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

NACO Pipeline User Manual

VLT-MAN-ESO-19500-3993

Issue 4.4.12

Date 19th May 2023

	Name	Date	Signature
Prepared:	L.K.Lundin	19th May 2023	
Approved:	P.Ballester		
Released:	M. Peron		

No content on this page

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.0	2007-06-29	All	For NACO pipeline 3.6.2
1.1	2012-05-12	All	For NACO pipeline 4.3.1
4.4.12	19th May 2023	All	For NACO pipeline 4.4.12

Contents

1 Introduction

1.1 Purpose

The NACO 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), in the quick-look assessment of data, in the generation of master calibration data, in the reduction of scientific exposures, and in the data quality control. Additionally, the NACO 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 NACO data reduction sequence with the NACO pipeline.

This manual is a complete description of the data reduction recipes offered by the NACO pipeline, reflecting the status of the NACO pipeline as of 19th May 2023 (version 4.4.12).

1.2 Acknowledgements

Since the beginning of the NACO operations in 2002, several people have been involved in the NACO pipeline project. Nicolas Devillard developed the first version of the pipeline and Yves Jung brought substantial contributions to that version. Yves Jung later wrote a new CPL-based version of the pipeline, and in late 2005 the maintenance of this pipeline was passed on to Lars Lundin.

From Paranal, the instrument scientists, first Chris Lidman and then Nancy Ageorges have contributed with valuable feed-back and ideas for improvement. In the operations team, Wolfgang Hummel at the beginning, and now Danuta Dobrzycka have acted as in depth testers, ensuring proper results from the different recipes.

1.3 Scope

This document describes the NACO pipeline used at ESO-Garching and ESO-Paranal for the purpose of data assessment and data quality control.

Updated versions of the present document may be found on [15]. For general information about the current instrument pipelines status we remind the user of [6]. Quality control information are at [5].

Additional information on QFITS, the Common Pipeline Library (CPL) and ESOREX can be found respectively at [10], [13]. The Gasgano tool is described in [14]. A description of the instrument is in [7]. The NACO instrument user manual is in [8] while results of Science Verifications (SV) are at [2].

1.4 Reference Documents and Applicable Documents

[1] ESO. *NACO Calibration Plan*. VLT-PLA-ESO-14200-2664.

[2] ESO, *http://www.eso.org/science/vltsv/nacosv/*. *NACO Science Verification home page*. 9

[3] ESO. *VLT Data Flow System Specifications for Pipeline and Quality Control*. VLT-SPE-ESO-19600-1233.

[4] ESO/DMO. *ESO DICB – Data Interface Control Document*. GEN-SPE-ESO-19400-0794 (3.0). 22, 23, 24

[5] ESO/DMO/DFO, *http://www.eso.org/observing/dfo/quality/*. *ESO-Data Flow Operation home page*. 9, 29

[6] ESO/DMO/DFO, *http://www.eso.org/observing/dfo/quality/*. *NACO Pipeline Current Status*. 9

[7] ESO/INS, *http://www.eso.org/instruments/naco/*. *NACO instrument home page*. 9, 12

[8] ESO/INS, *http://www.eso.org/instruments/naco/doc/*. *NACO instrument home page*. 9, 12

[9] ESO/SDD/DFS, *http://www.eso.org/observing/cpl/download.html*. *Common Pipeline Library User Manual*. VLT-MAN-ESO-19500-2720.

[10] ESO/SDD/DFS, *http://www.eso.org/cpl/*. *CPL home page*. 9, 39, 40, 41, 42

[11] ESO/SDD/DFS. *Deliverables Specification*. VLT-SPE-ESO-19000-1618 (2.0).

[12] ESO/SDD/DFS. *DFS Pipeline & Quality Control – User Manual*. VLT-MAN-ESO-19500-1619.

[13] ESO/SDD/DFS, *http://www.eso.org/cpl/esorex.html*. *ESOREX home page*. 9, 16, 18

[14] ESO/SDD/DFS, *http://www.eso.org/gasgano/*. *Gasgano User's Manual*. VLT-PRO-ESO-19000-1932. 9, 11, 15, 22

[15] ESO/SDD/DFS, *http://www.eso.org/pipelines*. *NACO Pipeline Web Page*. 9

2 Overview

In collaboration with instrument consortia, the Data Flow Systems Department (DFS) of the Data Management and Operation Division is implementing data reduction pipelines for the most commonly used VLT/VLTI instrument modes. These data reduction pipelines have the following three main purposes:

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

Master calibration product creation: pipelines are used to produce master calibration products (e.g. , combined dark frames, super-flats, wavelength dispersion solutions).

Science product creation: using pipeline-generated master calibration products, science products are produced for the supported instrument modes (e.g. , combined NACO jitter stacks; bias-corrected, flat-fielded FORS images, wavelength-calibrated UVES spectra). The accuracy of the science products is limited by the quality of the available master calibration products and by the algorithmic implementation of the pipelines themselves. In particular, adopted automatic reduction strategies may not be suitable or optimal for all scientific goals.

Instrument pipelines consist of a set of data processing modules that can be called from the command line, from the automatic data management tools available on Paranal or from Gasgano.

ESO offers two front-end applications for launching pipeline recipes, *Gasgano* [14] and *EsoRex*, both included in the pipeline distribution (see Appendix A, page 46). These applications can also be downloaded separately from *http://www.eso.org/gasgano* and *http://www.eso.org/cpl/esorex.html*. An illustrated introduction to Gasgano is provided in the "Quick Start" Section of this manual (see page 13).

The NACO instrument and the different types of NACO raw frames and auxiliary data are described in Sections 3, 6.1, and 7.

A brief introduction to the usage of the available reduction recipes using Gasgano or EsoRex is presented in Section 4.

In section 5 we advice the user about known data reduction problems.

An overview of the data reduction, what are the input data, and the recipes involved in the calibration cascade is provided in Section 8.

More details on what are inputs, products, quality control measured quantities, and controlling parameters of each recipe is given in Section 9.

More detailed descriptions of the data reduction algorithms used by the individual pipeline recipes can be found in Section 10.

In Appendix A the installation of the NACO pipeline recipes is described.

3 NACO Instrument Description

NACO (NAOS-CONICA) is installed at the Nasmyth B focus of UT4. It provides adaptive optics assisted imaging, imaging polarimetry, coronography and spectroscopy, in the 1-5 microns range. It also includes a Fabry-Perot unit in the 2-2.5 microns range.

