:

Note: The installation script uses the compiler which is found first in the path. If more than one compiler is installed on the system, make sure that your preferred compiler will be found first when the installation script is executed.

```
1> tar -zxf vircam-kit-X.Y.Z.tar.gz
2> cd vircam-kit-X.Y.Z
3> ./install_pipeline
```

Then follow the instructions on the screen. Once the script finishes successfully and the path variables have been set, the installation is complete.

4.3 EsoRex, Reflex, and Gasgano

ESO offers three different tools to process data obtained with VIRCAM: one command line tool (*EsoRex*), and two GUI based tools (*Gasgano* and *Reflex*). This manual describes how to use the pipeline with *EsoRex*; this

option offers the most control on the reduction process.

Gasgano is useful as file browser for exploring VIRCAM data sets, however it is not recommended as an application to run the recipes. Using *Gasgano* to run VIRCAM recipes is not discussed further in this manual.

Reflex and the VIRCAM workflow provide a convenient way to execute a fixed setup of the reduction chain; it includes automatic data organization and processing. See [2] and [8] for details on how to use the VIRCAM Reflex workflow and Reflex itself.

5 Data Description

VIRCAM pipeline data can be separated into four general categories: *raw* files, *static calibration* files, *catalogue* files, and *product* files. Raw files are the unprocessed output of the VIRCAM instrument. Static calibration files are a set of mandatory and optional files for various calibration purposes. The catalogue files are optional data used to calibrate photometry and astrometry. Product files are the output of the VIRCAM pipeline processing (as reduced images, master calibration files, object catalogues, etc.). All general categories of files are described in this section.

In preparation for (and during) pipeline processing, all VIRCAM data must be classified into specific categories (e.g. DARK, FLAT, etc.) These files must then be associated with each other and with one (or more) recipes. This classification and association is typically done using FITS header keywords. *Gasgano* and *Reflex* can do the classification automatically by using the specially crafted file describing the classification and association rules (provided with the pipeline); the preferred method is to use *Reflex*. With *EsoRex*, the user must classify and associate files themselves. Note that for large collections of files this can be tedious; the ESO convention on filenames also make this process prone to error. See §6.2.2 for recommendations on classification and association.

5.1 Raw Data

Each VIRCAM raw FITS file contains a primary unit (without data), and 16 extensions (one for each of the 16 chips). Each extension is a 2048x2048 image containing data from a single chip; the EXTNAME keyword identifies the chip name (e.g. `DET1.CHIP3` for chip number 3). The extension number in the raw files usually matches the chip number. For example, extension 1 contains data from chip 1 and extension 2 contains data from chip 2. The same ordering of the raw files is maintained in the pipeline product files.

Raw VIRCAM data retrieved from the ESO archive have the standard ESO archive file names, i.e. an instrument identifier followed by a time stamp, e.g. VCAM.2013-06-20T08:40:45.959. The time stamp corresponds to the contents of the FITS header keywords `MJD-OBS` and `DATE-OBS` respectively, i.e. to the date and time when the exposure was taken (a difference of 1 ms between the file name and the contents of the keywords may be present).

The files returned by the archive are **compressed** using the *fpack* utility; this is indicated by the file name suffix `.fits.fz` instead of the regular `.fits` suffix. The raw data files included in the pipeline kit (demonstration data) are also compressed. However, they have the `.fits` suffix in order for them to be recognised by the `Reflex` environment. These "tile compressed" files may be unpacked using the *funpack* tool distributed as part of the CFITSIO package [1]. There is no need to uncompress the raw data if the recipes are run either from the command line using *EsoRex* or inside the *Reflex* environment.

5.2 Static Calibration Data

The VIRCAM pipeline kit comes with a number of files that are used as part of the calibration of raw data. These files are referred to as "static" calibration files in order to distinguish them from output of recipes that are also used for calibration. Note that the "static" files that come with a particular version of the pipeline may

[1] *https://heasarc.gsfc.nasa.gov/fitsio/*

differ from other versions of the pipeline. Moreover, as set of reference files (prefix `REF_`) are only part of the reflex demo-data and not of the pipeline kit itself.

A short summary of each of these files is given below. A more detailed description can be found in §A.1.

`vircam_phot.fits`: a binary FITS table containing quantities used to convert the VIRCAM photometric system to/from a reference system (e.g. 2MASS). There is one extension per system, and one row in each extension for each VIRCAM filter. The `PRO.CATG` value for this file is `PHOTCAL_TAB`.

The UK in-kind VIRCAM archive reprocessing and new ESO VIRCAM imaging pipeline use the 2MASS catalogue to astrometrically and photometrically calibrate the science fields.

The colour transformations used for the conversion between the VIRCAM photometric system and the system used for photometric calibration (e.g. 2MASS) can be found in the fits tables included in the data distribution (`PRO.CATG` = `PHOTCAL_TAB`).

For the current VIRCAM detectors and filters, the transformations used in the `PHOTCAL_TAB` file to the 2MASS filters are:

$$\begin{aligned}
\texttt{Z_VIRCAM} &= 2.025 * \texttt{Jmag_2MASS} - 1.025 * \texttt{Hmag_2MASS} \\
\texttt{Y_VIRCAM} &= 1.610 * \texttt{Jmag_2MASS} - 0.610 * \texttt{Hmag_2MASS} \\
\texttt{J_VIRCAM} &= 0.923 * \texttt{Jmag_2MASS} + 0.077 * \texttt{Hmag_2MASS} \\
\texttt{H_VIRCAM} &= 0.032 * \texttt{Jmag_2MASS} + 0.968 * \texttt{Hmag_2MASS} \\
\texttt{Ks_VIRCAM} &= 0.010 * \texttt{Jmag_2MASS} + 0.990 * \texttt{Kmag_2MASS} \\
\texttt{NB118_VIRCAM} &= 1.100 * \texttt{Jmag_2MASS} - 0.100 * \texttt{Hmag_2MASS} \\
\texttt{NB980_VIRCAM} &= 1.680 * \texttt{Jmag_2MASS} - 0.680 * \texttt{Hmag_2MASS}
\end{aligned}$$

Please consult the following website for updates on the relations:
http://casu.ast.cam.ac.uk/surveys-projects/vista/technical/photometric-properties

`chan_tab_init.fits`: a binary FITS table containing the x/y pixel location of the 16 channels on each VIRCAM detector. This file is used by `vircam_linearity_analyse` to identify the areas over which the detector linearity should be tabulated. It is also used by some recipes when comparing a master frame with a reference frame; the comparison is evaluated over a user-specified number of areas within each channel. The file was created using the `vircam_linearity_analyse` recipe; see the PHU keywords for the data provenance. The `PRO.CATG` value for this file is `CHANNEL_TABLE_INIT`.

`master_chan_tab.fits`: a binary FITS table containing reference values for the detector linearity for the 16 channels on each VIRCAM detector. These values may be calculated explicitly from any set of data using the `vircam_linearity_analyse` recipe; those values can then be used in subsequent recipes in the data reduction cascade. If a user chooses not to run `vircam_linearity_analyse` on their data, the reference values in this file should be used by other recipes. This file was generated using `vircam_linearity_analyse` with data from Aug 2010 and contain values close to the median values derived from all VIRCAM data to date; the provenance of the file can be seen in the primary header. The `PRO.CATG` value for this file is `MASTER_CHANNEL_TABLE`.

`detector_noise.fits`: a binary FITS table containing reference values for the read noise and gain of each VIRCAM detector. These values may be calculated explicitly from any set of data using the

vircam_detector_noise recipe; those values can then be used in subsequent recipes in the data reduction cascade. If a user chooses not to run vircam_detector_noise, the reference values in this file will be used by other recipes. This file was generated using vircam_detector_noise with data from March 2014 and contain values close to the median values derived from all data taken since VIRCAM began taking data; the provenance of the file can be seen in the primary header. The PRO.CATG value for this file is MASTER_READGAIN_TABLE.

REF_DARK_XXX_YYY.fits: a multi-extension FITS file containing a *reference* dark image for each VIRCAM detector. The values of XXX and YYY refer to the DET.DIT and DET.NDIT of the reference image, respectively. These files are used by the vircam_dark_combine recipe to compare the recipe product dark image ('master' dark) to a reference dark image. Only a selection of seven common DIT/NDIT pairs are provided with the detector in correlated double-sampling mode. The reference files were created using vircam_dark_combine with raw data from May 2015; the provenance of each file can be seen in the primary headers. After the recipe created these products, the update_procatg.py script was used to change the PRO.CATG value to REFERENCE_DARK so that these files can be used by Reflex. The statistical distribution of pixels for the reference files are near the median values for all darks taken with VIRCAM.

REF_FLAT_XXX.fits: a multi-extension FITS file containing a *reference* twilight flat image for each VIRCAM detector. The values of XXX refer to the filter used for the reference image (INS.FILT1.NAME). These files are used by the vircam_twilight_flat_combine recipe to compare the recipe product flat image ('master' flat) to a reference flat image. A selection of flats from seven filters are provided with the detector in correlated double-sampling mode. The reference files were created using vircam_twilight_flat_combine with data from Feb 2013; the provenance of each file can be seen in the primary headers. After the recipe created these products, the update_procatg.py script was used to change the PRO.CATG value to REFERENCE_TWILIGHT_FLAT so that the files can be used by Reflex.

SFD_dust_4096_ngp.fits: a FITS image containing the reddening values E_{B-V} in the northern Galactic hemisphere. These values are derived from the dust map described in [7]. The file is used by some VIRCAM recipes for photometric calibration purposes. The PRO.CATG value for this file is SCHLEGEL_MAP_NORTH.

SFD_dust_4096_sgp.fits: as above, but for the southern Galactic hemisphere and with PRO.CATG set to SCHLEGEL_MAP_SOUTH.

Note that the update_procatg.py script is part of the collection of static calibration files; this enables users to create their own reference files.

5.3 Photometric and Astrometric Catalogues

The VIRCAM pipeline can use a number of optional files to calibrate the astrometry and photometry of processed images. **These files are only necessary if a user wishes to process data without an internet connection.** If a user has an internet connection, the VIRCAM recipes can retrieve the required data automatically

through the Strasbourg astronomical Data Center[2] (CDS). It is recommended that VIRCAM data use the 2MASS point source catalogue [1] for the astrometric and photometric calibration.

A short summary of each of these files is given below; see §A.1 for further details.

`master_2mass_tbl.fits`: a FITS binary table used by recipes as an index of the 2MASS point sources. This file has a `PRO.CATG` keyword with value `'MASTER_2MASS_CATALOGUE'`.

`npscXXX.fits`: a FITS binary table containing information about the 2MASS point sources. There is one file for each degree of right ascension, i.e. npsc000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$. These files must reside in the same directory as the `'MASTER_2MASS_CATALOGUE'` file in order to work in combination with the pipeline.

By default, the catalogue fields needed to calibrate the images are retrieved by the pipeline from the CDS. The full catalogue data files are rather large (about 30 Gigabytes), but if needed, can be obtained on request[3].

5.4 Pipeline Products

A brief description of the naming and content of pipeline products is given below; see §6 and §8 for more details.

5.4.1 Naming convention for files

The standard method for naming of a data product file for an ESO pipeline is for the product to be given a 'predictable' name. For example, two runs of the same recipe should yield products with exactly the same names. For recipes that generate several products of the same type, each file will have the same root name, but will have a number appended so that each file name is unique. This is the default naming convention for all the recipes in this package.

For those who would like a more descriptive file name, many of the recipes offer a command line switch called `prettynames`. This system works by creating the output file name from the input file name with an added suffix which denotes the type of product. For example, the recipe `vircam_science_process` produces calibrated exposure files and stacked exposures plus variance images for both. A SoF would include, amongst other things, a list of raw exposures. If the first in the list was called `VCAM.2011-12-25T02:20:16.502.fits`, then the calibrated exposure and its variance would be called `VCAM.2011-12-25T02:20:16.502_ex.fits` and `VCAM.2011-12-25T02:20:16.502_ex_var.fits`, respectively. Stacks are named after the first raw exposure in the SoF, hence this stack and its variance map would be called `VCAM.2011-12-25T02:20:16.502_st.fits` and `VCAM.2011-12-25T02:20:16.502_st_var.fits`, respectively.

The full lists the default file names and the `prettyname` suffixes for each product type are given in §8.

[2] https://cds.unistra.fr/
[3] *https://support.eso.org*

5.4.2 Images

All images are created as multi-extension FITS files. The primary HDU only contains keywords; there is no data array. The primary HDU keywords are derived from the raw images used to create it. The PHU keywords provide general information about the observation; the extension keywords are relevant to the particular detector. If a detector is flagged as 'dead', a 'dummy' data array will still be created for the appropriate FITS extension, i.e. an image product will always have 16 extensions.

5.4.3 Catalogues

Object catalogues are FITS binary tables that contain information about statistically significant sources of emission detected in a recipe product image. They have the same number of extensions as the images from which the catalogue is derived.

5.4.4 Diagnostic tables

Some recipes create FITS binary tables that provide a statistical description of the difference between a master calibration image and a reference image.

6 Data Reduction

6.1 Getting Started with EsoRex

EsoRex is a command-line tool used to execute the VIRCAM recipes. Assuming *EsoRex* is in your `$PATH`, the general structure of an *EsoRex* command line is

```
1> esorex [esorex_options] [recipe [recipe_options] [sof1 [sof2]...]]
```

where `recipe` is the name of a recipe and `sof1, sof2, ...` are the names of files containing a set of frames (see below). Note that a user may point `EsoRex` to the location of recipe libraries by defining the `$ESOREX_PLUGIN_DIR` environment variable or by using the `recipe-dir` command line option. See [3] for a complete description of *EsoRex*.

A full list of *EsoRex* options can be listed with the command

```
1> esorex --help
```

The names and current values of recipe parameters are shown with the command

```
1> esorex --help <recipe name>
```

or

```
1> esorex --man-page <recipe name>
```

Note that the current parameter value can be set from a configuration file or on the command line. The default parameter value used by the recipe is shown between square brackets at the end of the one-line description of the parameter.

A full list of recipes that can be executed with *EsoRex* is shown with the command

```
1> esorex --recipes
```

The last argument of an *EsoRex* command is the name of one (or more) "set-of-frames" or "SoF" file(s). A SoF file is a text file comprising a list of input files for the recipe along with a one-word description of that file. Each line is composed of a `FILENAME`, followed by whitespace, followed by a `TAG`. The `TAG` tells the recipe what kind of file it is and how it should be used in the recipe (e.g. `DARK, OBJECT`). Blank lines and those beginning with the '`#`' character are ignored. If several SoF files are provided, the recipe will concatenate the files before execution. If the number of files with the same tag exceeds the number accepted by the recipe, only the first files that appear in the concatenated SoF are used by the recipe. It is strongly recommended that absolute paths to files be provided rather than relative paths. However, *EsoRex* will recognise environment variables (e.g. `$RAW_DATA_DIR`).

An example of a valid set-of-frames is:

```
1> cat dark.sof
/data/vircam/raw/VCAM.2014-02-21T09:48:53.153.fits DARK
$RAW_DATA_DIR/VCAM.2014-02-21T09:50:36.645.fits DARK
#$RAW_DATA_DIR/VCAM.2014-02-21T09:55:04.515.fits DARK
```

6.2 Data Organization, Classification, & Association

Running the VIRCAM pipeline recipes using *EsoRex* requires the user to construct the appropriate set-of-frames file(s). There are three steps to this process:

classify each input file with the correct `TAG` to be used by each recipe.

organize a set of input files for each recipe, e.g. a group of science observations taken as part of the same template, or a group of calibration files taken with the same filter as the science observations, etc.

associate the output of some recipes as input to other recipes.

6.2.1 Header Keywords

The following table summarizes selected FITS *primary* header keywords that are useful in classifying data files. The keywords are grouped by their context and intended use.

Primary Header Keyword Name	Description
Keywords for frame classification:	
`INSTRUME`	Name of the instrument, e.g. `VIRCAM`
`DPR.CATG`	Raw data frame product category, e.g. `CALIB`
`DPR.TYPE`	Raw data frame product type, e.g. `DARK`
`DPR.TECH`	Raw data frame observation technique, e.g. `IMAGE`
`TPL.ID`	Name of template used to create raw frame, e.g. `VIRCAM_img_cal_dark`
`PRO.CATG`	Pipeline product category, e.g. `MASTER_DARK`
Keywords describing an observation:	
`OBJECT`	Target name
`RA`	Approximate telescope pointing RA (J2000) [deg]
`DEC`	Approximate telescope pointing DEC (J2000) [deg]
`MJD-OBS`	Modified Julian date near start of exposure
`DATE-OBS`	Human readable format of `MJD-OBS`
`OBS.NAME`	Name of the Observation Block (OB)
`OBS.START`	Start time of the OB in `DATE-OBS` format
`OBS.TARG.NAME`	Same as `OBJECT`
`TPL.START`	Start time of the template (within the OB)
`TPL.EXPNO`	Exposure sequence number within the template
`TPL.NEXP`	Number of exposures within the template

continued on next page

Primary Header Keyword Name	Description
Keywords describing the instrument/detector setup:	
`DET.DIT`	Detector integration time [sec]
`DET.NDIT`	Number of integrations, each with duration `DET.DIT`
`DET.NCORRS.NAME`	Name of detector read-out mode, e.g. "Double"
`INS.FILT1.NAME`	Name of filter in first filter position

Almost all VIRCAM pipeline product files report a number of Quality Control (QC) parameters. These QC parameters are values which are computed by the recipes as indicators of the quality of the raw data and the reduction process. They are available from the FITS header of the pipeline products as hierarchical keywords starting with the leading group component 'QC'.