The adaptive optics system, NAOS, is equipped with both visible and infrared wavefront sensors. It contains 5 dichroics which split the light from the telescope between CONICA and one of the NAOS wavefront sensors.

CONICA is the infrared camera and spectrometer attached to NAOS and is equipped with an Aladdin 3 1026 $\times$ 1024 pixel InSb array detector.

It contains several wheels carrying masks/slits (including focal plane coronographic masks), filters, polarizing elements, grisms and several cameras allowing diffraction limited sampling across the full wavelength range. CONICA is not intented to be operated without NAOS.

Figure 3.0.1 shows a picture of the instrument mounted on the fourth unit telescope (Yepun) on the VLT.

Figure 3.0.1: *NACO at UT4.*

See [7], [8] for a more complete description of NACO.

4 Quick start

This section describes the most immediate usage of the NACO pipeline recipes.

4.1 NACO pipeline recipes

The current NACO pipeline supports imaging mode only. It does this with a set of 9 stand-alone recipes involved in the data reduction cascade.

The 9 recipes in are:

naco_img_jitter: Main reconstruction routine, including dark correction, flatfield calibration, bad pixels cleaning, and images correlation and recombination.

naco_img_dark: Creates a master dark frame to calibrate the dark current.

naco_img_twflat: Uses a set of twilight images to create a master flat field frame and a bad pixels map.

naco_img_lampflat: Uses a set of lamp images to create a master flat field frame and a bad pixels map.

naco_img_strehl: Strehl computation with error estimate using standard star observations.

naco_img_zpoint: Zero point measurements using standard star observations.

naco_img_checkfocus: Checks the focus.

naco_img_slitpos: Makes a precise slit analysis to help the slit positioning on the instrument.

naco_img_detlin: Measures the linearity of the detector.

4.2 An introduction to Gasgano and EsoRex

Before being able to call pipeline recipes on a set of data, the data must be opportunely classified, and associated with the appropriate calibrations. The *Data Classification* consists of tasks such as: "What kind of data am I?", e.g. , DARK, "to which group do I belong?", *e.g.*, to a particular Observation Block or template. *Data Association* is the process of selecting appropriate calibration data for the reduction of a set of raw science frames. Typically, a set of frames can be associated if they share a number of properties, such as instrument and detector configuration. As all the required information is stored in the FITS headers, data association is based on a set of keywords (called "association keywords") and is specific to each type of calibration.

The process of data classification and association is known as data organisation. The *DO Category* is the label assigned to a data type as a result of data classification.

An instrument pipeline consists of a set of data processing modules that can be called from different host applications, either from the command line with *Esorex*, from the automatic data management tools available at Paranal, or from the graphical *Gasgano* tool.

Gasgano is a data management tool that simplifies the data organisation process, offering automatic data classification and making the data association easier (*even if automatic association of frames is not yet provided*).

Gasgano determines the classification of a file by applying an instrument specific rule, while users must provide this information to the recipes when they are executed manually using *Esorex* from the command line. In addition, *Gasgano* allows the user to execute directly the pipeline recipes on a set of selected files.

4.2.1 Using Gasgano

To get familiar with the NACO pipeline recipes and their usage, it is advisable to begin with *Gasgano*, because it provides a complete graphic interface for data browsing, classification and association, and offers several other utilities such as easy access to recipes documentation and preferred data display tools.

Gasgano can be started from the Command Line Interface in the following way:

```
gasgano &
```

Figure 4.2.1 shows the *Gasgano* main window.

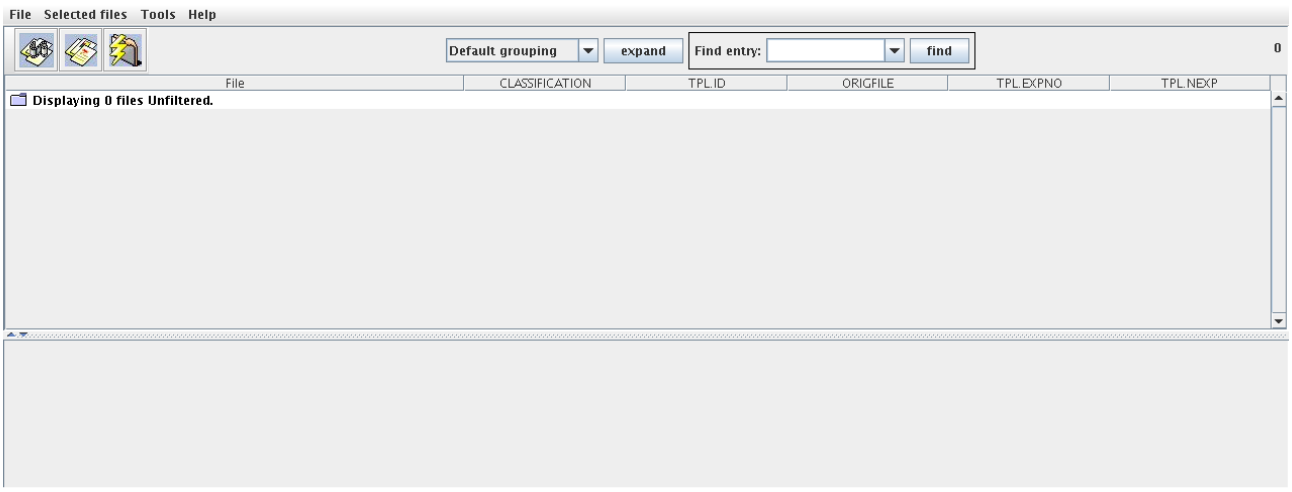

Figure 4.2.1: *The Gasgano main window.*

With the pull-down-menu *File->Add/Remove Files* directories containing NACO data can be added, as shown in figure 4.2.2 on the next page. Figure 4.2.3 on page 16 shows the example data set distributed with the public release of the NACO pipeline with their classification as CAL_DARK.

The data are hierarchically organised as preferred by the user. After each file name are shown the classification, the template id, the original filename, the template exposure number and the number of exposures in the template.

More information about a single frame can be obtained by clicking on its name: the corresponding FITS file header will be displayed on the bottom panel, where specific keywords can be opportunely filtered and searched. Images and tables may be easily displayed using the viewers specified in the appropriate *Preferences* fields.

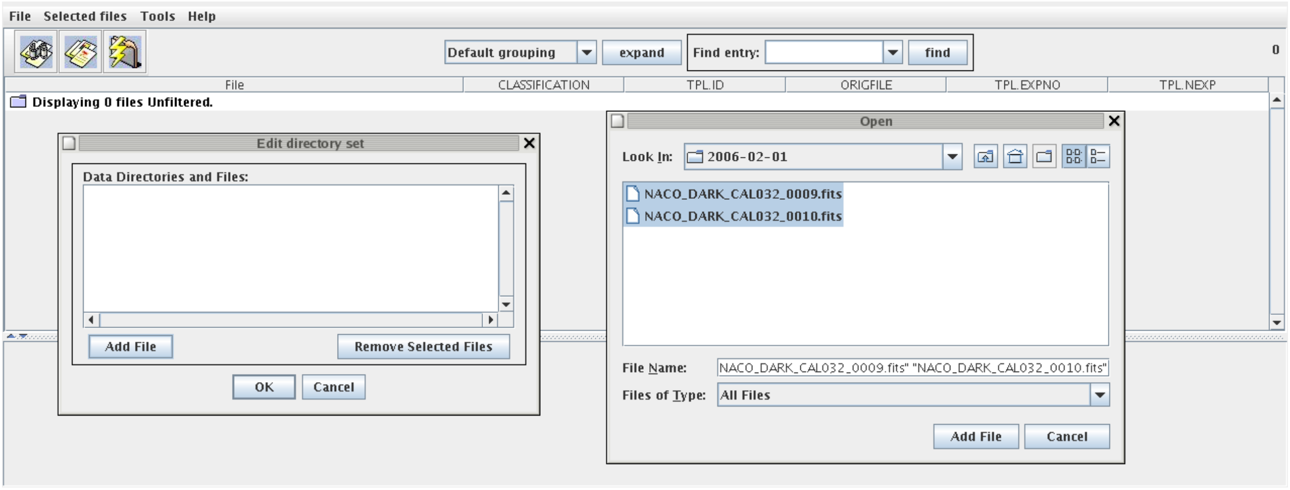

Figure 4.2.2: *The Gasgano main window with the file loader window on top showing two NACO files marked for loading.*

Frames can be selected from the main window with a <CTRL>-left-click for processing by the appropriate recipe: on Figure 4.2.4 on page 17, a set of dark calibration FITS-files have been selected and the pull-down-menu with the NACO recipes is shown.

Selecting the appropriate recipe, *naco_img_dark* , will open a *Gasgano* recipe execution window (see Figure 4.2.5 on page 19), having all the specified files listed in its *Input Frames* panel.

Help about the recipe may be obtained from the *Help* menu. Before launching the recipe, its parameters may be modified on the *Parameters* panel (on top). The window contents might be saved for later use by selecting the *Save Current Settings* entry from the *File* menu, as shown in figure.

At this point the recipe can be launched by pressing the *Execute* button. Messages from the running recipe will appear on the *Log Messages* panel at bottom, and in case of successful completion the products will be listed on the *Output Frames* panel, where they can be easily viewed and located back on the Gasgano main window. The succesful processing of the example data can be seen in figure 4.2.6 on page 20.

Please refer to the *Gasgano User's Manual* [14] for a more complete description of the *Gasgano* interface.

4.2.2 Using EsoRex

EsoRex is a command line utility for running pipeline recipes. It may be embedded by users into data reduction scripts for the automation of processing tasks. On the other side, *EsoRex* doesn't offer all the facilities available with *Gasgano*, and the user must classify and associate the data using the information contained in the FITS header keywords (see Section 6.1, page 22). The user should also take care of defining the input set-of-frames and the appropriate configuration parameters for each recipe run:

The set-of-frames: Each pipeline recipe is run on a set of input FITS data files. When using *EsoRex* the file names must be listed together with their DO category in an ASCII file, the *set-of-frames* (SOF), that is

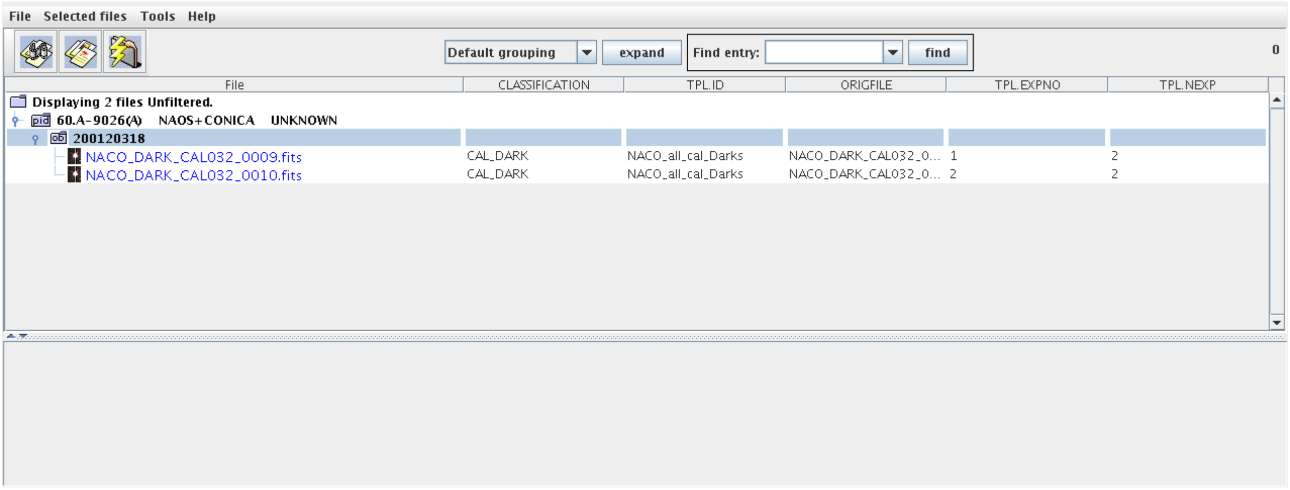

Figure 4.2.3: *The Gasgano main window with the example FITS files loaded.*

required when launching a recipe. [1]

Here is an example of SOF, valid for the *naco_img_jitter* recipe:

```
/data/2002-03-27/NACO.2002-03-27T08:30:20.560.fits   IM_JITTER_OBJ
/data/2002-03-27/NACO.2002-03-27T08:31:45.132.fits   IM_JITTER_OBJ
/data/2002-03-27/NACO.2002-03-27T08:33:05.673.fits   IM_JITTER_OBJ
/data/2002-03-27/NACO.2002-03-27T08:34:27.519.fits   IM_JITTER_OBJ
/data/calib/flat.fits                                MASTER_IMG_FLAT
/data/calib/bpm.fits                                 BPM
```

Note that the NACO pipeline recipes do not verify the correctness of the DO category specified by the user in the SOF. The reason of this lack of control is that NACO recipes are just one component of the complete pipeline running on Paranal, where the task of data classification and association is carried out by separate applications. Using *Gasgano* as an interface to the pipeline recipes will however ensure a correct classification of all the data frames, assigning the appropriate DO category to each one of them (see section 4.2.1 on page 14).