6.2.2 Classification

As discussed in §6.1, each recipe must be provided with a *set-of-frames* (SoF) file or files. The SoF file(s) consist of a list of filenames and frame tags for each file. This section discusses the valid frame tags for each recipe and recommendations for how to determine the frame tag from header keywords. Recall that VIRCAM data fall into one of four categories: raw data (§5.1), static calibration data (§5.2), photometric/astrometric catalogues (§5.3), and pipeline product data (§5.4). The tables below describe the frame tags for each category in turn.

Table 6.1 shows the frame tags for VIRCAM raw data files and the recipes that accept files with those tags. It is recommended that these tags be determined from a unique combination of the primary header keywords DPR.CATG, DPR.TYPE, and DPR.TECH, as shown in the last three columns.

Frame tag	Recipe(s)	DPR.CATG	DPR.TYPE	DPR.TECH
FLAT_LAMP_LINEARITY	vircam_linearity_analyse	CALIB	FLAT,LAMP,LINEARITY	IMAGE
FLAT_LAMP_CHECK	vircam_linearity_analyse	CALIB	FLAT,LAMP,CHECK	IMAGE
DARK_LINEARITY	vircam_linearity_analyse	CALIB	DARK,LINEARITY	IMAGE
DARK_CHECK	vircam_linearity_analyse	CALIB	DARK,CHECK	IMAGE
DARK	vircam_dark_combine	CALIB	DARK	IMAGE
FLAT_TWILIGHT	vircam_twilight_flat_combine	CALIB	FLAT,TWILIGHT	IMAGE
DARK_GAIN	vircam_detector_noise	CALIB	DARK,GAIN	IMAGE
FLAT_LAMP_GAIN	vircam_detector_noise	CALIB	FLAT,LAMP,GAIN	IMAGE
OBJECT	vircam_science_process	SCIENCE	OBJECT	IMAGE,JITTER
OBJECT_EXTENDED	vircam_science_process	SCIENCE	OBJECT_EXTENDED	IMAGE,JITTER
SKY_OFFSET	vircam_science_process	SCIENCE	SKY_OFFSET	IMAGE

Table 6.1: Frame tags for raw VIRCAM data

Table 6.2 shows the frame tags for VIRCAM static calibration files and the recipes that accept files with those tags. As stated before, by default, the catalogue fields needed to calibrate the images are retrieved by the pipeline from the CDS. The full catalogue data files are rather large (about 30 Gigabytes), but if needed, can be obtained on request[4]. The third column shows the filename provided in the pipeline kit that is recommended to be identified with the tag; see §5.2 for a description of each file. A user is free to create her own static calibration file for any tag, but it must meet the data format specification outlined in §A.1. Note that some of these tags are *optional* inputs to the recipe; see §8 for details.

Frame tag	**Recipe(s)**	**Pipeline kit filename**
`REFERENCE_DARK`	vircam_dark_combine	`REF_DARK_XXX_YYY.fits`
`REFERENCE_TWILIGHT_FLAT`	vircam_twilight_flat_combine	`REF_FLAT_XXX.fits`
`CHANNEL_TABLE_INIT`	vircam_linearity_analyse, vircam_dark_combine, vircam_twilight_flat_combine	`chan_tab_init.fits`
`MASTER_READGAIN_TABLE`	vircam_science_process	`readgain.fits`
`MASTER_CHANNEL_TABLE`	vircam_dark_combine, vircam_twilight_flat_combine, vircam_science_process	`master_chan_tab.fits`
`PHOTCAL_TAB`	vircam_science_process, vircam_science_postprocess	`vircam_phot.fits`
`SCHLEGEL_MAP_NORTH`	vircam_science_process, vircam_science_postprocess	`SFD_dust_4096_ngp.fits`
`SCHLEGEL_MAP_SOUTH`	vircam_science_process, vircam_science_postprocess	`SFD_dust_4096_sgp.fits`

Table 6.2: Frame tags for VIRCAM static calibration data

Table 6.3 shows the frame tag for VIRCAM photometric/astrometric catalogue index file and the recipes that accept files with those tags. The third column shows the filename that is recommended to be identified with the tag; see §5.2 for a description of each file.

Frame tag	**Recipe(s)**	**Pipeline kit filename**
`MASTER_2MASS_CATALOGUE`	vircam_science_process, vircam_science_postprocess	`master_2mass_tbl.fits`

Table 6.3: Frame tag for index file to VIRCAM photometric and astrometric catalogue

Table 6.4 lists the input frame tags accepted by each VIRCAM recipe. Note that the same tag may be accepted by more than one recipe. Optional input frames are enclosed by square brackets. More details about the required and optional input frame tags for each VIRCAM recipe can be found in the recipe reference chapter (§8).

[4] *https://support.eso.org*

Recipe	Frame tag
vircam_linearity_analyse	DARK_LINEARITY, FLAT_LAMP_LINEARITY, CHANNEL_TABLE_INIT, [DARK_CHECK], [FLAT_LAMP_CHECK]
vircam_dark_combine	DARK, [REFERENCE_DARK], [MASTER_BPM], [MASTER_CONF], [MASTER_CHANNEL_TABLE], [CHANNEL_TABLE_INIT]
vircam_twilight_flat_combine	FLAT_TWILIGHT, MASTER_DARK, [REFERENCE_TWILIGHT_FLAT], [MASTER_BPM], [MASTER_CONF], [MASTER_CHANNEL_TABLE], [CHANNEL_TABLE_INIT]
vircam_detector_noise	DARK_GAIN, FLAT_LAMP_GAIN, [MASTER_BPM], [MASTER_CONF]
vircam_science_process	OBJECT, OBJECT_EXTENDED, [SKY_OFFSET], MASTER_DARK, MASTER_TWILIGHT_FLAT, MASTER_CONF, MASTER_READGAIN_TABLE, MASTER_CHANNEL_TABLE, PHOTCAL_TAB, SCHLEGEL_MAP_NORTH, SCHLEGEL_MAP_SOUTH, [MASTER_2MASS_CATALOGUE], [MASTER_OBJMASK], [MASTER_SKY]
vircam_science_postprocess	JITTERED_IMAGE_SCI, CONFIDENCE_MAP_SCI, PHOTCAL_TAB, MASTER_CONF, SCHLEGEL_MAP_NORTH, SCHLEGEL_MAP_SOUTH, [MASTER_2MASS_CATALOGUE]

Table 6.4: Accepted frame tags for each VIRCAM recipe

6.2.3 Organization

It is important to ensure that the input files sent to a recipe represent a coherent group of data. The variation in users' observing strategies, science outcomes, etc. make it difficult to provide concrete rules for how to do this. It is recommended that calibration data and science data are grouped together by detector read-out mode (`DET.NCORRS.NAME`), integration time (`DET.DIT` and `DET.NDIT`), and filter (`INS.FILT1.NAME`). Science observations are generally grouped together by target name (`OBS.TARG.NAME`) and either template (`TPL.START`) or OB (`OBS.START`). The use of calibration data with different properties than the science data may yield less than ideal results.

6.2.4 Association

Care must also be taken in associating the output of some recipes with input to other recipes. There are many instances where an output file from one recipe may be used as an input files for another (or the same!) recipe. In this situation, the frame tag for the input files is (usually) identical to the `PRO.CATG` value of the output file. There is one exception to this: `vircam_science_process` creates a `MEAN_OFFSET_SKY` image and is a model of the sky subtracted from the input science frames. A user may chose to use this product as a pre-defined `MASTER_SKY` as an input to this recipe in order to avoid the creation of a sky model (and hence shorten the execution time). Table 6.5 lists `PRO.CATG` keyword values for all output files that may be used as an input file for a VIRCAM recipe. The lefthand column is the name of the recipe that generates a file with the `PRO.CATG`

value(s) in the right hand column. Note that this table is not a complete list of all outputs from all of the recipes; see §8 for details.

Recipe	**Output** `PRO.CATG`
vircam_linearity_analyse	`MASTER_CHANNEL_TABLE`, `MASTER_BPM`
vircam_dark_combine	`MASTER_DARK`
vircam_twilight_flat_combine	`MASTER_TWILIGHT_FLAT`, `MASTER_CONF`
vircam_detector_noise	`MASTER_READGAIN_TABLE`
vircam_science_process	`JITTERED_IMAGE_SCI`, `CONFIDENCE_MAP_SCI`, `MEAN_OFFSET_SKY`

Table 6.5: Output pipeline products that may be used as input frames

6.3 Data Reduction Cascade

Figures 6.1–6.2 show a visual representation and overview of the data reduction with the VIRCAM pipeline. The cascade for generating master calibration data is shown in Figure 6.1. The cascade for processing science frames is shown in Figure 6.2.

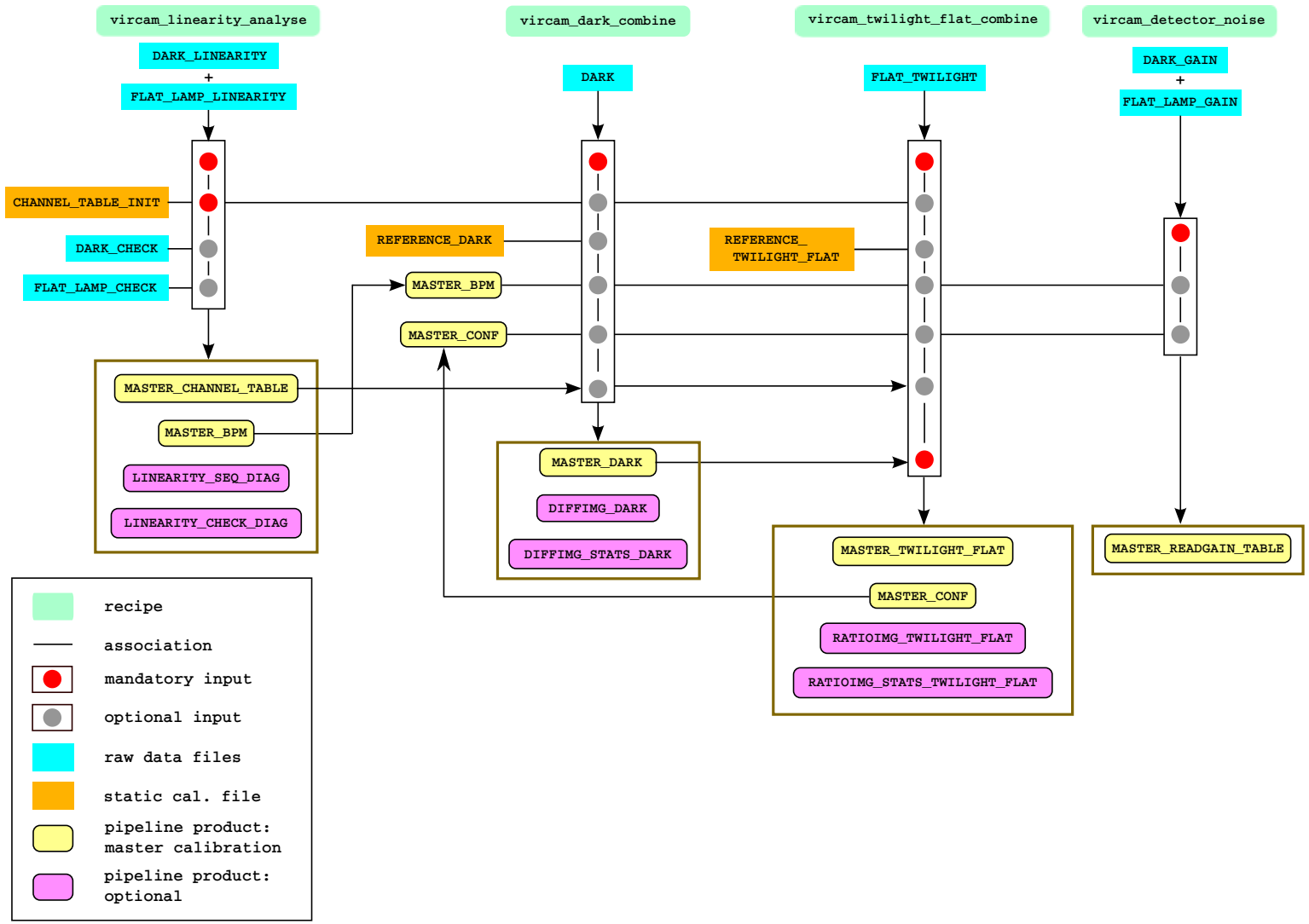

Figure 6.1: Data reduction cascade and tag association for VIRCAM master calibration recipes.

6.4 Calibration

This section describes the steps to generate the 'master' calibration data: a master linearity table, dark, flat field, bad pixel map, confidence map, and detector noise and gain values. These steps should be taken in the order they appear in this manual. Note that the recipes described in this section are run with most recipe parameters set to their default values with the exception of the parameter `ext`. The default value (`1`) instructs the recipes to only process data on one chip. In the sections below, this parameter has been changed to `0` so that all chips are processed.

Only a brief description of the recipes is provided here; see §8 for details. Most of the file names listed below can be found in the demo data as part of the pipeline kit. Note, however, that the absolute pathnames shown in the SoF files depend on where the user has installed her data; they may not be the same as shown below.

6.4.1 Linearity

The VIRCAM detector is known to suffer from non-linearity at the level of $\approx 2 - 10\%$ (at 10,000 ADUs). The `vircam_linearity_analyse` recipe quantifies the magnitude of this non-linearity from a set of special calibration data. The pipeline kit already comes with a `MASTER_CHANNEL_TABLE` that quantifies the non-linearity from data taken in Aug 2010. However, the level of non-linearity is known to change significantly ($\approx$ few percent) over timescales of $\sim$ months. Users are strongly advised to run the `vircam_linearity_analyse` recipe on data that were taken as close as possible to the time of the science observations.

The recipe uses a series of dome flat images (and associated dark images) to calculate the linearity coefficients and to identify bad pixels. The recipe requires a sequence of files with tags `DARK_LINEARITY` and `FLAT_LAMP_LINEARITY`. The coordinate boundaries of the channels on each chip must be provided in a `CHANNEL_TABLE_INIT` file. An optional list of 'monitor' files may also be provided to the recipe; they have tags `DARK_CHECK` and `FLAT_LAMP_CHECK`. A monitor sequence is used to measure any change in the light source while the linearity sequence is taken.

The recipe creates a `MASTER_CHANNEL_TABLE` file with non-linearity values for each channel on each chip. It also creates a bad pixel mask with the `MASTER_BPM` tag. If requested, the recipe will also create two files with diagnostic information: `LINEARITY_SEQ_DIAG` and `LINEARITY_CHECK_DIAG`.

As an example, to run this recipe using the demo data for a sequence of linearity flats, the SoF file `linearity.sof` should contain these files:

```
1> cat linearity.sof
$STATIC_CAL_DATA_DIR/chan_tab_init.fits CHANNEL_TABLE_INIT
$RAW_DATA_DIR/v20140330_00081.fits DARK_CHECK
$RAW_DATA_DIR/v20140330_00055.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00056.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00057.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00058.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00059.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00060.fits DARK_LINEARITY
```

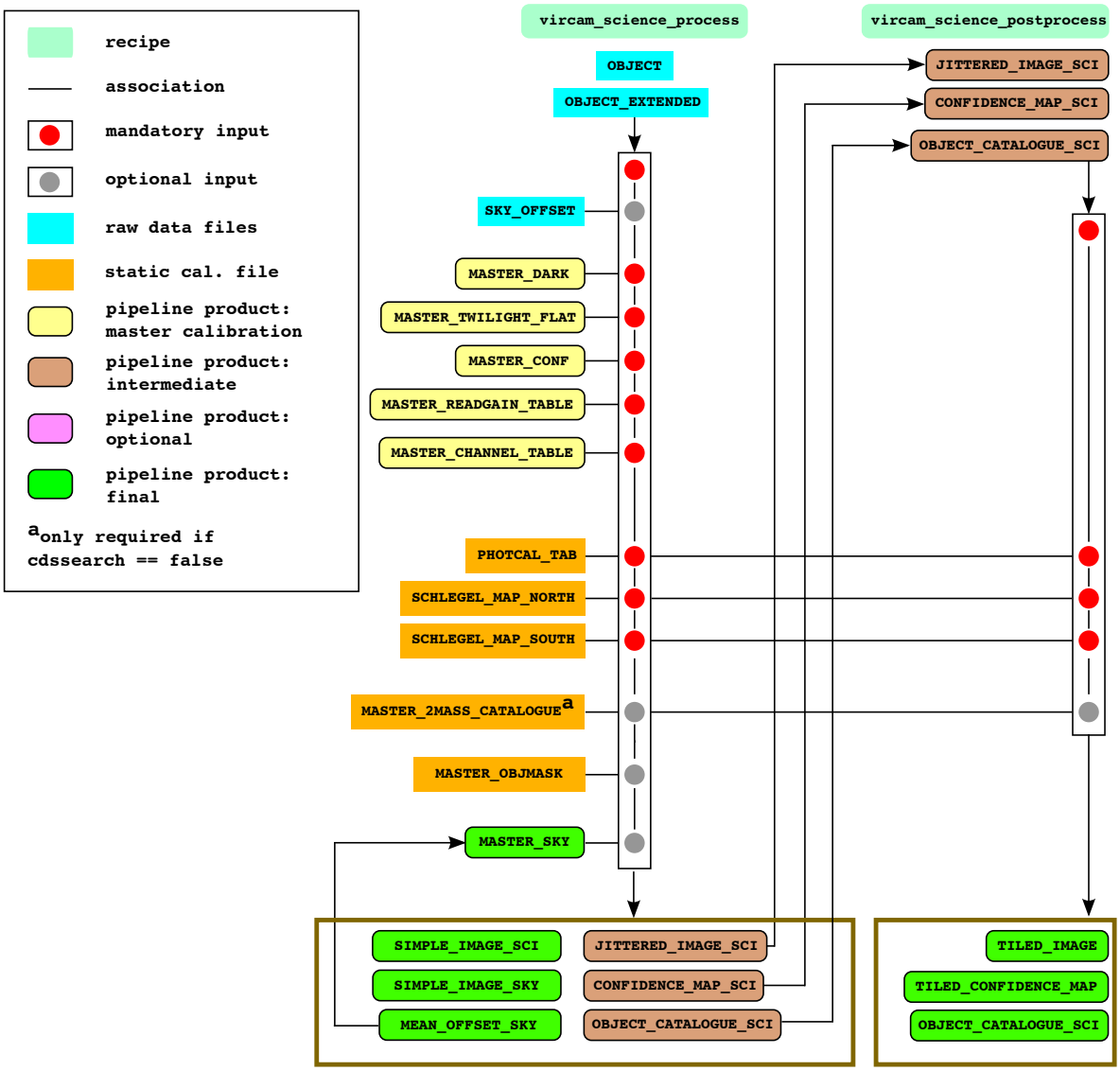

Figure 6.2: Data reduction cascade and tag association for VIRCAM science recipes.

```
$RAW_DATA_DIR/v20140330_00061.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00062.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00063.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00064.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00065.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00066.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00067.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00068.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00069.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00070.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00071.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00072.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00073.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00074.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00075.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00076.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00077.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00078.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00079.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00080.fits DARK_LINEARITY
$RAW_DATA_DIR/v20140330_00083.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00085.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00087.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00089.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00091.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00093.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00095.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00097.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00099.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00101.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00103.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00105.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00107.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00109.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00111.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00113.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00115.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00117.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00119.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00121.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00123.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00125.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00127.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00129.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00131.fits FLAT_LAMP_CHECK
```

```
$RAW_DATA_DIR/v20140330_00133.fits FLAT_LAMP_CHECK
$RAW_DATA_DIR/v20140330_00082.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00084.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00086.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00088.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00090.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00092.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00094.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00096.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00098.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00100.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00102.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00104.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00106.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00108.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00110.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00112.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00114.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00116.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00118.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00120.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00122.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00124.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00126.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00128.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00130.fits FLAT_LAMP_LINEARITY
$RAW_DATA_DIR/v20140330_00132.fits FLAT_LAMP_LINEARITY
```

where the environmental variables (`'$'` prefix) refer to the location of the user's data.

For illustration purposes, the recipe parameter `diagnostic` is changed from its default values (`false`) to `true` in order to create optional diagnostic files. The complete list of recipe parameters are specified in a recipe configuration file:

```
1> cat vircam_linearity_analyse.rc
vircam.vircam_linearity_analyse.adjust=true
vircam.vircam_linearity_analyse.diagnostic=true
vircam.vircam_linearity_analyse.extenum=0
vircam.vircam_linearity_analyse.hthr=8.0
vircam.vircam_linearity_analyse.lthr=8.0
vircam.vircam_linearity_analyse.maxbpmfr=10
vircam.vircam_linearity_analyse.norder=4
```

The command to run the recipe with these recipe input parameters is

```
1> esorex --recipe-config=vircam_linearity_analyse.rc
```

```
vircam_linearity_analyse linearity.sof
```

This will create four files:

`lchantab.fits`: the `MASTER_CHANNEL_TABLE` pipeline product file.

`bpm.fits`: the `MASTER_BPM` pipeline product file.

`ldiag1.fits`: the `LINEARITY_SEQ_DIAG` pipeline product file.

`ldiag2.fits`: the `LINEARITY_CHECK_DIAG` pipeline product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings.

6.4.2 Dark

The next required step is to combine a set of raw dark frames into one 'master' dark image. The `vircam_dark_combine` recipe takes a list of `DARK` frames (with the same detector and exposure time parameters), and combines them to form a mean dark (after rejecting outlying pixel values).

If a `REFERENCE_DARK` file is supplied in the SoF file, then a difference image is formed between the `REFERENCE_DARK` and the `MASTER_DARK`. This difference image (`DIFFIMG_DARK`) can be useful for looking at the evolution of the structure of the dark current and the reset anomaly over time. It can also be useful as a quick check that the recipe has created a reasonably valid master dark. The `REFERENCE_DARK` should have been taken with the same detector settings as the `MASTER_DARK` (e.g., readout mode, DIT, NDIT).

If a master channel table (`MASTER_CHANNEL_TABLE`) or an initial channel table (`CHANNEL_TABLE_INIT`) are provided in the SoF, the median value of the difference image and the root-mean-square deviation within several small areas on the chip are written to a 'difference image statistics table' (`DIFFIMG_STATS_DARK`). These optional input files are only used to identify channel boundaries; no linearity information is used in the creation of the master dark. If neither a `MASTER_CHANNEL_TABLE` nor `CHANNEL_TABLE_INIT` are provided, a statistics table will not be created. If both a `MASTER_CHANNEL_TABLE` and a `CHANNEL_TABLE_INIT` are provided, the `MASTER_CHANNEL_TABLE` will be used.

The inclusion of either a master confidence map (`MASTER_CONF`) or a bad pixel mask (`MASTER_BPM`) in the SoF file can aid in masking out bad pixels when evaluating the statistics that appear in `DIFFIMG_STATS_DARK`. The confidence map and/or bad pixel map are only used when evaluating the statistics; they are not used in the creation of the `MASTER_DARK`. If neither a `MASTER_CONF` nor `MASTER_BPM` are provided, the statistics are calculated assuming that all pixels in the difference image are valid. If both a `MASTER_CONF` and a `MASTER_BPM` are provided, the `MASTER_CONF` will be used.

As an example, in order to make a master dark image from the demo data where DIT = 4 sec, NDIT = 1, the SoF file `dark.sof` should contain these files:

```
1> cat dark.sof
$RAW_DATA_DIR/v20140329_00827.fits DARK
```

```
$RAW_DATA_DIR/v20140329_00828.fits DARK
$RAW_DATA_DIR/v20140329_00829.fits DARK
$RAW_DATA_DIR/v20140329_00830.fits DARK
$RAW_DATA_DIR/v20140329_00831.fits DARK
$STATIC_CALIB_DATA/REF_DARK_4_1.fits REFERENCE_DARK
$MASTER_CALIB_DATA/bpm.fits MASTER_BPM
$MASTER_CALIB_DATA/lchantab.fits MASTER_CHANNEL_TABLE
```

where the environmental variables (`'$'` prefix) refer to the location of the user's data. The command to create the master dark for all chips using most of the default recipe input parameters is

```
1> esorex vircam_dark_combine --ext=0 dark.sof
```

This will create three files:

`darkcomb.fits`: the `MASTER_DARK` pipeline product file.

`darkdiff.fits`: the `DIFFIMG_DARK` pipeline product file.

`darkdifftab.fits`: the `DIFFIMG_STATS_DARK` pipeline product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. If a `REFERENCE_DARK` is omitted from the SoF file, the `darkdiff.fits` and `darkdifftab.fits` files will not be created. A screenshot of each of the master dark is shown in Figure 6.3.

6.4.3 Flat Field

The next required step is to combine a set of raw twilight flat frames into one master flat image. The `vircam_twilight_flat_combine` recipe takes a list of `FLAT_TWILIGHT` frames (with the same detector and filter parameters), corrects each frame for dark current, and combines them to form a normalised, mean twilight flat (after rejecting outlying pixel values). The median value of a master twilight flat frame for each chip is 1.0. The recipe also creates a master confidence maps.

Note that an additional required input to the recipe is a `MASTER_DARK`. This `MASTER_DARK` should have been created from raw data that share the same detector settings (e.g. readout mode, DIT, NDIT) as the raw twilight flat data.

Users are recommended to include a `MASTER_BPM` in the SoF for this recipe. The bad pixel map is used to create a more accurate master confidence map. It is also recommended to include a `MASTER_CHANNEL_TABLE` in the SoF to correct the raw flat for non-linearities in the detector.

If a `REFERENCE_TWILIGHT_FLAT` file is supplied in the SoF file, then a ratio image is formed between the `REFERENCE_TWILIGHT_FLAT` and the `MASTER_TWILIGHT_FLAT`. This ratio image (`RATIOIMG_TWILIGHT_FLAT`) can be useful for looking at the evolution of the structure of the filter or flat field over time. It can also be useful as a quick check that the recipe has created a reasonably valid master flat. The `REFERENCE_TWILIGHT_FLAT` should have been taken with the same filter as the

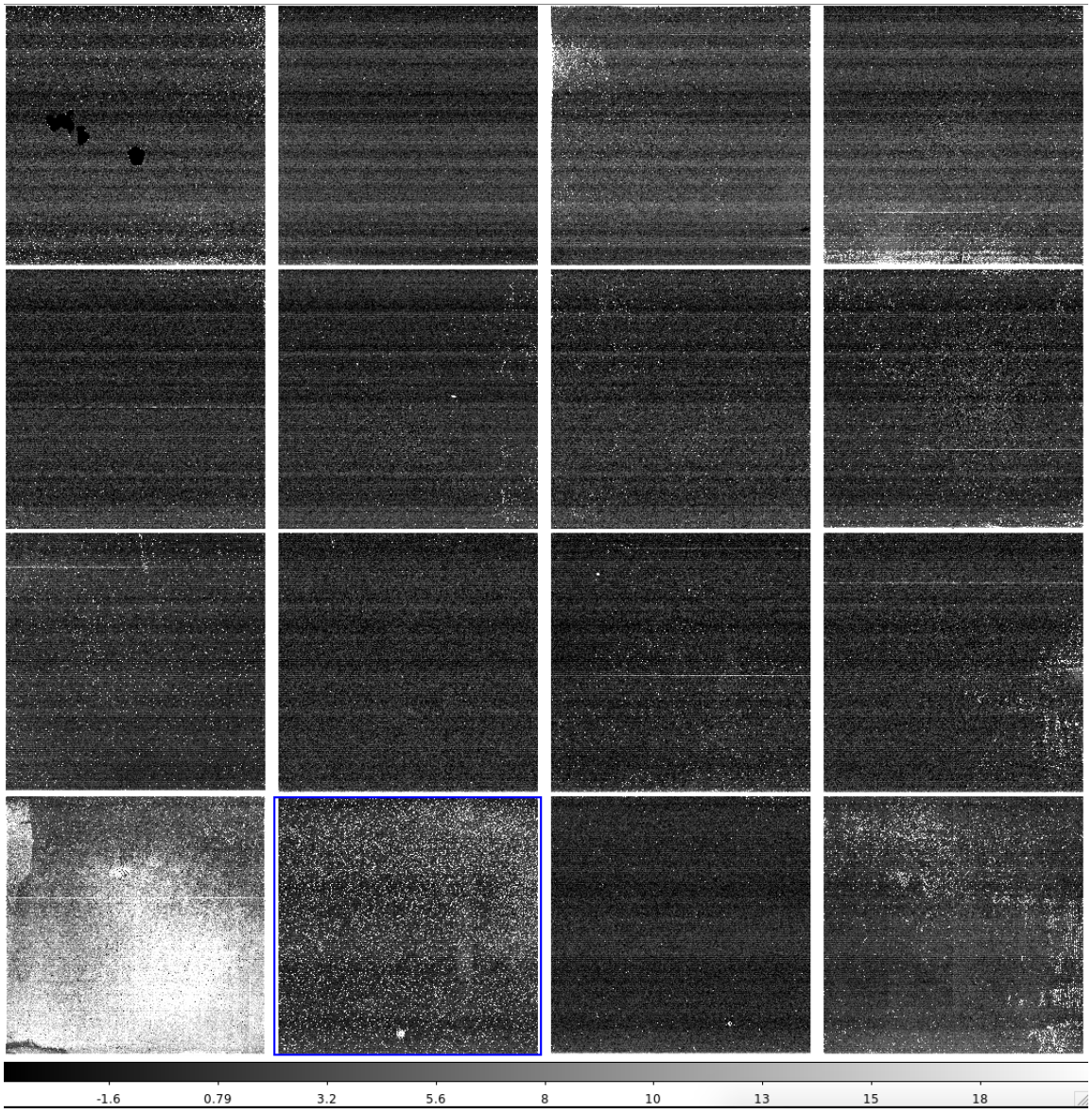

Figure 6.3: Example VIRCAM master dark for DIT = 4 sec, NDIT = 1. Each panel shows one chip, with the chip number increasing from 1 in the upper left to 16 in lower right.

MASTER_TWILIGHT_FLAT.

If a master channel table (MASTER_CHANNEL_TABLE) or an initial channel table (CHANNEL_TABLE_INIT) are provided in the SoF, the median value of the ratio image and the root-mean-square deviation within several small areas on the chip are written to a 'ratio image statistics table' (RATIOIMG_STATS_TWILIGHT_FLAT). These channel table files are used to identify channel boundaries on each chip. If neither a MASTER_CHANNEL_TABLE nor CHANNEL_TABLE_INIT are provided, a statistics table will not be created. If both a MASTER_CHANNEL_TABLE and a CHANNEL_TABLE_INIT are provided, the MASTER_CHANNEL_TABLE will be used.

The inclusion of either a master confidence map (MASTER_CONF) or a bad pixel mask (MASTER_BPM) in the SoF file can aid in masking out bad pixels when evaluating the statistics that appear in RATIOIMG_STATS_TWILIGHT_FLAT. If neither a MASTER_CONF nor MASTER_BPM are provided, the statistics are evaluated assuming that all pixels in the ratio image are valid. If both a MASTER_CONF and a MASTER_BPM are provided, the MASTER_CONF will be used.

For example, in order to make a master flat image from the demo data with the K_S filter, the SoF file should contain these files:

```
1> cat flat.sof
$RAW_DATA_DIR/v20140329_00001.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00002.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00003.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00004.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00005.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00006.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00007.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00008.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00009.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00010.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00011.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00012.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00013.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00014.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00015.fits FLAT_TWILIGHT
$RAW_DATA_DIR/v20140329_00016.fits FLAT_TWILIGHT
$MASTER_CAL_DATA/darkcomb.fits MASTER_DARK
$MASTER_CAL_DATA/bpm.fits MASTER_BPM
$MASTER_CAL_DATA/lchantab.fits MASTER_CHANNEL_TABLE
$STATIC_CAL_DATA/REF_FLAT_Ks.fits REFERENCE_TWILIGHT_FLAT
```

where the environmental variables ('$' prefix) refer to the pathname of the user's data. Note that the MASTER_DARK frame here is **not the same file** as the one described in §6.4.2 above. The reason is that the raw twilight flat frames here were taken with DIT = 3 sec, NDIT = 1 instead of DIT = 4 sec, NDIT = 1. The MASTER_DARK appropriate for the creation of a master flat here should be created from the collection of raw dark frames in the demo data set with DIT = 3 sec, NDIT = 1.

The command to execute the recipe is

```
1> esorex vircam_twilight_flat_combine --ext=0 flat.sof
```

This will create five files:

`twilightcomb.fits`: the `MASTER_TWILIGHT_FLAT` pipeline product file.

`twilightconf.fits`: the `MASTER_CONF` pipeline product file.