A recipe handling an incorrect SOF may stop or display unclear error messages at best. In the worst cases, the recipe would apparently run without any problem, producing results that may look reasonable, but are actually flawed.

EsoRex syntax: The basic syntax to use ESOREX is the following:

esorex [esorex_options] recipe_name [recipe_options] set_of_frames

To get more information on how to customise ESOREX (see also [13]) run the command:

esorex - -help

To generate a configuration file esorex.rc in the directory $HOME/.esorex run the command:

[1] The set-of-frames corresponds to the *Input Frames* panel of the *Gasgano* recipe execution window (see Figure 4.2.5 on page 19).

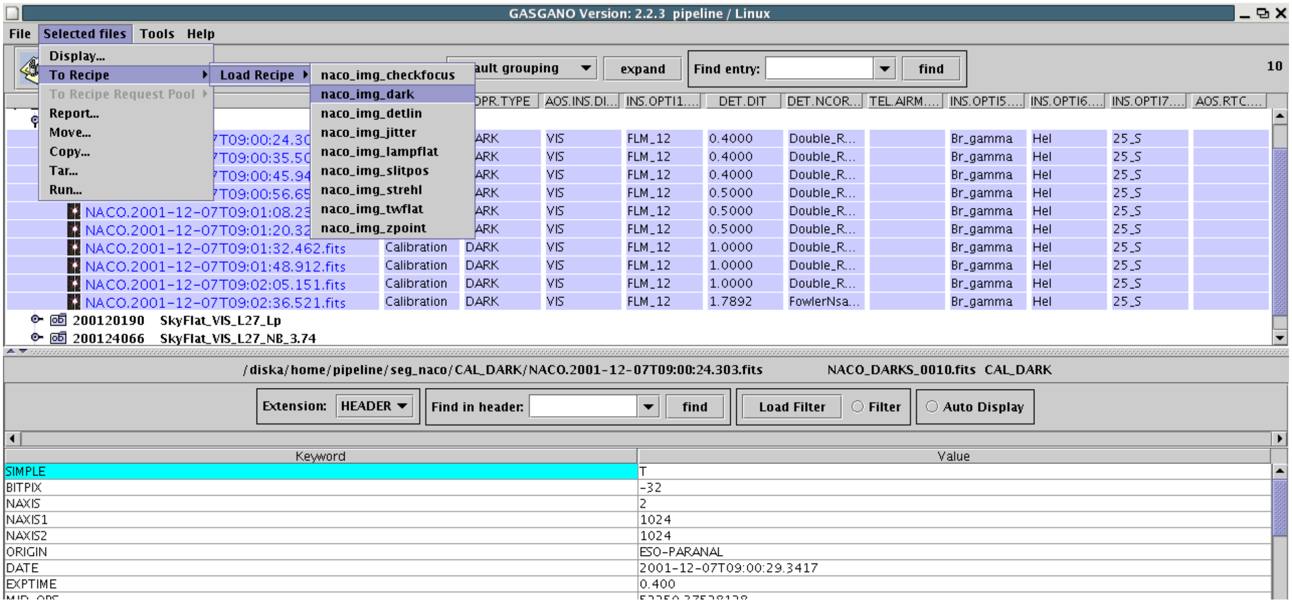

Figure 4.2.4: *The Gasgano main window with selected files and the pull-down-menu with the NACO recipes.*

esorex - -create-config

A list of all available recipes, each with a one-line description, can be obtained using the command:

esorex - -recipes

All recipe parameters (aliases) and their default values can be displayed by the command

esorex - -params recipe_name

To get a brief description of each parameter meaning execute the command:

esorex - -help recipe_name

To get more details about the given recipe give the command at the shell prompt:

esorex - -man-page recipe_name

Recipe configuration: Each pipeline recipe may be assigned an *EsoRex* configuration file, containing the default values of the parameters related to that recipe.[2] The configuration files are normally generated in the directory `$HOME/.esorex`, and have the same name as the recipe to which they are related, with the file name extension `.rc`. For instance, the recipe *naco_img_jitter* has its *EsoRex* generated configuration file named `naco_img_jitter.rc`, and is generated with the command:

esorex - -create-config naco_img_jitter

The definition of one parameter of a recipe may look like this:

```
# --xcorr
# Cross correlation search and measure sizes.
```

[2]The *EsoRex* recipe configuration file corresponds to the *Parameters* panel of the *Gasgano* recipe execution window (see Figure 4.2.5 on page 19).

```
naco.naco_img_jitter.xcorr=40,40,65,65
```

In this example, the parameter `naco.naco_img_jitter.xcorr` is set to the value `40,40,65,65`. In the configuration file generated by *EsoRex*, one or more comment lines are added containing information about the possible values of the parameter, and an alias that could be used as a command line option.

The recipes provided by the NACO pipeline are designed to be usable in a cascade of data reduction steps, each controlled by its own parameters. For this reason and to prevent parameter name clashes we specify as parameter prefix not only the instrument name but also the name of the step they refer to. Shorter parameter aliases are made available for use on the command line.

The command

esorex - -create-config recipe_name

generates a default configuration file **recipe_name.rc** in the directory **$HOME/.esorex**[3].

A recipe configuration file different from the default one can be specified on the command line:

esorex - -recipe-config=my_alternative_recipe_config

Recipe parameters are provided in Section 9 and their role is described in Section 10.

More than one configuration file may be maintained for the same recipe but, in order to be used, a configuration file not located under `$HOME/.esorex`, or having a name different from the recipe name, should be explicitly specified when launching a recipe.

Recipe execution: A recipe can be run by specifying its name to *EsoRex*, together with the name of a set-of-frames. For instance, the following command line would be used to run the recipe *naco_img_jitter* for processing the files specified in the set-of-frames `naco_img_jitter.sof`:

esorex naco_img_jitter naco_img_jitter.sof

The recipe parameters can be modified either by editing directly the used configuration file, or by specifying new parameter values on the command line using the command line options defined for this purpose. Such command line options should be inserted after the recipe name and before the SOF name, and they will supersede the system defaults and/or the configuration file settings. For instance, to set the *naco_img_jitter* recipe *xcorr* parameter to `20,20,65,65`, the following should be typed:

esorex naco_img_jitter - -xcorr="20,20,65,65" naco_img_jitter.sof

For more information on *EsoRex*, see [13].