`twilightratio.fits`: the `RATIOIMG_TWILIGHT_FLAT` pipeline product file.

`twilightratiotab.fits`: the `RATIOIMG_STATS_TWILIGHT_FLAT` pipeline product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. If a `REFERENCE_TWILIGHT_FLAT` is omitted from the SoF file, the `twilightratio.fits` and `twilightratiotab.fits` files will not be created. Screenshots of some of these pipeline product files are shown in Figures 6.4 - 6.6.

6.4.4 Detector Noise [optional]

In order to process science images, a `MASTER_READGAIN_TABLE` file is needed. This file is a table containing the readnoise and gain for each VIRCAM chip. If a user chooses to use the `MASTER_READGAIN_TABLE` provided with the pipeline kit, they should skip this section. However, if a user would like to create their own `MASTER_READGAIN_TABLE` file from raw data, she should use the `vircam_detector_noise` recipe described here.

The `vircam_detector_noise` recipe takes one pair of raw `DARK_GAIN` frames and one pair of raw dome flats (`FLAT_LAMP_GAIN`) frames as mandatory input. To get sensible results, each of these files must have identical DIT values and NDIT = 1. A `MASTER_BPM` or `MASTER_CONF` may be provided to ensure that bad pixels are not included in the recipe processing. The recipe compares the dark-corrected raw flat frames and calculates the readnoise and gain of each chip. Analysis of data since July 2011 shows that the gain varies randomly at the level of $\approx$ 2-3% percent over time, with a few larger discontinuities corresponding to changes in the detector electronics. Users are advised to select two flat fields of similar illumination that are bright enough to have a very high S/N ratio, but not so bright that they are near saturation. It is also recommended to run this recipe using data close in time to the science frames, because the detector readnoise and gain vary over time.

As an example, to create a `MASTER_READGAIN_TABLE` file from the demo data, the SoF file could contain these files:

```
1> cat readgain.sof
$RAW_DATA_DIR/v20140330_00046.fits DARK_GAIN
$RAW_DATA_DIR/v20140330_00047.fits DARK_GAIN
$RAW_DATA_DIR/v20140330_00048.fits FLAT_LAMP_GAIN
$RAW_DATA_DIR/v20140330_00049.fits FLAT_LAMP_GAIN
$MASTER_CAL_DATA/twilightconf.fits MASTER_CONF
```

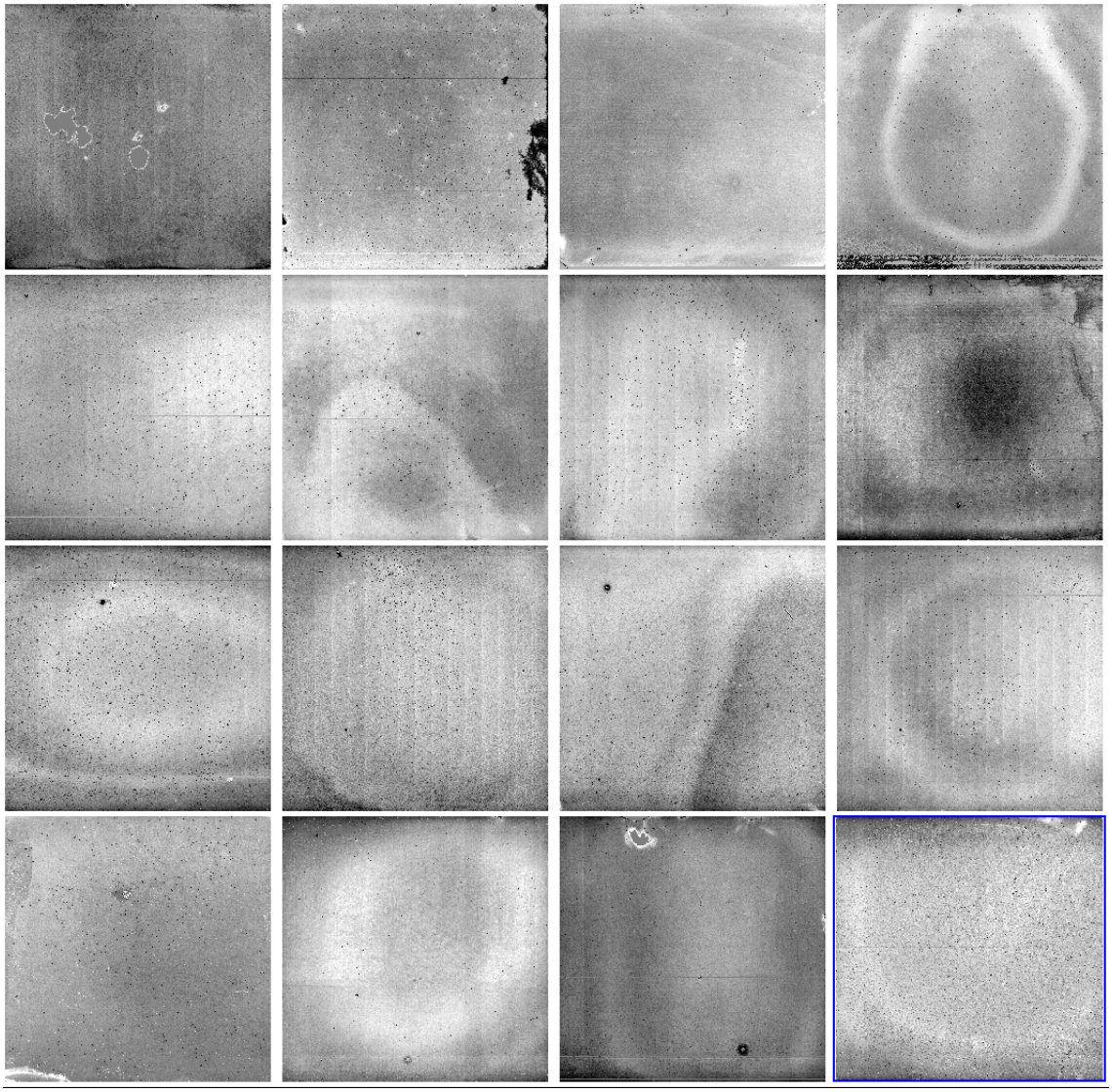

Figure 6.4: Example VIRCAM master twilight flat for filter K_S; all 16 chips are shown.

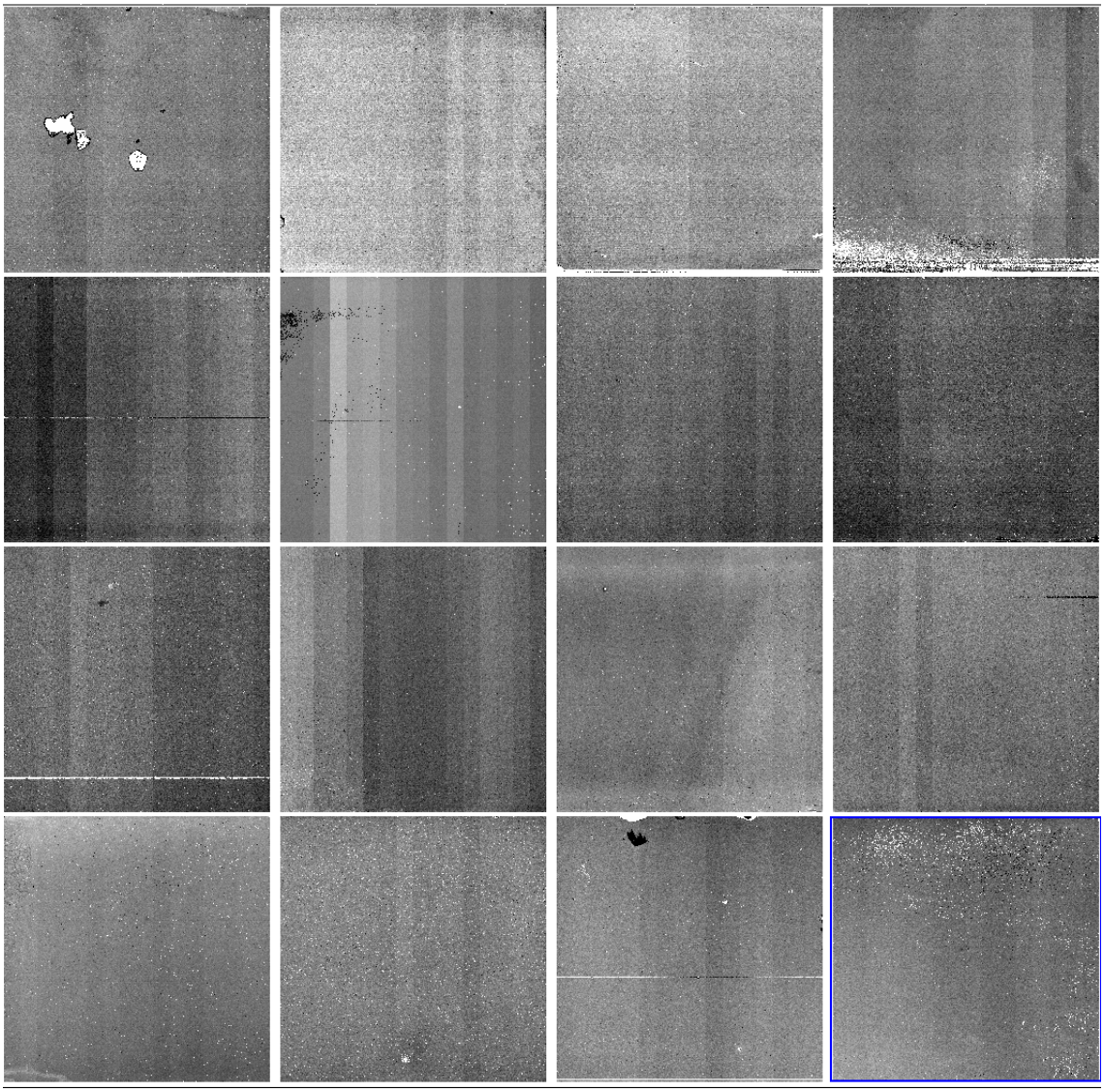

Figure 6.5: Example `RATIOIMG_TWILIGHT_FLAT`, i.e. the ratio of a master twilight flat and a reference twilight flat.

Select
All
Invert

	xmin 1J pixels Modify	xmax 1J pixels Modify	ymin 1J pixels Modify	ymax 1J pixels Modify	chan 1J pixels Modify	mean 1E ADU Modify	median 1E ADU Modify	variance 1E ADU**2 Modify	mad 1E ADU Modify
1	1	128	1	256	1	8.538052E-01	8.538485E-01	3.177749E-05	2.339005E-03
2	1	128	257	512	1	8.536872E-01	8.536715E-01	1.215041E-05	2.280712E-03
3	1	128	513	768	1	8.539745E-01	8.539820E-01	1.200080E-05	2.291024E-03
4	1	128	769	1024	1	8.537162E-01	8.537064E-01	1.217713E-05	2.309710E-03
5	1	128	1025	1280	1	8.535472E-01	8.535683E-01	1.204814E-05	2.289891E-03
6	1	128	1281	1536	1	8.521054E-01	8.521070E-01	1.271157E-05	2.364099E-03
7	1	128	1537	1792	1	8.520393E-01	8.520104E-01	1.248416E-05	2.313495E-03
8	1	128	1793	2048	1	8.542032E-01	8.542527E-01	1.413070E-05	2.525389E-03

Figure 6.6: Extract from example VIRCAM `RATIOIMG_STATS_TWILIGHT_FLAT` table, i.e. a statistical description of `RATIOIMG_TWILIGHT_FLAT`. The first eight rows from the first extension of the file are shown, corresponding to the first channel on `DET1.CHIP1`.

where the environmental variables (`'$'` prefix) refer to the pathname of the user's data. Note that the `MASTER_CONF` frame here may not necessarily be the one described in §6.4.3 above. The `FLAT_LAMP_GAIN` files can be part of the set used to create a `MASTER_CONF` file to be used in this recipe, but this is not necessary. Another option is to use a `MASTER_BPM` which, in principle, should be the same for all filters. The recipe uses either of these two files only to identify and discard bad pixels.

The command to create the master readgain file for all chips is

```
1> esorex vircam_detector_noise --ext=0 readgain.sof
```

This will create one file:

`detector_noise.fits`: the `MASTER_READGAIN_TABLE` pipeline product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings.

6.5 Science Observations

There are two recipes for processing VIRCAM science images: `vircam_science_process` and `vircam_science_postprocess`. A typical sequence of VIRCAM science images consists of two or more observations that are offset from each other by a small angular distance ('jitter'); the magnitude of each jitter is usually a random number bounded by some user-defined limit. The `vircam_science_process` recipe combines a sequence of raw, jittered images into one 'stacked' image (also called a 'pawprint'). This combination is done separately for each chip, i.e. a pawprint comprises sixteen FITS extensions (one for each chip). The recipe also creates one fully processed and calibrated image for each raw input image.

By contrast, the `vircam_science_postprocess` recipe combines a number of calibrated pawprints into one 'tile' image. This includes all chips together into the PHU (i.e. a tile has no FITS extensions). A tile is usually created from six pawprints that completely fill in the gaps between chips. The postprocessing recipe may also correct for varying background emission when creating a catalogue of objects in the tile ('nebulise'). If a user does not want to create a tile, the `vircam_science_postprocess` recipe does not need to be run; a stack/pawprint image is fully calibrated. However, if a user does want to create a tile or a catalogue of 'nebulised' objects, both recipes must be run in series.

Users are **strongly cautioned** that, by default, all objects in the reference catalogue (e.g. 2MASS) are used for photometric calibration. If the reference catalogue is from a magnitude-limited survey that is shallower than the VIRCAM data, a bias will adversely affect this calibration at the ≈ 0.05 mag level. This bias is due to the large number of faint objects with large errors and/or inaccurate colours. This effect can be mitigated by excluding stars in the reference catalogue that have errors greater than a certain value. This threshold in error is specified by the `magerrcut` recipe parameter. A better approach may be to create a custom photometric reference catalogue by applying preferred colour and/or error cuts to a master catalogue, e.g. 2MASS.

6.5.1 Creating a pawprint with vircam_science_process

Each of the raw, jittered (`OBJECT` or `OBJECT_EXTENDED`) images are first corrected for instrumental signatures, e.g. linearity, dark current and flat-fielding. The sky emission is removed using offset sky observations

and/or a user-specified algorithm (see §9). The astrometry and photometry of individual images are calibrated against 2MASS [1]. For each raw input image, a calibrated image is created with the tag `SIMPLE_IMAGE_SCI`. For each of these calibrated images, there is an option for a catalogue of detected objects to be created with the tag `OBJECT_CATALOGUE_SCI`. One or more mean sky images are created with the tag `MEAN_OFFSET_SKY`. The number of mean sky images is determined by the total elapsed time over which the sequence of input science frames were acquired; see §9.3 for details. If a user supplies the recipe with a set of offset sky observations (`SKY_OFFSET`), the recipe will create calibrated versions of each of those input sky frames with the tag `SIMPLE_IMAGE_SKY`. In addition, the sky emission in the science images will be removed using **only** the offset skies. The stacked image (pawprint) and associated confidence maps are created with tags `JITTERED_IMAGE_SCI`, `CONFIDENCE_MAP_SCI`, respectively. A catalogue of objects detected in each pawprint is created with the tag
`OBJECT_CATALOGUE_SCI`.

The recipe takes a set of `OBJECT` or `OBJECT_EXTENDED` files observations as mandatory input. A set of optional offset `SKY_OFFSET` files may be also be provided. An additional eight calibration files are required as illustrated in Figure 6.2. Note that the `PHOTCAL_TAB` file must have data to calculate the conversion between the VIRCAM photometric system and 2MASS. Astrometric and photometric calibration can be done either by using remote catalogues through CDS or by using a catalogue on a local machine. To use CDS, the recipe parameter `cdssearch` should be set to `true`. If a static calibration index file (`MASTER_2MASS_CATALOGUE`) is specified in the SoF **and** the `cdssearch` parameter is set to `true`, the recipe will use CDS and will ignore the index file. The only recommended catalogue for both astrometry and photometry is 2MASS.

If a user wishes to estimate the sky using the `pawsky_mask_pre` algorithm, a `MASTER_OBJMASK` file may be specified in the SoF (see §9.3).

If a user would like to skip the sky creation performed by the recipe, a processed sky image may be specified with the tag `MASTER_SKY`. To ensure an accurate sky subtraction, it is recommended that these master sky files come from the recipe itself (but is not necessary). Note that in this instance, a sky product file (e.g., `MEAN_OFFSET_SKY`) have a different tag than the input file (e.g. `MASTER_SKY`). Skipping the sky creation in the recipe may result in a significantly faster recipe execution time.

Before running this recipe, users are strongly advised to first run it with `preview_only = true` to see how the recipe will process the data, create skies, etc.