[3] If a number of recipe parameters are specified on the command line, the given values will be used in the created configuration file.

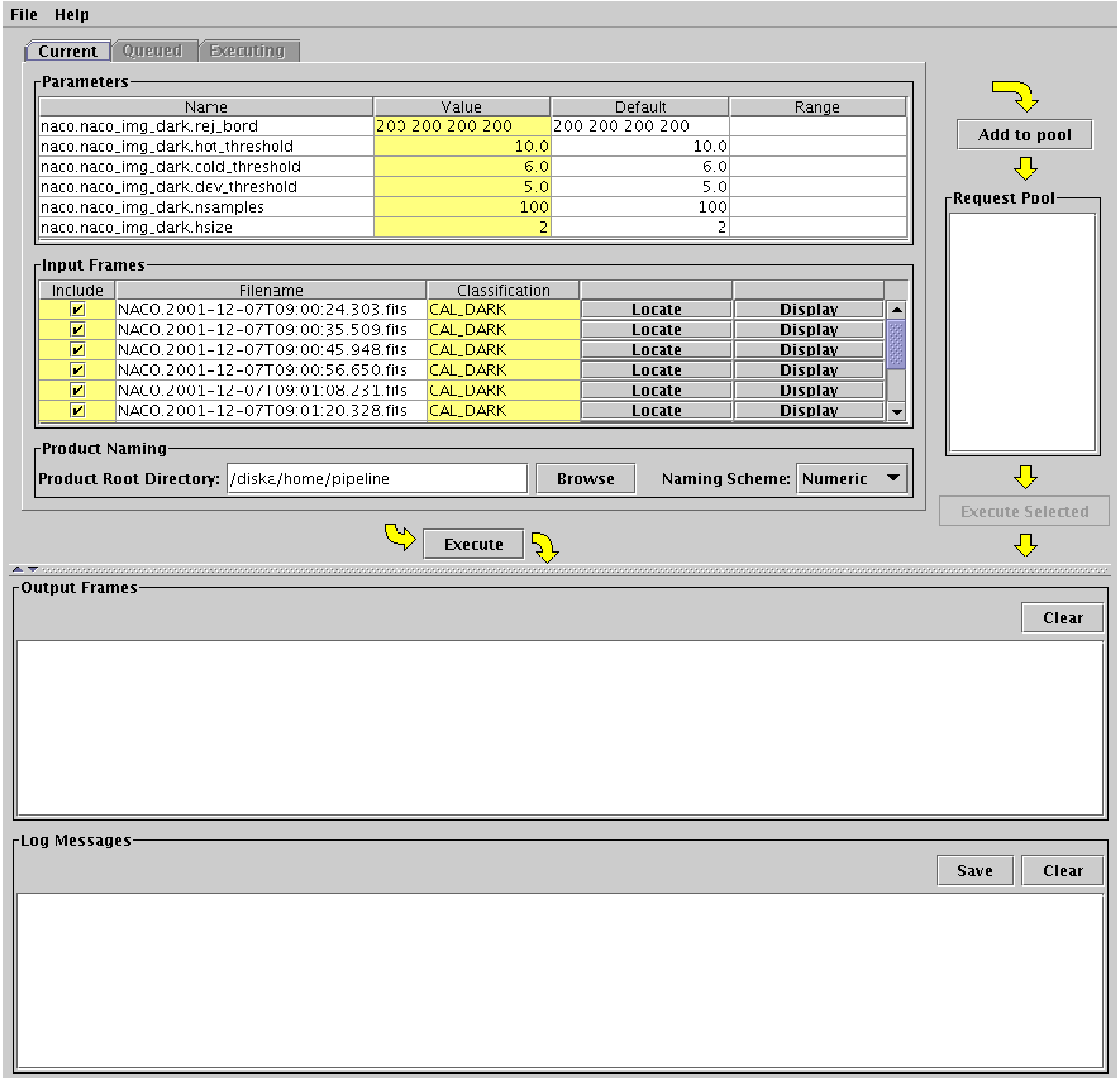

Figure 4.2.5: *The Gasgano recipe window with the recipe naco_img_dark.*

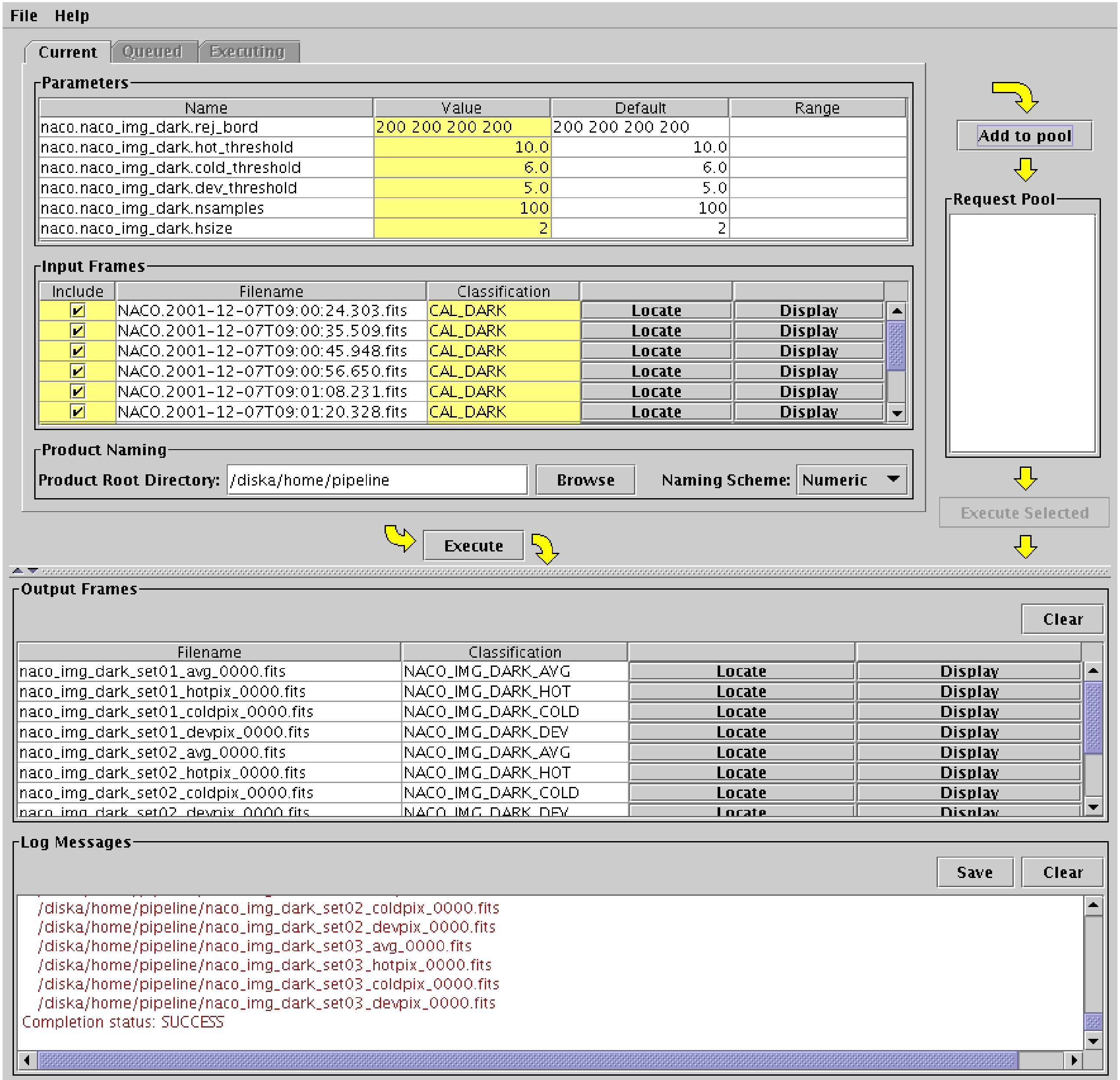

Figure 4.2.6: *The Gasgano recipe window with the recipe naco_img_dark successfully completed.*

5 Known Problems

There are some features on the NACO data that are challenging for the data reduction pipeline, or simply that cannot be corrected by an automatic tool. For some of these, some additional interactive data analysis may be necessary to remove them.

One of them is described below:

5.1 50 Hz noise

The 50 Hz noise effect comes and disappears in a sporadic manner in NACO data, appearing as horizontal stripes in the images.

5.2 Vertical Stripes

Similarly, images may contain vertical stripes every 8 pixels in a quadrant.

6 Instrument Data Description

NACO data uses the FITS format and can be separated into *raw* frames and *product* frames. Raw frames are the unprocessed output of the NACO instrument observations, while product frames are the result of the NACO pipeline processing.

Any raw or product frame can be classified on the basis of a set of keywords read from its header. Data classification is typically carried out by the DO or by *Gasgano* [14], that apply the same set of classification rules. The association of a raw frame with calibration data (e.g. of a science frame with a dark or flat field) can be obtained by matching the values of a different set of header keywords.

Each kind of *raw* frame is typically associated to a single NACO pipeline recipe, i.e. the recipe assigned to the reduction of that specific frame type. In the pipeline environment this recipe would be launched automatically.

In the following all raw NACO data frames, that can be reduced by the NACO pipeline version 4.4.12, are listed, together with the keywords used for their classification and correct association. The indicated *DO category* is a label assigned to any data type after it has been classified, which is then used to identify the frames listed in the *Set of Frames* (see section 4.2.2 on page 15).

6.1 General Data Layout

A raw NACO file always has the images stored in the primary FITS data unit. In nodding mode, it consists of one image, in chopping mode, the two Half-Cycle images are stored as a cube with NAXIS3=2.

6.2 Imaging frames

- **Science observation (Jittered object and sky images):**

  Processed by: `naco_img_jitter`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
`SCIENCE`	`OBJECT`	`IMAGE,JITTER`	`IM_JITTER_OBJ`
`SCIENCE`	`SKY`	`IMAGE,JITTER`	`IM_JITTER_SKY`
`SCIENCE`	`SKY`	`POLARIMETRY,WIRE_GRID,JITTER`	`POL_JITTER_SKY`
`SCIENCE`	`OBJECT`	`POLARIMETRY,WIRE_GRID,JITTER`	`POL_JITTER_OBJ`

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Imaging dark calibration:**

  Processed by: `naco_img_dark`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	DARK	IMAGE	CAL_DARK

See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Imaging Twilight flat field calibration:**

  Processed by: `naco_img_twflat`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	FLAT,SKY	IMAGE	CAL_FLAT_TW

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Imaging Lamp flat field calibration:**

  Processed by: `naco_img_lampflat`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	FLAT,LAMP	IMAGE	CAL_FLAT_LAMP

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Strehl Ratio Computation:**

  Processed by: `naco_img_strehl`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	PSF	IMAGE	CAL_PSF
TECHNICAL	PSF	IMAGE	TECH_PSF

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Zero Point calibration image:**

  Processed by: `naco_img_zpoint`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	STD	IMAGE,JITTER	CAL_STD_JITTER

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Focus check calibration:**

  Processed by: `naco_img_checkfocus`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
TECHNICAL	FOCUS	IMAGE	TECH_FOCUS

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Image for slit position calibration:**

  Processed by: `naco_img_slitpos`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
TECHNICAL	SLIT	IMAGE	SLIT_IMG

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

- **Lamp and dark images for linearity calibration:**

  Processed by: `naco_img_detlin`

  Association keywords: `INSTRUME = NACO`

  Classification:

DPR.CATG	DPR.TYPE	DPR.TECH	DO Category
CALIB	LAMP,LINEARITY	IMAGE	CAL_DETLIN_LAMP
CALIB	OTHER,LINEARITY	IMAGE	CAL_DETLIN_DARK

  See [4] for a definition of the values of DPR.CATG, DPR.TYPE and DPR.TECH.

7 Static Calibration Data

For NACO, the static calibration data are the standard stars catalogs. These data are actually stored internally in the pipeline distribution, and automatically available when the pipeline is installed. These data cannot be updated or corrected without a complete re-installation of the pipeline. In future versions, it is planned to take them out of the distribution and put them in separate calibration files to allow the users to update them.

7.1 Standard Star Catalogs

The standard star catalogs are used by the `naco_img_zpoint` recipe to get the magnitude of the observed standard star to compute the Zero Point.

Stars are currently taken from the following catalogs:

- Arnica (41 entries)
- ESO Van der Bliek (264 entries)
- NACO and ISAAC specific (9238 entries)
- LCO Palomar (64 entries)
- LCO Palomar NICMOS red stars (26 entries)
- MSSSO Photometric (54 entries)
- MSSSO Spectroscopic (343 entries)
- SAAO Carter (67 entries)
- UKIRT extended (54 entries)
- UKIRT fundamental (33 entries)
- UKIRT LM (36 entries)
- UKIRT standards (89 entries)

Each entry in the catalog corresponds to one standard star. They all contain the following information:

- The name of the star
- The position (RA / DEC) of the star
- The spectral type of the star
- The magnitudes in the bands J, H, K, Ks, L, M, Lprime, Mprime

The pipeline uses the position (RA / DEC) to select the proper standard star, rather than the star name which is included in the catalog just as a reference point.