An example of how to process the demo data for the target 'b203' is described below. It is assumed that the appropriate `MASTER_DARK, MASTER_TWILIGHT_FLAT, MASTER_CONF`, `MASTER_READGAIN_TABLE`, and `MASTER_CHANNEL_TABLE` have already been created. The SoF file might look like:

```
1> cat pawprint.sof
$RAW_DATA_DIR/v20140329_00760.fits OBJECT
$RAW_DATA_DIR/v20140329_00761.fits OBJECT
$RAW_DATA_DIR/v20140329_00762.fits OBJECT
$RAW_DATA_DIR/v20140329_00763.fits OBJECT
$RAW_DATA_DIR/v20140329_00764.fits OBJECT
$RAW_DATA_DIR/v20140329_00765.fits OBJECT
$RAW_DATA_DIR/v20140329_00766.fits OBJECT
$RAW_DATA_DIR/v20140329_00767.fits OBJECT
$RAW_DATA_DIR/v20140329_00768.fits OBJECT
```

```
$RAW_DATA_DIR/v20140329_00769.fits OBJECT
$RAW_DATA_DIR/v20140329_00770.fits OBJECT
$RAW_DATA_DIR/v20140329_00771.fits OBJECT
$MASTER_CAL_DATA/darkcomb.fits MASTER_DARK
$MASTER_CAL_DATA/twilightcomb.fits MASTER_TWILIGHT_FLAT
$MASTER_CAL_DATA/twilightconf.fits MASTER_CONF
$MASTER_CAL_DATA/detector_noise.fits MASTER_READGAIN_TABLE
$MASTER_CAL_DATA/lchantab.fits MASTER_CHANNEL_TABLE
$TWOMASS_CAT_DATA/master_2mass_tbl.fits MASTER_2MASS_CATALOGUE
$STATIC_CAL_DATA/vircam_phot.fits PHOTCAL_TAB
$SCHLEGEL_DATA/SFD_dust_4096_ngp.fits SCHLEGEL_MAP_NORTH
$SCHLEGEL_DATA/SFD_dust_4096_sgp.fits SCHLEGEL_MAP_SOUTH
```

where the environmental variables (`'$'` prefix) refer to the pathname of the user's data.

The command to execute the recipe with the default recipe parameters is

```
1> esorex vircam_science_process pawprint.sof
```

This will create 31 files:

`exp_[1-12].fits`: the twelve `SIMPLE_IMAGE_SCI` product files; one for each input `OBJECT` file. The number in the filename is the chronological order of the files in the SoF.

`sky_1.fits`: the `MEAN_OFFSET_SKY` product file.

`stack_[1-6].fits`: the `JITTERED_IMAGE_SCI` product files, one for each pair of jittered `OBJECT` files.

`stack_cat_[1-6].fits`: the six `OBJECT_CATALOGUE_SCI` files associated with each `JITTERED_IMAGE_SCI` file.

`stack_conf_[1-6].fits`: the six `CONFIDENCE_MAP_SCI` files associated with each `JITTERED_IMAGE_SCI` file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true'. A screenshot of one of the pawprints is shown in Figure 6.7.

6.5.2 Creating a tile with vircam_science_postprocess

A tile is created by combining a set of calibrated pawprint images and their corresponding confidence maps . A catalogue of objects detected on each pawprint must also be provided in order to refine the astrometric and photometric calibration. After the images are combined, the resulting tile is calibrated astrometrically and photometrically with 2MASS. The tile and corresponding confidence map are created with the tags

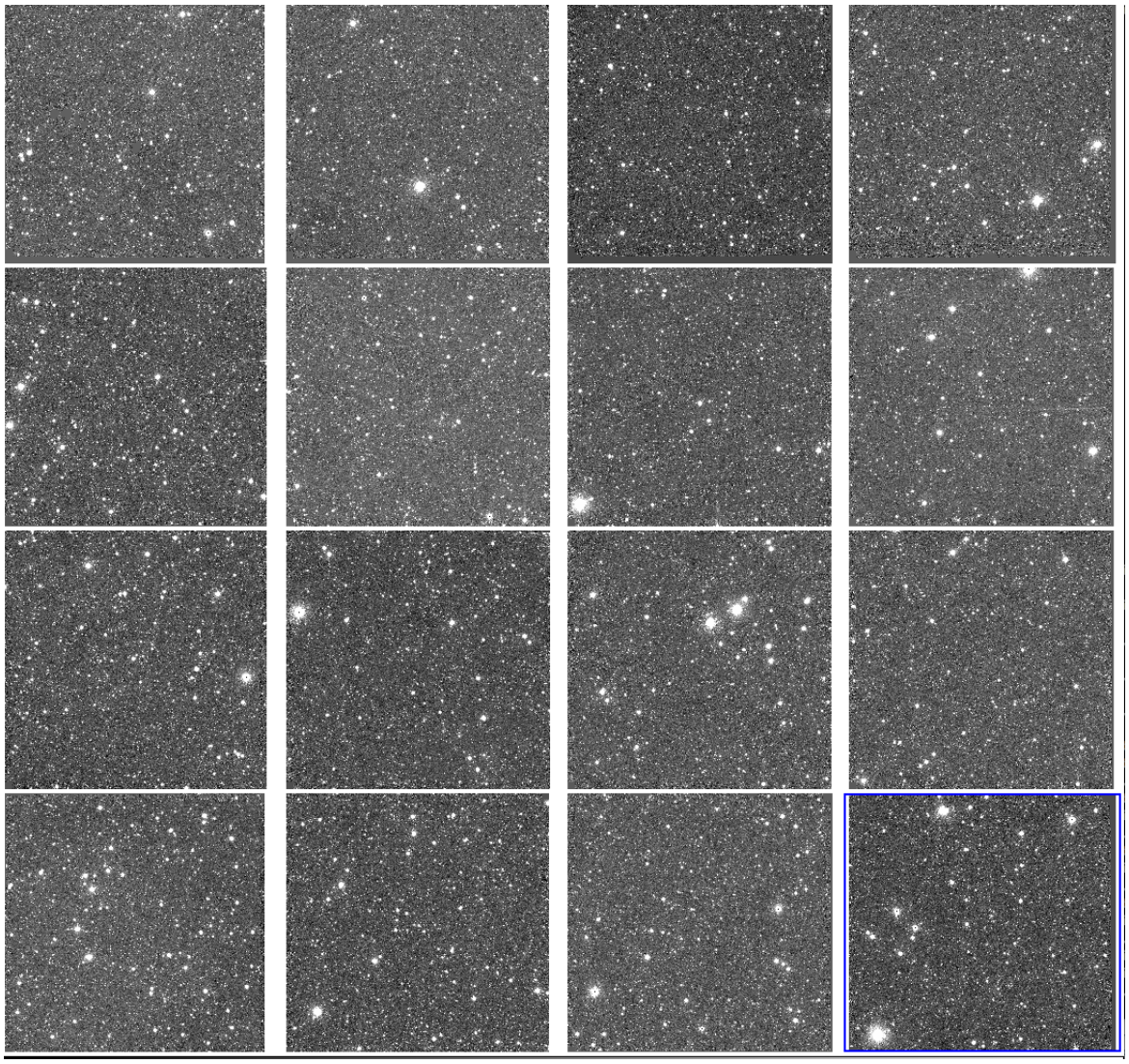

Figure 6.7: An example stack/pawprint created by `vircam_science_process` from twelve `OBJECT` images in the demo dataset. All sixteen chips are shown for `stack_1.fits`.

TILED_IMAGE and TILED_CONFIDENCE_MAP, respectively. A catalogue of objects detected in the tile has the tag OBJECT_CATALOGUE_SCI.

The tile may show emission in the background that varies on different spatial scales. This may be due to incomplete sky subtraction and/or diffuse astrophysical nebular emission. It may affect the accuracy of photometric measurements of objects in the image, especially in regions where the background varies strongly on small scales. There is a recipe parameter that offers the option of removing the effect of this variable background before creating the object catalogue through a process called 'nebulising' (see §9). If this option is used, only the object catalogue is affected; the output tiled image retains the variable background emission.

The recipe takes a set of JITTERED_IMAGE_SCI and CONFIDENCE_MAP_SCI files as mandatory input. The objects detected on each pawprint (OBJECT_CATALOGUE_SCI) is optional. An additional three calibration files are required as illustrated in Figure 6.2. Note that the PHOTCAL_TAB file must have data to calculate the conversion between the VIRCAM photometric system and 2MASS. Astrometric and photometric calibration can be done either by using remote catalogues through CDS or by using a catalogue on a local machine. To use CDS, the recipe parameter cdssearch should be set to true. If a static calibration index file (MASTER_2MASS_CATALOGUE) is specified in the SoF **and** the cdssearch parameter is set to true, the recipe will use CDS and will ignore the index file. The only recommended catalogue for both astrometry and photometry is 2MASS.

An example of how to create a tile from the demo data toward the target 'b203' is described below; this uses the output of the recipe described in §6.5.1 above. The SoF file might look like:

```
1> cat tile.sof
$SCI_PRO_DIR/stack_1.fits JITTERED_IMAGE_SCI
$SCI_PRO_DIR/stack_2.fits JITTERED_IMAGE_SCI
...
$SCI_PRO_DIR/stack_6.fits JITTERED_IMAGE_SCI
$SCI_PRO_DIR/stack_conf_1.fits CONFIDENCE_MAP_SCI
$SCI_PRO_DIR/stack_conf_2.fits CONFIDENCE_MAP_SCI
...
$SCI_PRO_DIR/stack_conf_6.fits JITTERED_IMAGE_SCI
$SCI_PRO_DIR/stack_cat_1.fits OBJECT_CATALOGUE_SCI
$SCI_PRO_DIR/stack_cat_2.fits OBJECT_CATALOGUE_SCI
...
$SCI_PRO_DIR/stack_cat_6.fits OBJECT_CATALOGUE_SCI
$TWOMASS_CAT_DATA/master_2mass_tbl.fits MASTER_2MASS_CATALOGUE
$STATIC_CAL_DATA/vircam_phot.fits PHOTCAL_TAB
$STATIC_CAL_DATA/SFD_dust_4096_ngp.fits SCHLEGEL_MAP_NORTH
$STATIC_CAL_DATA/SFD_dust_4096_sgp.fits SCHLEGEL_MAP_SOUTH
```

where the environmental variables ('$' prefix) refer to the pathname of the user's data. For brevity, ellipses are used in place of the full range of files.

The command to create the pipeline products using the default recipe parameters is

```
1> esorex  vircam_science_postprocess tile.sof
```

This will create 3 files:

`tile_1.fits`: the `TILED_IMAGE` product file.

`tile_conf_1.fits`: the `TILED_CONFIDENCE_MAP` product file.

`tile_cat_1.fits`: the `OBJECT_CATALOGUE_SCI` product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true'. A screenshot of the tile is shown in Figure 6.8.

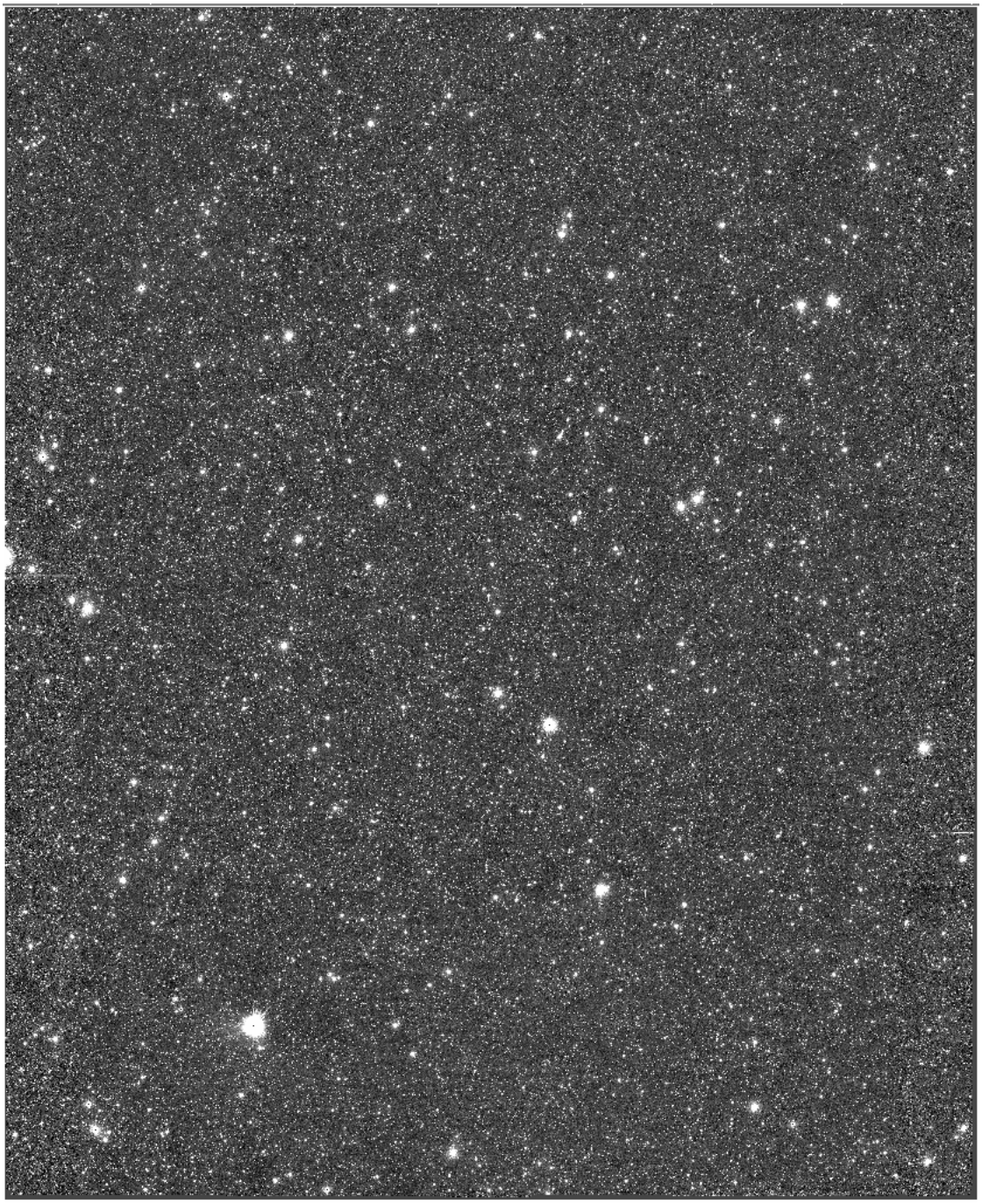

Figure 6.8: A tile created by `vircam_science_postprocess` toward the target 'b203' in the demo dataset.

7 Troubleshooting

This section describes the cause of, and solution to, some common errors messages.

7.1 Log file

All recipes run through `esorex` will generate a logfile with varying levels of information about what the recipe is doing, warnings, etc. The messages that appear in the logfile are prefixed by a timestamp and one of the following:

`[ INFO ]` Informational message about what the recipe is doing, e.g. creating output files.

`[ DEBUG ]` More detailed message about what the recipe is doing, e.g. a statistical description of an intermediate data product. These kind of messages will only appear if a user changes the `esorex` default settings for `msg-level` and/or `log-level` to `debug`. When submitting bug reports, it is recommended that the `debug` option be turned on.

`[WARNING]` A non-fatal error message that indicates something may have gone wrong during the processing, e.g. not enough photometric standards were found during calibration. A user is advised to check the data products carefully if a `WARNING` is issued.

`[ ERROR ]` A fatal error that indicates a serious problem occurred during the processing and the recipe has stopped, e.g. the recipe configuration file cannot be parsed.

The logfile should be submitted with any bug report.

8 Recipe Reference

This section provides a detailed description of each VIRCAM recipe, including input files, parameters, output files, and QC parameters. A description of some of the common algorithms used by one or more recipes can be found in §9. The file format specifications are provided in §A.

Note that some recipe parameters have a shortened name, or 'alias'. In tables of recipe parameters below, the name of the alias is shown in parentheses. When specifying a parameter on the command line, the alias must be used. When specifying a parameter in a configuration file, the default/'long' name must be used.

The parameter type "float" below refers to a 32-bit floating point number.

8.1 vircam_linearity_analyse

8.1.1 Description

Form master dark images from the input raw frames and use these to dark correct a series of dome flat exposures. Using the dark corrected dome flat series, work out linearity coefficients for each data channel.

8.1.2 Input frames

Frame tag	Constraint	# files
`DARK_LINEARITY`	Required	$1 < n$
`FLAT_LAMP_LINEARITY`	Required	$1 < n$
`CHANNEL_TABLE_INIT`	Required	1
`DARK_CHECK`	Optional	$1 < n$
`FLAT_LAMP_CHECK`	Optional	$1 < n$

Users are advised to provide the output from the standard 'VIRCAM_img_cal_linearity' template to this recipe in order to yield reasonable results.

8.1.3 Recipe parameters

Parameter (alias)	Type	Values **default**,other	Description
`norder` (`nord`)	int	**4**, $1 < n < 6$	Order of polynomial fit to linearity curve. The zeroth term is defined to be zero, so the number of coefficients this routine will derive is the same as the polynomial order.

continued on next page

Parameter (alias)	Type	Values **default**,other	Description
`lthr`	float	**8.0**, $0.0 < x$	During combination, reject pixels more than `lthr` units of σ below the background.
`hthr`	float	**8.0**, $0.0 < x$	During combination, reject pixels more than `lthr` units σ above the background.
`maxbpmfr`	int	**10**, $1 \leq n$?	Max number of frames used in bpm analysis (according to order of appearance in SoF)
`adjust`	bool	**true**, false	If true, adjust the statistics measured in the flats by the 'check' flat level. The 'check' flats are interleaved with the linearity flats and ensure that the light source is stable. Any fluctuation in the check flat level can be reflected in the linearity sequence statistics.
`diagnostic`	bool	**false**, true	If true, create diagnostic files
`extenum` (`ext`)	int	**1**, $0 \leq n \leq 16$	Chip number to process. If `0`, process all chips

8.1.4 Product frames

PRO.CATG	Filename	Description
`MASTER_CHANNEL_TABLE`	`lchantab.fits`	Linearity table for each channel and each chip
`MASTER_BPM`	`bpm.fits`	Master bad pixel map
`LINEARITY_SEQ_DIAG`	`ldiag1.fits`	Table of diagnostic info for linearity sequence
`LINEARITY_CHECK_DIAG`	`ldiag2.fits`	Table of diagnostic info for monitor sequence

8.1.5 Quality control parameters

The following QC parameters are found in each extension of the `MASTER_CHANNEL_TABLE`:

`QC.LINEARITY`: percentage non-linearity at 10000 ADU [percent]

`QC.LINERROR`: error of non-linearity at 10000 ADU [percent]

`QC.SCREEN_TOTAL`: total range in screen variation [percent]

`QC.SCREEN_STEP`: maximum step in screen variation [percent]

The following QC parameters are found in each extension of the `MASTER_BPM`:

`QC.BAD_PIXEL_STAT`: Fraction of pixels that are bad

`QC.BAD_PIXEL_NUM`: Number of pixels that are bad

8.2 vircam_dark_combine

8.2.1 Description

Combine a list of dark frames into a mean ('master') dark frame. Optionally compare the output frame to a reference dark frame.

8.2.2 Input frames

Frame tag	Constraint	# files
`DARK`	Required	$1 < n$
`REFERENCE_DARK`	Optional	1
`MASTER_BPM`	Optional	1
`MASTER_CONF`	Optional	1
`MASTER_CHANNEL_TABLE`	Optional	1
`CHANNEL_TABLE_INIT`	Optional	1

If both a `MASTER_CHANNEL_TABLE` and a `CHANNEL_TABLE_INIT` are provided, the `MASTER_CHANNEL_TABLE` will be used. If both `MASTER_BPM` and `MASTER_CONF` are given, the `MASTER_CONF` will be used. Also note that in practice, the number of raw `DARK` files should be at least 5 in order to get useful results.

8.2.3 Recipe parameters

Parameter (alias)	Type	Values **default**,other	Description
`combtype`	int	**1**, 2	When combining images, use the mean (`2`) or median (`1`) pixel value.

continued on next page

Parameter (alias)	Type	Values **default**,other	Description
`scaletype`	int	**1**, $0 \leq n \leq 3$	If non-zero, when combining images, scale each image before combination to bring background values into agreement. Use additive offset (1), multiplicative offset (2), or exptime (3).
`xrej`	bool	**true**, false	If true, an extra rejection cycle is performed after combination, i.e. an extra pass ensures that what has been clipped deserves to have been clipped.
`thresh`	float	**5.0**, $0 < x$	During combination, reject pixels more than `thresh` sigma above the background
`ncells`	int	**8**, 1,2,4,16,32,64	If a reference dark is provided, divide each 128pix-wide channel on each chip into `ncells` when calculating the statistical difference between the reference dark and master dark.
`extenum` (`ext`)	int	**1**, $0 \leq n \leq 16$	Chip number to process. If `0`, process all chips

8.2.4 Product frames

PRO.CATG	Filename	Description
`MASTER_DARK`	`darkcomb.fits`	Master dark image
`DIFFIMG_DARK`	`darkdiff.fits`	If a reference dark is provided in SoF, this is the difference between the master dark and reference dark, i.e. (MASTER - REFERENCE).
`DIFFIMG_STATS_DARK`	`darkdifftab.fits`	If a reference dark is provided in SoF, this is a table with statistical description of the difference image within small areas of each chip.

continued on next page

PRO.CATG	Filename	Description

8.2.5 Quality control parameters

The following QC parameters are found in each extension of the `MASTER_DARK`:

`QC.DARKMED`: Median of mean dark frame

`QC.DARKRMS`: RMS of mean dark frame

`QC.PARTICLE_RATE`: [N/(detector*sec)] Particle rate

`QC.NHOTPIX`: Number of hot pixels

`QC.HOTFRAC`: Hot pixel fraction

`QC.STRIPERMS`: RMS of stripe pattern

`QC.RON12`: Estimate of readnoise + stripe RMS [ADU]

The following QC parameters are found in the `DIFFIMG_DARK`:

`QC.DARKDIFF_MED`: Median of difference image

`QC.DARKDIFF_RMS`: RMS of difference image

8.3 vircam_twilight_flat_combine

8.3.1 Description

Combine a list of twilight flat frames into a mean ('master') frame. The steps are: 1) apply linearity correction to input raw frames, 2) apply dark correction, 3) combine the images into a master twilight flat frame (and calculate a gain correction) and 4) create a master confidence map. If a reference twilight flat is provided, the recipe will create an image of the ratio between the two along with a table describing the statistical properties of the ratio image.

8.3.2 Input frames

Frame tag	**Constraint**	**# files**
`FLAT_TWILIGHT`	Required	$1 < n$
`MASTER_DARK`	Required	1

continued on next page

Frame tag	Constraint	# files
REFERENCE_TWILIGHT_FLAT	Optional	1
MASTER_BPM	Optional	1
MASTER_CONF	Optional	1
MASTER_CHANNEL_TABLE	Optional	1
CHANNEL_TABLE_INIT	Optional	1

For best results, it is strongly recommended that at least 10 raw FLAT_TWILIGHT files be included in a SoF, each with counts between 10,000-20,000 ADUs. If both a MASTER_CHANNEL_TABLE and a CHANNEL_TABLE_INIT are provided, the MASTER_CHANNEL_TABLE will be used. If both MASTER_BPM and MASTER_CONF are provided, only the MASTER_CONF will be used.

8.3.3 Recipe parameters

Parameter (alias)	Type	Values **default**,other	Description
`lthr`	float	**4000**, $0.0 < x$	Discard pixels with values below `lthr` ADUs in the raw flat images.
`hthr`	float	**12000**, $0.0 < x$	Discard pixels with values higher than `hthr` ADUs in the raw flat images.
`combtype`	int	**1**, 2	When combining images, use the mean (`2`) or median (`1`) pixel value.
`scaletype`	int	**2**, $0 \leq n \leq 3$	If non-zero, when combining images, scale each image before combination to bring background values into agreement. Use additive offset (1), multiplicative offset (2), or exptime (3).
`xrej`	bool	**true**, false	If true, an extra rejection cycle is performed after combination, i.e. an extra pass ensures that what has been clipped deserves to have been clipped.
`thresh`	float	**5.0**, $0.0 < x$	During combination, reject pixels more than `thresh` sigma above the background
`ncells`	int	**8**, 1,2,4,16,32,64	If a reference flat is provided, divide each 128pix-wide channel on each chip into `ncells` when calculating the statistical difference between the reference flat and master flat.
`extenum` (`ext`)	int	**1**, $0 \leq n \leq 16$	Chip number to process. If `0`, process all chips

8.3.4 Product frames

PRO.CATG	Filename	Description
MASTER_TWILIGHT_FLAT	twilightcomb.fits	Combined twilight flat image
MASTER_CONF	twilightconf.fits	Confidence map derived from raw flat images
RATIOIMG_TWILIGHT_FLAT	twilightratio.fits	If a reference flat is provided in SoF, this is the ratio of the master flat to the reference flat, i.e. (MASTER/REFERENCE).
RATIOIMG_STATS_TWILIGHT_FLAT	twilightratiotab.fits	If a reference flat is provided in SoF, this is a table with statistical description of the ratio image within small areas of each chip.

8.3.5 Quality control parameters

The following QC parameters are found in the `MASTER_TWILIGHT_FLAT`:

`QC.FLATRMS`: RMS of output flat

`QC.FLATMIN`: Ensemble minimum

`QC.FLATMAX`: Ensemble maximum

`QC.FLATAVG`: Ensemble average

`QC.FLATRNG`: Ensemble range

`QC.TWIPHOT`: [adu] Estimated photon noise

`QC.TWISNRATIO`: Estimated S/N ratio

The following QC parameters are found in the `RATIOIMG_TWILIGHT_FLAT`:

`QC.FLATRATIO_MED`: Median of ratio map

`QC.FLATRATIO_RMS`: RMS of ratio map

8.4 vircam_detector_noise

8.4.1 Description

Measures the read noise and gain of a chip using two dome flat exposures and two dark exposures. The flats should have the same illumination. All four frames should have the same exposure time. If the SOF contains more than two files of a given type, the others are ignored.

8.4.2 Input frames

Frame tag	Constraint	# files
`DARK_GAIN`	Required	2
`FLAT_LAMP_GAIN`	Required	2
`MASTER_BPM`	Optional	1
`MASTER_CONF`	Optional	1

Note that if both `MASTER_BPM` and `MASTER_CONF` are given, the `MASTER_CONF` will be used.

8.4.3 Recipe parameters

Parameter (alias)	Type	Values **default**,other	Description
`thresh`	float	**5.0**, $0.0 < x$	During combination, reject pixels more than `thresh` sigma above the background
`extenum` (`ext`)	int	**1**, $0 \leq n \leq 16$	Chip number to process. If `0`, process all chips

8.4.4 Product frames

PRO.CATG	Filename	Description
`MASTER_READGAIN_TABLE`	`detector_noise.fits`	Table containing readnoise and gain for each chip

8.4.5 Quality control parameters

The following QC parameters are found in each extension of the `MASTER_READGAIN_TABLE`:

`QC.READNOISE`: Calculated detector readnoise [e-]

`QC.CONAD`: Calculated detector gain [e-/ADU]

`QC.COUNTS`: Dark corrected dome counts [ADU]

`QC.LAMPFLUX`: Dark corrected flux level [ADU/sec]

8.5 vircam_science_process

8.5.1 Description

Process a complete pawprint for VIRCAM data. Correct for non-linearity, remove dark current, apply a flat field, destripe the images, apply a gain correction, remove the sky background, and combine the images into a pawprint. Create a catalogue of location and fluxes of objects in the pawprint and use that to calibrate the astrometry and photometry; apply the solution to individual calibrated science frames.

8.5.2 Input frames

Frame tag	Constraint	# files
OBJECT or OBJECT_EXTENDED	Required	$1 < n$
SKY_OFFSET	Optional	$1 < n$
MASTER_DARK	Required	1
MASTER_TWILIGHT_FLAT	Required	1
MASTER_CONF	Required	1
MASTER_READGAIN_TABLE	Required	1
MASTER_CHANNEL_TABLE	Required	1
PHOTCAL_TAB	Required	1
SCHLEGEL_MAP_NORTH	Required	1
SCHLEGEL_MAP_SOUTH	Required	1
MASTER_2MASS_CATALOGUE	Optional	1
MASTER_OBJMASK	Optional	1
MASTER_SKY	Optional	1

Note that if only one OBJECT or OBJECT_EXTENDED file is provided, a sky estimation is impossible. If two OBJECT or OBJECT_EXTENDED files are provided, a sky can be created in principle but it is not likely to be accurate.

8.5.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
destripe	bool	**true**, false	If true, destripe the raw images before processing
skyalgo	string	**auto**, master, none, pawsky_mask, pawsky_mask_pre, tilesky, tilesky_minus, pawsky_minus	To see details on the sky creation, run the recipe with preview_only = true. See §9.3 for more details.
preview_only	bool	**false**, true	If true, don't run recipe but instead print out a summary of how data will be processed
minphotom	int	**1**, $1 \leq n \leq 10^5$	During photometric calibration, at least minphotom number of standards must be used.

continued on next page

Parameter	Type	Values **default**,other	Description
`prettynames`	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.
`cdssearch`	bool	**false**, true	If `true`, retrieve subset of 2MASS catalogue from CDS for calibration
`stk_cat_ipix`	int	**5**, $1 \leq n \leq 1024$	min. number of pixels for catalogued objects in pawprint
`stk_cat_thresh`	float	**2.0**, $10^{-6} \leq x \leq 10^{10}$	detection threshold above sky in units of σ above bkg for catalogued objects in pawprint
`stk_cat_icrowd`	bool	**true**, false	if true, use deblending for catalogued objects in pawprint
`stk_cat_rcore`	float	**3.0**, $10^{-6} \leq x \leq 1024$	size of standard aperture in units of pixels for catalogued objects in pawprint
`stk_cat_nbsize`	int	**64**, $1 \leq x \leq 2048$	size of smoothing box when estimating bkg for catalogued objects in pawprint
`stk_lthr`	float	**5.0**, $10^{-6} \leq x \leq 10^{10}$	when stacking, reject pixel more than `stk_lthr` units of σ below sky bkg. Input files are scaled to a common background before rejection.
`stk_hthr`	float	**5.0**, $10^{-6} \leq x \leq 10^{10}$	when stacking, reject pixel more than `stk_hthr` units of σ above sky bkg. Input files are scaled to a common background before rejection.
`stk_method`	string	**linear**, nearest	when stacking, an RA/DEC is calculated for each input pixel. If `nearest`, the pixel in the output stack whose centre is nearest to that RA/DEC is given that input pixel. If `linear`, the exact pixel location of the RA/DEC on the output grid is calculated and the contribution from the input pixel is spread to the neighbouring 4 pixels that surround that point and is weighted by the fractional overlap.

continued on next page

Parameter	Type	Values **default**,other	Description
`stk_seeing`	bool	**false**, true	when stacking, weight each image by seeing value
`stk_fast`	string	**auto**, fast, slow	If set to `auto`, the stacking method is `slow` is there are more than `stk_nfst` input files. See §9.1.
`stk_nfst`	int	**30**, any int	See §9.1
`str_filt`	bool	**true**, false	If true, filter the stripe pattern
`psm_ipix`	int	**5**, $1 \leq n \leq 10^5$	for pawsky_mask sky estimation, min. number of pixels for objects
`psm_niter`	int	**5**, $1 \leq n \leq 100$	for pawsky_mask sky estimation, number of iterations
`psm_thresh`	float	**2.0**, $10^{-6} \leq x \leq 10^{10}$	for pawsky_mask sky estimation, detection threshold above sky in units of σ above bkg for objects
`psm_nbsize`	int	**50**, $1 \leq x \leq 2048$	for pawsky_mask sky estimation, size of smoothing box in pixels when estimating bkg for objects
`psm_smkern`	float	**2.0**, $10^{-6} \leq x \leq 5$	for pawsky_mask sky estimation, size of smoothing kernel in pixels
`cacheloc`	string	**.**, any string	directory where cache of calibration catalogue data ('.catcache') is stored relative to current working directory (.)
`magerrcut`	float	**100.0**, any float	objects in the photometric reference catalogue with a magnitude error greater than this value will be excluded from the calibration

continued on next page

Parameter	Type	Values **default**,other	Description

8.5.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
SIMPLE_IMAGE_SCI	exp_[N].fits	<FILE>_ex.fits	Calibrated image of each input OBJECT or OBJECT_EXTENDED file
SIMPLE_IMAGE_SKY	exp_[N].fits	<FILE>_ex.fits	Calibrated image of each input SKY_OFFSET file
MEAN_OFFSET_SKY	sky_[N].fits	sky_<FIRSTFILE>_ex.fits	Mean sky frame subtracted from science images
JITTERED_IMAGE_SCI	stack_[N].fits	<FIRSTFILE>_st.fits	Calibrated stack (pawprint) of all input science frames
CONFIDENCE_MAP_SCI	stack_conf_[N].fits	<FIRSTFILE>_st_conf.fits	Confidence map of JITTERED_IMAGE_SCI
OBJECT_CATALOGUE_SCI	stack_cat_[N].fits	<FIRSTFILE>_st_cat.fits	Catalogue of objects in each JITTERED_IMAGE_SCI

The index number [N] refers to the chronological order of the files in the SoF.

Note that <FIRSTFILE> is the first file listed in the SoF among the collection of files used to create that product.