8 Data Reduction

This section provides first an overview of the main issues the data reduction needs to address, with a list of the required recipes plus data. Secondly it indicates the data reduction sequence necessary to reduce calibration and science data.

8.1 Data reduction overview

In SW[4] imaging mode, the used observation technique is jittering. Small shifts in position are applied between successive frames. This way, with a set of a sufficient number of frames, it is possible to make a precise estimation of the sky brightness for all the pixels of the detector, the sky estimation being the most important and difficult part usually in Infra Red imaging.

In LW[5] mode, the level of the background is too high, and it becomes necessary to apply the chopping observation technique to get rid of the sky. The chopping consists of moving the M2 mirror at a high frequency to take two simultaneous images of two parts of the sky. Those two images are referred to as Half-Cycle frames. They are subtracted to remove the sky contribution.

In LW imaging mode, the chopping is used together with the jitter technique.

8.1.1 Dark variations (SW)

The NACO infrared detector dark is a function of the detector integration time (DIT). It also varies in time, most notably at the rows where the reading of the detector starts, that is rows 1, 2, 3, ... and rows 513, 514, 515, ...

These dark variations are non-uniform across the array, but are uniform along most rows. These variations are usually not a serious problem, and do not prevent one from using all the images.

8.1.2 Odd-even column effect (SW)

This effect can be seen as an offset between the odd and even columns of the array. It is a function of the illumination of the array and it evolves with time.

8.1.3 50Hz pickup (SW)

The Aladdin array suffers from 50Hz noise, which appears as stripes that are almost aligned with detector rows. The strength of the pickup depends on how the array is readout and is a function of time.

8.1.4 Detector linearity (SW/LW)

The detector linearity, as measured over a representative region of the array, can be fitted with the polynomial

[4] Short Wavelength uses filters with wavelengths shorter than 2.5 μm.
[5] Using Long Wavelength filters.

$$f_{\mathrm{C}} = a * f_{\mathrm{M}} + b * f_{\mathrm{M}}^2 + c * f_{\mathrm{M}}^3 + d * f_{\mathrm{M}}^4 \tag{1}$$

where f_{M} is the the measured flux and f_{C} is the corrected flux.

8.1.5 Electronic ghosts

For bright sources, one can see electronic ghosts which are 8, 16, 24, etc. rows away from the true source. The amplitude of these ghosts depends on the brightness of the source, their position relative to the central row and how fast the array is read. The faster the array is read, the stronger these ghosts are.

8.2 Required input data

To be able to reduce science data one needs to use raw data, calibration data and pipeline recipes in a given sequence which provides all the input necessary to each pipeline recipe. This sequence is called a data reduction cascade.

Calibration data products can be generated from raw data using the pipeline recipes.

8.3 Reduction Cascade

The imaging reduction cascade is described on figure 8.3.1.The map indicates that a master flat field with a bad pixel map from the twilight flat recipe is needed. The use of flat fields and bad pixel maps are in fact optional, and the master flat field can be provided by the lampflat recipe.

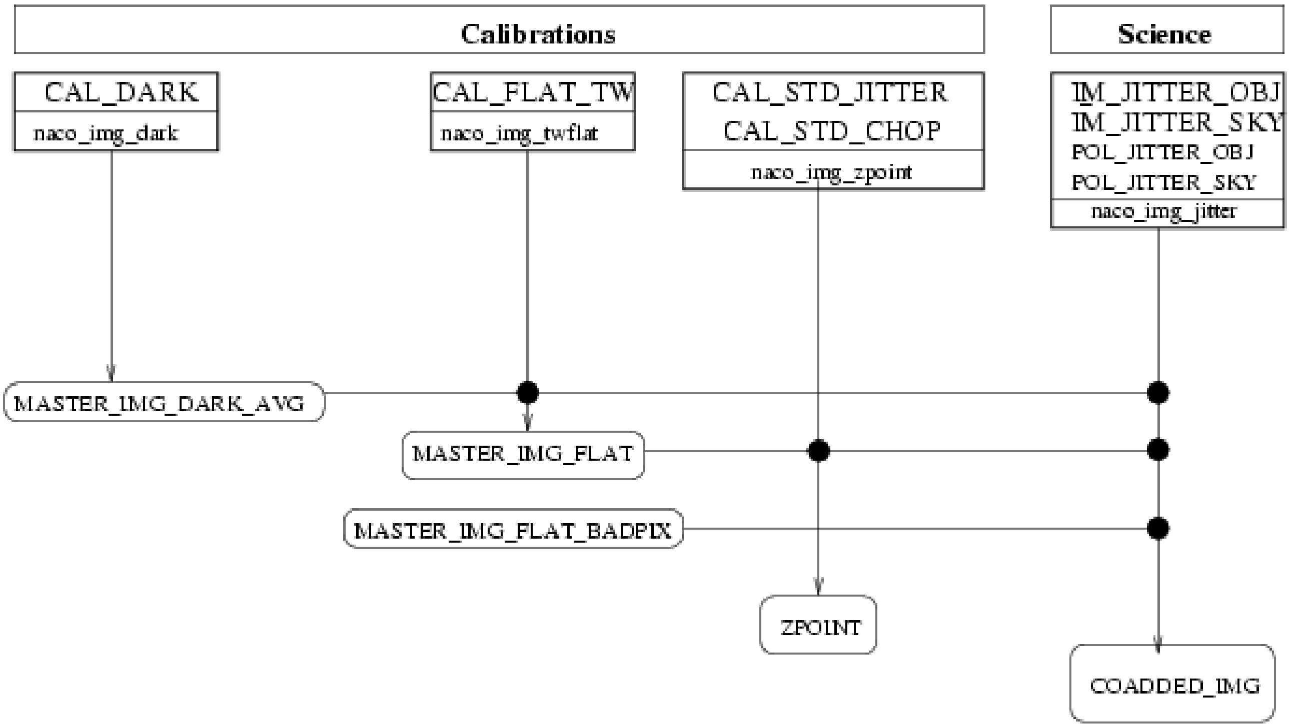

Figure 8.3.1: *NACO Association Map in imaging mode. The rectangles indicate raw data, and specifies the matching recipe, while rounded rectangles indicate calibration.*

9 Pipeline Recipe Interfaces

In this section we provide for each recipe examples of the required input data.