8.5.5 Quality control parameters

The following QC parameters are found in each extension of the `SIMPLE_IMAGE_SCI` and `CONFIDENCE_MAP_SCI` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

The following QC parameters are found in each extension of the `JITTERED_IMAGE_SCI` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in each extension of the `OBJECT_CATALOGUE_SCI` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

8.6 vircam_science_postprocess

8.6.1 Description

Mosaic a list of VIRCAM pawprints/stacks and generate a catalogue. Optionally nebulise before generating the catalogue.

8.6.2 Input frames

Frame tag	Constraint	# files
`JITTERED_IMAGE_SCI`	Required	$1 < n$
`CONFIDENCE_MAP_SCI`	Required	Same as `JITTERED_IMAGE_SCI`

continued on next page

Frame tag	Constraint	# files
OBJECT_CATALOGUE_SCI	Required	Same as JITTERED_IMAGE_SCI
PHOTCAL_TAB	Required	1
SCHLEGEL_MAP_NORTH	Required	1
SCHLEGEL_MAP_SOUTH	Required	1
MASTER_2MASS_CATALOGUE	Optional	1

8.6.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
nebulise	bool	**false**, true	If true, object catalogue is derived from nebulised tile
grout	bool	**true**, false	If true, object catalogue is 'grouted', i.e. photometry is adjusted to account for variations between pawprints
minphotom	int	**20**, $1 \leq n \leq 10^5$	During photometric calibration, at least minphotom number of standards must be used.
prettynames	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.
cdssearch	bool	**false**, true	If true, retrieve subset of 2MASS catalogue from CDS for calibration
neb_medfilt	int	**101**, $11 \leq n \leq 2047$	Median filter size in pixels for nebuliser
neb_linfilt	int	**33**, $3 \leq n \leq 2047$	Linear filter size in pixels for nebuliser
cat_ipix	int	**4**, $1 \leq n \leq 10^5$	min. number of pixels for catalogued objects
cat_thresh	float	**1.25**, $0 < x \leq 10^{10}$	detection threshold above sky in units of σ above bkg for catalogued objects
cat_icrowd	bool	**true**, false	if true, use deblending for catalogued objects

continued on next page

Parameter	Type	Values **default**,other	Description
`cat_rcore`	float	**3.0**, $0 < x \leq 1024$	size of standard aperture in units of pixels for catalogued objects
`cat_nbsize`	int	**64**, $1 \leq x \leq 2048$	size of smoothing box in pixels when estimating bkg for catalogued objects
`cacheloc`	string	**.**, any string	directory where cache of calibration catalogue data ('.catcache') is stored relative to current working directory (.)
`magerrcut`	float	**100.0**, any float	objects in the photometric reference catalogue with a magnitude error greater than this value will be excluded from the calibration

8.6.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
TILED_IMAGE	tile_1.fits	<FIRSTFILE>_tl.fits	Calibrated tile made from all pawprints
TILED_CONFIDENCE_MAP	tile_conf_1.fits	<FIRSTFILE>_tl_conf.fits	Confidence map of TILED_IMAGE
OBJECT_CATALOGUE_SCI	tile_cat_1.fits	<FIRSTFILE>_tl_cat.fits	Catalogue of objects in TILED_IMAGE

Note that <FIRSTFILE> is the first file listed in the SoF among the collection of files used to create that product.

8.6.5 Quality control parameters

The following QC parameters are found in the `TILED_IMAGE` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in the `OBJECT_CATALOGUE_SCI` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in the `TILED_CONFIDENCE_MAP` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1

`QC.WCS_DCRVAL2`: [deg] change in crval2

`QC.WCS_DTHETA`: [deg] change in rotation

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot)

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

9 Recipe Algorithms

9.1 Stacking

The stacking module used by the VIRCAM pipeline recipes for the science observations has two modes of operation, a `slow` and a `fast` algorithm. The algorithm needs to stack not just the science data, but also the science variance data. It creates an output stack, output stack variance and an output stack confidence map. The volume of data that is read and written can make the memory required quite large and this can be increased further if the jitter offsets are large too.

The 'fast' algorithm can be used when stacking a small number of images . As the name implies it is usually pretty quick, but it is very greedy with memory. For stacking problems with more images it is better to use the slow algorithm. The recipes that use the stacking module also offer an *auto* mode. This allows the recipe to decide which algorithm to use (if the number of input files is less than `stk_nfst`, the `fast` mode is used).

As part of the stacking process, all images are placed onto the same photometric zeropoint before rejection. In addition, all input images are first scaled to the same exposure time (multiplicative factor) and then the background values are brought into agreement (through scalar addition).

When stacking, an RA/DEC is calculated for each input pixel. If `stk_method = nearest`, the pixel in the output stack whose centre is nearest to that RA/DEC is given that input pixel. If `stk_method = linear`, the exact pixel location of the RA/DEC on the output grid is calculated and the contribution from the input pixel is spread to the neighbouring 4 pixels that surround that point and is weighted by the fractional overlap.

Changes in throughput (i.e. magnitude zero point) and/or seeing between consecutive raw science frames within an OB are not taken into account as part of the stacking procedure. Users are advised to check for changes in seeing or photometric conditions in each image before stacking; no warning are issued by the recipe. Aberrant images may be discarded from the input list to avoid unexpected results.

9.2 Confidence maps

A confidence map is similar to a weight map. These are initially derived from the master twilight flat field for the detector and hence gives an indication of the relative quantum efficiency of each pixel, rather than an indication of the poisson uncertainty in the pixel flux. The map is stored as an integer array that has been normalised to a mean of 100. Bad pixels (either dead or hot) are given a confidence value of zero. If images are stacked, then the confidence map is propagated and can be used as a weight map or an exposure time map. Note that when median values are used as part of stacking, the variance of the median is a factor of $\sqrt{\pi/2}$ greater than the variance of the mean.

9.3 Sky Estimation

Estimating the sky background well is one of the hardest jobs in infrared imaging. The infrared sky varies both spatially and temporally over short time scales. The jittering of exposures is a method used to allow for the observational data to be self-calibrating with respect to the estimation of the background emission. The basic idea is the stack all object images without dithering but with rejection in order to remove all astronomical objects

from the background stack. However if astronomical objects are included in the observation that are larger than the typical jitter step, then it is possible that the rejection will fail and the background stack will be contaminated by emission from such objects.

For this reason the VIRCAM science recipe contains a number background estimation algorithms. Any of them can be chosen using the `skyalgo` command line switch. There is also a mode where `skyalgo=auto` where the recipe will choose what it believes to be the best algorithm. Users can see which algorithm is chosen by running the recipe with `preview_only = true`). The automatically selected algorithm is also reported in the `DRS.SKYALGO` keyword in each extension of the `MEAN_OFFSET_SKY` recipe product.

In the situation with large extended objects in the science observations, there is also the option to include offset sky exposures in the SoF. These are exposures that are well away from the object of interest and within the same OB as the science exposures. Below is a description of some of the algorithms supported by the `vircam_science_process` recipe.

Note that when using any algorithm (other than `none`), the recipes may create more than one sky frame from a set of input files. If the elapsed time between the first and last frame is more than $\approx$ 30 minutes, the set is split apart into chunks. A sky is created from each 'chunk' only if there are more than $\approx$ 8 members of each chunk. If not, some chunks are merged together to ensure a reasonable sky is created.

Also note that a 'running' sky algorithm is not implemented as an option. The pipeline is set up so that if there are long OBs, skys are made that cover periods of roughly 30 minutes each (so long as there are sufficient exposures).

`pawsky_mask`: The OB is of a simple jittered pawprint. The algorithm starts by doing a simple stack with rejection of all the science images to form an initial background image. The exposures are then all background corrected using this image and then combined using the jitter offsets to form a stacked image. An object detection is performed on this stacked image to create an object mask. The stacked image, initial sky background map and the sky corrected observations are thrown away. Now a new background map is formed by doing a stack of the input images with rejection, but this time also using the object mask we've just created. The procedure re-estimation of the background and masking object pixels continues until the number of masked pixels converges.

`pawsky_mask_pre`: This is the same as the pawsky_mask algorithm, except that a master object mask is given in the SoF. In this case there is no need to iterate. It is absolutely crucial for the object mask to (1) cover the whole area of sky covered by the observations and (2) to have an accurate world coordinate system represented in its FITS header.

`master`: If this option is chosen then a master background map should be included in the SoF. This must be used with extreme caution as a master sky is unlikely to be a good match for a given OB unless they are spatially and temporally close to each other.

`none`: If this is chosen, then no background correction will be done at all.

`auto`: If this is chosen then the recipe attempts to make a decision on which is the best algorithm based on what is in the SoF.

A filled tile of the sky requires a set of 6 pawprints to be taken. Each pawprint is separated by at least half a detector size from any other. This allows exposures from different pawprints to be combined into background

images where the only overlap of astronomical images we would expect would be extended sources whose size is greater than half a detector or chance overlaps of stars in crowded regions. There are two additional sky estimation algorithms that can be used when creating a tile:

`tilesky`: This method takes advantage of the scenario described above, where we have six pawprints, all offset by at least a half of a detector. The algorithm first combines a set of images, one from each pawprint, into an intermediate sky. This should form a set of intermediate background images (one for each exposure in a jitter sequence). These intermediate background images are then combined to form the final background image. Using the two rejection stages is very efficient at removing astronomical sources in the images. This is the recommended sky algorithm when six pawprints are present (and pawprints created with any number of jitters for each one).

`tilesky_minus`: This is a modification of the tilesky algorithm where a separate tilesky background image is produced for each science image. The background image for each science image is formed from all of the images in the tile with the exception of the science image to be corrected. The algorithm does not save each of the separate background images though. Rather, a representative tilesky background image is produced for the set of observations.

9.4 Object detection/catalogue generation

In order to provide quality control, and astrometric and photometric calibration information, the recipes generate detected object (i.e. stars, galaxies) catalogues for each target frame.

Objects are detected and parameterised using the processed images and confidence maps. A high-level summary of this process is:

- estimate the local sky background over the field and track any variations at adequate resolution to eventually remove them
- detect objects/blends of objects and keep a list of pixels belonging to each blend for further analysis (see [5] for details)
- parameterise the detected objects, i.e. perform astrometry, photometry and some sort of shape analysis.

Note that pixels are weighted in the catalogue generation according to their value in the confidence map. Hence if a pixel is marked as bad, then it is not included in the aperture flux. The number of bad pixels with an aperture is reported in the 'Error_bit_flag' column of the output table. The presence of bad pixels will also be reflected in the average confidence for the aperture (column 'Av_conf').

9.4.1 Background analysis and object detection

The possibly-varying sky background is estimated automatically, prior to object detection, using a combination of robust iteratively-clipped estimators.

Any variation in sky level over the frame is dealt with by forming a coarsely sampled background map grid. Within each background grid pixel (specified by recipe parameter `stk_cat_nbsize`), an iteratively k-sigma

clipped median value of 'sky' is computed based on the histogram of flux values within the grid pixel zone. A robust estimate of sigma can be computed using the Median of the Absolute Deviation (MAD) from the median. This will then be further processed to form the frame background map.

After removing the (possibly) varying background component, a similar robust estimate of the average sky level and sky noise per pixel can be made. This forms part of the quality control measures and also helps to robustly determine the detection threshold for object analysis.

Individual objects are detected using a standard matched filter approach. Since the only images difficult to locate are those marginally above the sky noise, assuming constant noise is a good approximation (after factoring in the confidence map information) and the majority of these objects will have a shape dominated by the point spread function (PSF), which thereby defines the filter to use.

9.4.2 Object parameterisation

The following image parameters are computed efficiently for each detected object:

Isophotal intensity: integrated flux within the boundary defined by the threshold level, i.e. the 0th object moment

$$I_{iso} = \sum_i I(x_i, y_i) \tag{1}$$

For objects where $I(x_i, y_i)$ is well described by a Gaussian, the isophotal intensity is related to the total intensity by the factor $(1 - I_t/I_p)^{-1}$, where I_p is the peak flux and I_t is the threshold level (all relative to sky).

Position: intensity-weighted location of object on the image, i.e. 1st moment:

$$x_0 = \sum_i xI(x_i, y_i)/I_{iso} \tag{2}$$

$$y_0 = \sum_i yI(x_i, y_i)/I_{iso} \tag{3}$$

Covariance matrix: triad of intensity-weighted 2nd moments used to estimate the eccentricity/ellipticity, position angle, and intensity-weighted size of an object:

$$\sigma_{xx} = \sum_i (x_i - x_0)^2 I(x_i, y_i)/I_{iso} \tag{4}$$

$$\sigma_{xy} = \sum_i (x_i - x_0)(y_i - y_0) I(x_i, y_i)/I_{iso} \tag{5}$$

$$\sigma_{yy} = \sum_i (y_i - y_0)^2 I(x_i, y_i)/I_{iso} \tag{6}$$

The shape parameters are derived by considering these quantities in relation to an elliptical Gaussian function with the same 2nd moments. The scale size ($\sqrt{\sigma_{rr}}$) is computed with $\sigma_{rr} = \sigma_{xx} + \sigma yy$. The eccentricity is $f = \sqrt{(\sigma_{xx} - \sigma_{yy})^2 + 4\sigma_{xy}}/\sigma_{rr}$. The position angle θ is computed with $tan(2\theta) = 2\sigma_{xy}/(\sigma_{yy} - \sigma_{xx})$. The ellipticity ($e$) is simpler to interpret for estimation potential image distortions (e.g. trailing) and is related to the eccentricity by $e = 1 - \sqrt{(1-f)/(1+f)}$.

Areal profile: a variant on the radial profile, which measures the area of an object at various intensity levels. Unlike a radial profile, which needs a prior estimate of the object centre, the areal profile provides a single-pass estimate of the object area at m discrete intensity levels $T + p_j$ where $p_j; j = (1...m)$ are intensity levels relative to threshold T (usually spaced logarithmically to give even sampling). The threshold T can be specified using the recipe parameter `stk_cat_thresh`.

Peak height: a useful related addition to the areal profile information and is defined as $I_p = max(I(x_i, y_i))$ or alternatively measured by extrapolation from the areal profile if the image is saturated.

Seeing: The areal profile provides a direct method to estimate the seeing of objects in an image by enabling the average area of stellar images (point sources) at half the peak height ($< A >$) to be estimated. The seeing, or FWHM, is then given by $FWHM = 2\sqrt{< A > /\pi}$.

Aperture flux: a series of aperture fluxes are required for object morphological classification. Aperture flux is defined as the integrated flux with some radius r of the object centre:

$$I_{ap} = \sum_{i \in r}^{N} I_i - N \times sky \tag{7}$$

where boundary pixels are weighted pro-rate (soft-edged aperture photometry). A series of these is used to define the curve-of-growth (I_{ap} vs. r) for each object.

The error in the aperture flux is calculated as:

$$\sigma_{ap} = \sqrt{I_{ap}/gain + \pi r^2(\sigma^2_{globalsky} + \sigma^2_{localsky})} \tag{8}$$

where $\sigma_{localsky}$ is a robust estimate of the sky variance among adjacent cells (defined by `stk_cat_nbsize`), $\sigma_{globalsky}$ is a robust estimate of the sky variance over the whole image, and the $gain$ is the value calculated by the `detector_noise` recipe.

To convert aperture fluxes to magnitudes, an aperture correction must be applied. This correction is calculated from the curve-of-growth and accounts for missing flux in a particular aperture. For example, the magnitude of an object using the flux in the third bin ($i = 3$) may be calculated as

$$m = \texttt{MAGZPT} - 2.5\texttt{log}_{10}(\texttt{APER_FLUX_3}/\texttt{EXPTIME}) - \texttt{APCOR3}) \tag{9}$$

where `MAGZPT, EXPTIME,` and `APCOR3` appear in the FITS header of the catalogue file; `APER_FLUX_3` is a column in the FITS binary table of the catalogue file.

For overlapping objects that are deblended, the boundaries are in practice simultaneously fitted top-hat functions (to minimise the effects of crowding). Object external to the blend are also flagged and not included in the large radius summations.

9.4.3 Morphological classification

The recipes produce a series of background-corrected flux measures for each object in a set of 'soft-edged' apertures of radius $r/2, r/\sqrt{2}, r, r\sqrt{2}, 2r, ...12r$, where r is specified by the recipe parameter `stk_cat_rcore`. Generally it is recommended that `stk_cat_rcore` be fixed as the median seeing for the site + telescope

+ camera. For VIRCAM, the default value of `stk_cat_rcore` is 3.0 pixels (1.02″). The average curve-of-growth for stellar objects is used to define automatically an aperture correction for each aperture used and also forms the basis for object morphological classification (required for isolating stellar images for seeing and trailing quality control).

The curve-of-growth of the flux for each object is compared with that derived from the (self-defining) locus of stellar objects, and combined with information on the ellipticity of each object, to generate the classification statistic. This statistic is designed to preserve information on the 'sharpness' of the object profile and is re-normalised, as a function of magnitude, to produce the equivalent of an $N(0,1)$ measure, i.e. a normalised Gaussian of zero-mean and unit variance. Objects lying within 2-3σ are generally flagged as stellar images, those below $\approx 2\sigma$ (i.e. sharper) as noise-like, and those above 2-3σ (i.e. more diffuse) as non-stellar. A by-product of the curve-of-growth analysis is the estimate of the average PSF aperture correction for each detector.

9.5 Nebulising

For observations with a large amount of nebulosity it can be very difficult to get a good estimate for the sky background. This is where the *nebuliser* can be useful. The nebuliser calculates sliding means and medians of an image to create a background image, which is essentially an unsharp mask. This image is subtracted from the original to produce an image with greatly reduced nebulosity.

If the user chooses to use the nebuliser in one of the science recipes, then the stacked image is corrected for nebulosity. The nebulised image is used to create the source catalogue and then thrown away. The output images will be the un-nebulised stack with its accompanying confidence and variance map along with the source catalogue derived from the nebulised image.

9.6 Destriping

The VIRCAM detectors suffer from an artefact called 'striping'. It is a form of pickup noise that comes from the controllers. Because a single controller is connected to a row of four detectors, all four detectors in a row will have the same stripe pattern. The pattern itself is low level (usually < 10 ADU) and is 10-20 pixels in breadth, as show in Figure 9.1 below. It is only evident in dark corrected and flat fielded images and may be corrected as part of the `vircam_science_process` recipe. If the recipe parameter `destripe = true`, a robust median filter algorithm incorporating data from each 'row' of detectors is used to remove this effect.

Destriping occurs after an image has been corrected for dark current, flat-fielding, non-linearity, and spatial variations in the sky emission. First, an iteratively clipped, robust median is calculated for each row (constant y value) on every detector. Any large-scale variations in the median (along the y-direction) are suppressed by replacing those values with the median value of those medians over a $\sim$ 100 pixel wide window. This defines a stripe 'profile' for that detector. A set of four profiles (one for each detector that share the same controller) is combined into one profile by calculating the median value of each profile (one value for each y-value). The value of each pixel in this master profile is then subtracted from every pixel in the same row among the four constituent images. This is repeated for each 'row' of detectors.

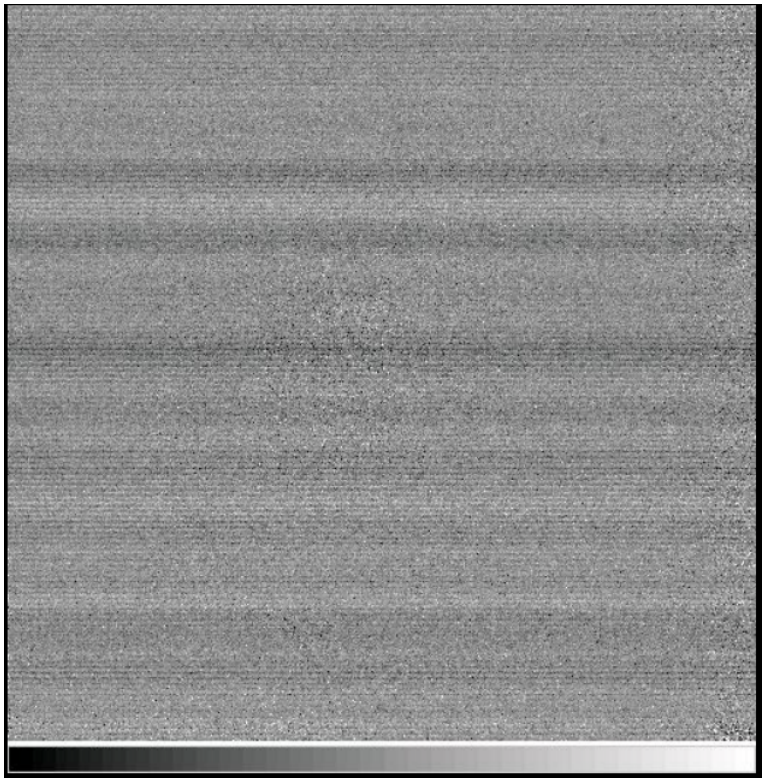

Figure 9.1: Example of ‘striping’ in VIRCAM.

9.7 Grouting

Users are advised to look at the seeing and magnitude zero point estimates in the pawprints before creating a tile. If these values vary significantly among the pawprints, the resultant tile may not be scientifically useful. The 'grouting' option in `vircam_science_postprocess` attempts to correct for this effect.

Each detector potentially has a different PSF and together with the varying seeing conditions that can happen during the observation of a tile this means that there are potentially 96 different PSFs which contribute to a single tile (some of which are combined in the overlap areas). In order to compute the correction for the aperture fluxes we normally assume that the PSF does not vary across the tile. This will inevitably introduce spatial photometric distortions at the level of a few % over the tile. In variable seeing conditions this effect will be significantly worse and can produce systematic spatial offsets of 10-20% or more in the photometry.

The process of 'grouting' tracks and post-corrects the variable flux within the main apertures for tile catalogues (tile images are unaffected). The grouting fix takes as input the tile catalogue and all the relevant stacked pawprint catalogues and confidence maps. It then computes differential aperture corrections at the location of each detected object, weighted by the confidence map that was used to drive the tiling of the image. The individual detector aperture corrections (#1-7), 96 of them for each aperture, define the differential aperture corrections i.e. the difference with respect to the median for the whole tile. All fluxes and associated errors for apertures #1-7 are corrected; larger apertures are negligibly affect by seeing variations and are currently left alone (ditto the other fluxes). The updated catalogue then has to be re-classified and re-photometrically calibrated. The correction software optionally can fix aberrant individual input detector aperture corrections and also adjust for changes in MAGZPT, i.e. caused by extinction variations, during the pawprint observations.

Grouting tests with VHS data and show significant improvements in the colour-magnitude diagrams and two-colour diagrams for the $\approx$ 20% of the data affected by large seeing variations during the OB. This manifests as either multiple stellar locii or very broadened stellar sequences. After grouting the incidence of problem tile catalogues drops by two orders of magnitude while leaving the other (good) 80% of tiles unaffected.

A File Formats

A.1 Static Calibration Files

The REFERENCE darks and flats are pipeline products of vircam_dark_combine and vircam_twilight_flat_combine, respectively. After creation, these files have the keyword PRO.CATG inserted into the PHU of each file using the supplied update_procatg.py script. The values of these keywords are REFERENCE_DARK and REFERENCE_TWILIGHT_FLAT, respectively.

The MASTER_READGAIN_TABLE file detector_noise.fits is the product of vircam_detector_noise. The MASTER_CHANNEL_TABLE file master_chan_tab.fits is the product of vircam_linearity_analyse.

The CHANNEL_TABLE_INIT file chan_tab_init.fits is also a product of vircam_linearity_analyse. However, this file has the keyword PRO.CATG modified in the PHU using the supplied update_procatg.py script. The values of this keyword is changed to CHANNEL_TABLE_INIT. Note that when this file is provided in a recipe SoF, only the first five columns are used; they describe the x/y coordinates of each VIRCAM detector channel.

The dust map files (SFD_dust_4096_ngp.fits and SFD_dust_4096_ngp.fits) are similar to the original FITS files provided by [7]. However, these files have the keyword PRO.CATG inserted into the PHU of each file using the supplied update_procatg.py script. The values of these keywords are SCHLEGEL_MAP_NORTH and SCHLEGEL_MAP_SOUTH, respectively.

The photometric calibration table file (vircam_phot.fits) is used to transform the instrumental magnitudes to a calibrated system. It is a binary FITS table with one extension per system. If using a local copy of a catalogue (e.g. 2MASS), an extension **must** be present that matches the name of the extension in the photometric catalogue index file (e.g. 'casu_2mass'). If using CDS cdssearch = true, one extension **must** be named '2mass'.

A description of the columns for each extension is:

Column	Name	Type	Description
1	filter_name	char	Name of the filter. It must match the value of the INS.FILT?.NAME keyword from the data being calibrated.
2	atm_extcoef	float (1E)	The extinction coefficient for airmass of unity for the given filter.
3	mag_offset	float (1E)	A scalar value to be added to the instrumental magnitude once the colour equation has been applied.

continued on next page

Column	Name	Type	Description
4	coleq_columns	char	The name(s) of magnitude columns from the photometric standards catalogue to be used in the colour equations, e.g. 'Jmag'. The names must be separated by a comma.
5	coleq_errcols	char	The name(s) of magnitude error columns from the photometric standards catalogue, e.g. 'e_Jmag'. There must be one name for each of the column names given in the coleq_columns field. Any objects with an error in any of the coleq_errcols greater than the value specified by the recipe parameter `magerrcut` will be excluded from the calibration. The names must be separated by a comma.
6	coleq_coefs	char	The colour equation coefficients. There must be one coefficient for each of the column names given in the coleq_columns field. This should be formatted as a sequence of floating point numbers separated by commas.
7	gal_extcoef	float (1E)	Multiply the E(B-V) estimate from the Schlegel maps by `gal_extcoef` to give the total galactic absorption in `filter_name`
8	default_zp	float (1E)	If the photometric calibration fails, this default zero point value is used.
9	default_zp_err	float (1E)	If photometric calibration fails, this default error on the zero point value is used.

A.2 Astrometric and Photometric Catalogue Files

A user may choose to calibrate their data using data on disk in the form of index files and catalogue data files. An index file is a map between celestial coordinates and the filename where the standard object data resides. It is provided in a SoF to tell a recipe where to find the standards to use in the calibration.

The index file is a binary FITS table where the extension name describes the catalogue, e.g. 'casu_2mass'. The columns are:

Column	Name	Type	Description

continued on next page

Column	Name	Type	Description
1	`filename`	char	Name of the file where data reside over a particular coordinate range. The file (or a link to it) must be found in the same directory as the index file.
2	`ramin`	float (1E)	Objects in `filename` have $\alpha >$ `ramin`.
3	`ramax`	float (1E)	Objects in `filename` have $\alpha \leq$ `ramax`
4	`decmin`	float (1E)	Objects in `filename` have $\delta >$ `decmin`
5	`decmax`	float (1E)	Objects in `filename` have $\delta \leq$ `decmax`

Note that all floating point numbers **must** be represented with 4-bytes (1E).

The catalogue data files are also binary FITS files. It must contain the columns described below; other columns are allowed but ignored. Note that the column names must appear as described below:

Column	Name	Type	Description
1	`RA`	float (1D)	Right ascension of the object (J2000). This **must** be represented as an 8-byte floating point number.
2	`Dec`	float (1D)	Declination of the object (J2000). This **must** be represented as an 8-byte floating point number.
3	`<mag>`	float (1E)	Magnitude of object in a particular filter. The name of this column must match one of those in the `coleq_columns` column in the photometric calibration table, e.g. 'Jmag'. There must be one `<mag>` column for each unique value of `coleq_columns`. The data in this column **must** be represented as a 4-byte floating point number. NULL values are allowed.
4	`e_<mag>`	float (1E)	Error in magnitude of object in a particular filter. The value of `<mag>` must match one of those in the `coleq_columns` column in the photometric calibration table, e.g. 'e_Jmag'. There must be one `e_<mag>` column for each unique value of `coleq_columns`. The data in this column **must** be represented as a 4-byte floating point number. NULL values are allowed.

continued on next page

Column	Name	Type	Description

A.3 Recipe Product Catalogues

The derived object catalogues are stored in multi-extension FITS files as binary tables, one for each image extension. Each detected object has an attached set of descriptors, forming the columns of the binary table, and summarising derived position, shape and intensity information (see section 9.4 for more details).

The following columns are present:

Column	Name	Description
1	Sequence_number	Running number for ease of reference, in strict order of image detections
2	Isophotal_flux	Standard definition of summed flux within detection isophote, apart from detection filter which is used to define pixel connectivity and hence which pixels to include. This helps to reduce edge effects for all isophotally derived parameters.
3 4 5 6	X_coordinate X_coordinate_err Y_coordinate Y_coordinate_err	The x, y intensity-weighted isophotal centre of gravity coordinates and errors with (1, 1) defined to be the centre of the first active pixel in the image array.
7 8 9	Gaussian_sigma Ellipticity Position_angle	Derived from second moment parameters. Equivalence between parameters and generalised elliptical Gaussian distribution is used to derive sigma = $(\sigma_a^2 + \sigma_b^2)^{1/2}$, ellipticity = $1.0 - \sigma_a/\sigma_b$, and position angle = angle of ellipse major axis wrt x axis in degrees
10 11 12 13 14 15 16 17	Areal_1_profile Areal_2_profile Areal_3_profile Areal_4_profile Areal_5_profile Areal_6_profile Areal_7_profile Areal_8_profile	The number of pixels above a series of threshold levels, relative to local sky. The levels are set at T, 2T, 4T, 8T, 16T, 32T, 64T and 128T where T is the analysis threshold (`stk_cat_thresh`). These can be thought of as a sort of poor man's radial profile. Note that for deblended, i.e. overlapping objects, only the first areal profile is computed and the rest are set to -1 (flagging the difficulty of computing accurate profiles). For blended images, Areal profile 8 is used to flag the start of the sequence of the deblended components by setting the first in the sequence to 0.
18 19	Peak_height Peak_height_err	Peak intensity and its error in ADU relative to local value of sky. Equivalent to zeroth-order aperture flux.
20 21 22 23 24 25 26 27	Aper_flux_1 Aper_flux_1_err Aper_flux_2 Aper_flux_2_err Aper_flux_3 Aper_flux_3_err Aper_flux_4 Aper_flux_4_err	Flux and error within a specified radial aperture where r_{core} =`stk_cat_rcore` and different rows correspond to (0.5, 1/$\sqrt{2}$, 1, $\sqrt{2}$, 2, 2$\sqrt{2}$, 4, 5, 6, 7, 8, 10, and 12) times r_{core}. These are a series of different radii soft-edged apertures designed to adequately sample the curve-of-growth of the majority of objects and to provide fixed-sized aperture fluxes for all objects. For example, in $\approx 0.8''$seeing, an r_{core} = $1''$aperture contains roughly 75% of the total flux of stellar images. The aperture fluxes are sky-corrected integrals (summations) with a soft-edge (i.e. pro-rata flux division for boundary pixels). However, for overlapping objects they are more subtle than this since they are in practice simultaneously fitted top-hat functions, to minimise the effects of crowding. Objects external to the blend are also flagged and not included in the large radius

continued on next page

Column	Name	Description
28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45	Aper_flux_5 Aper_flux_5_err Aper_flux_6 Aper_flux_6_err Aper_flux_7 Aper_flux_7_err Aper_flux_8 Aper_flux_8_err Aper_flux_9 Aper_flux_9_err Aper_flux_10 Aper_flux_10_err Aper_flux_11 Aper_flux_11_err Aper_flux_12 Aper_flux_12_err Aper_flux_13 Aper_flux_13_err	
46	Petr_radius	Petrosian radius r_p in pixels as defined in Yasuda, et al. 2001, AJ, 112, 1104.
47	Kron_radius	Kron radius r_k in pixels as defined by Bertin and Arnouts 1996, A&A Supp, 117, 393.
48	Half_radius	r_h estimate of object half-light radius.
49 50	Petr_flux Petr_flux_err	Petrosian flux and error to $2r_p$.
51 52	Kron_flux Kron_flux_err	Kron flux and error to $2r_k$.
53 54	Half_flux Half_flux_err	flux and error within half-light radius.
55	Error_bit_flag	Number of bad pixels within aperture of radius r_{core} - note this can be fractional due to soft-edged apertures
56	Sky_level	Local interpolated sky level from background tracker
57	Sky_rms	Local estimate of variation in sky level around object

continued on next page

Column	Name	Description
58	Av_conf	Average confidence level within r_{core} aperture. Useful for spotting spurious outliers in various parameter selection spaces
59 60	RA Dec	Single-precision (4 byte) RA and Dec of each object in degrees (only accurate to 50mas!). Astrometry should be derived more precisely from WCS in header and XY in columns 5 & 6.
61	Classification	Simple flag indicating most probable classification for object: [-2: Object is compact (maybe stellar), -1: Object is stellar, 0: Object is noise, 1: Object is non-stellar]. Saturated objects can be flagged by comparing the peak height + local sky with the `SATURATE` keyword in the header.
62	Statistic	An equivalent N(0,1) measure of how stellar-like an image is. It is used in deriving the classification (column 61) in a 'necessary but not sufficient' sense. This statistic is computed from a discrete curve-of-growth analysis from the peak and aperture fluxes and also factors in ellipticity information. The stellar locus is used to define the 'mean' and 'sigma' as a function of magnitude such that the 'statistic' can be normalised to an approximate N(0,1) distribution. See Irwin et al 1994 (SPIE 5493 411) for more details.
63–80	blank	