We also provide a list of the pipeline products for each recipe, indicating their default recipe name, the value of the FITS keyword HIERARCH ESO PRO CATG (in short PRO.CATG) and a short description.

For each recipe we also list the input parameters (as they appear in the recipe configuration file), the corresponding aliases for the command line usage, and their default values. Also quality control parameters are listed. Those are stored in the headers of the relevant pipeline products. More information on instrument quality control can be found at [5].

In addition to the products mentioned below, recipes which compute quality control parameters also produce a PAF (VLT PArameter File) which is an intermediate pipeline data file containing those QC values.

9.1 naco_img_jitter

This recipe is for imaging science data. It reduces the data to create a clean combined image.

9.1.1 Input

The recipe expects frames marked with IM_JITTER_OBJ and IM_JITTER_SKY (whether they are objects or sky frames) in non-chopping mode.

The user may optionally provide a bad pixel map (tagged BPM), a flat field (MASTER_IMG_FLAT) and/or dark frames (DARK), see 10.1.1 on page 39.

9.1.2 Output

The produced image is the combined image named naco_img_jitter.fits (PRO CATG = COADDED_IMG or COADDED_CUBE_IMG for cube data).

9.1.3 Quality control

QC BACKGD The median of the background values

QC IQ The Image Quality

QC FWHM PIX The average FWHM in pixels

QC FWHM ARCSEC The average FWHM in arc seconds

9.1.4 Parameters

off The offsets text file (default is NULL)

objs The correlation object(s) position(s) (default is NULL)

xcorr Cross correlation search and measure sizes in x and y (default is 40,40,65,65)

union Create the union of the frames. (default is true)

rej High and Low rejections for the stacking (default is 2,2)

9.2 naco_img_dark

This recipe creates one master dark image, one map of hot pixels, one map of cold pixels, one map of deviant pixels, for one or more instrument settings, and computes the Read-Out Noise (RON) of the detector.

9.2.1 Input

This recipe expects input frames classified as CAL_DARK. These frames are first categorized by the recipe by different settings (exposure time (EXPTIME), read-out mode (NCORRS), detector mode (MODE NAME), OPTI7 NAME, each setting is then reduced separately.

9.2.2 Output

For each instrument setting, numbered i, $i = 01, 02, 03, ...$, the following image products are created:

Master dark named naco_img_dark_set$i$_avg.fits with PRO CATG = NACO_IMG_DARK_AVG.

Map of hot pixels named naco_img_dark_set$i$_hotpix.fits with PRO CATG = NACO_IMG_DARK_HOT.

Map of cold pixels named naco_img_dark_set$i$_coldpix.fits with PRO CATG = NACO_IMG_DARK_COLD.

Map of deviant pixels named naco_img_dark_set$i$_devpix.fits with PRO CATG = NACO_IMG_DARK_DEV.

9.2.3 Quality control

The quality control parameters are computed for each setting. Within each setting, the Read-out Noise is computed on each of the difference images dark$_{j+1}$ - dark$_j$, $j = 1, 2, \cdots, N$setsize $- 1$.

QC DARKMED The mdiean of the central part of the master dark.

QC NBHOTPIX The number of hot pixels.

QC NBCOLPIX The number of cold pixels.

QC NBDEVPIX The number of deviant pixels.

QC RONj Read Out Noise computed from the difference of the images $j + 1$ and j.

9.2.4 Parameters

nsamples Number of samples used to compute the RON (default 100)

hsize Half-size of the boxes used to compute the RON (default 2)

hot_threshold Hot pixel map threshold (default 10.0)

cold_threshold Cold pixel map threshold (default 6.0)

dev_threshold Deviant pixel map threshold (default 5.0)

rej_bord Ignored left right bottom and top border for median and RON computation (default (in pixel) "200 200 200 200")

9.3 naco_img_twflat

This recipe creates one master flat field, optionally one bad pixel map, optionally one intercept image and optionally one error map for one or more instrument settings.

9.3.1 Input

This recipe expects input frames classified as CAL_FLAT_TW, and optionally dark frames classified as CAL_DARK. If, for a given instrument setting, dark frames are provided, there must be either one per flat frame, or just one (which would be used for dark correction on all the frames with that setting).

These frames are first categorized by the recipe by different settings (DIT, read-out mode name (NCORRS NAME), detector mode (MODE NAME), filter (OPT[456] ID)), each setting is then reduced separately.

9.3.2 Output

For each instrument setting, numbered i, $i = 01, 02, 03, ...$, the following image products are created:

Master flat named naco_img_twflat_set$i$.fits with PRO CATG = MASTER_IMG_FLAT.

Map of bad pixels (optional) named naco_img_twflat_set$i$_bpm.fits with PRO CATG = MASTER_IMG_FLAT_BADPIX.

Intercept Map (optional) with the constant value of the fit, is named naco_img_twflat_set$i$_inter.fits with PRO CATG = MASTER_IMG_FLAT_INTERC.

Error Map (optional) with the error bar of the fit, is named naco_img_twflat_set$i$_errmap.fits with PRO CATG = MASTER_IMG_FLAT_ERRMAP.

9.3.3 Quality control

None.

9.3.4 Parameters

rej_bord Ignored left right bottom and top border for median and RON computation (default (in pixel) "200 200 200 200")

thresholds Low and high thresholds for the Bad Pixel Map (default "0.5 2.0")

proport Use the proportional fit (default false)

bpm Create the bad pixel map (default false)

intercept Create the intercept image (default false)

errmap Create the error map (default false)

9.4 naco_img_lampflat

This recipe creates for each of one or more instrument settings a master lamp flat field.

9.4.1 Input

This recipe requires input frames classified as CAL_FLAT_LAMP and with values of the FITS cards with key "HIERACH ESO INS LAMP2 SET" that alternate between zero and non-zero (with non-zero for the first frame). It is further required that the light from the lamp is in a range without a significant thermal background.

These frames are first categorized by the recipe by different settings (DIT, read-out mode name (NCORRS NAME), detector mode (MODE NAME), filter (OPT[564] ID), OPTI3 and OPTI7 NAME), each setting is then reduced separately.

9.4.2 Output

For each instrument setting, numbered i, $i = 01, 02, 03, ...$, the following image products are created:

Master lamp flat named naco_img_lampflat_set$i$.fits with PRO CATG = MASTER_LAMP_FLAT.

9.4.3 Quality control

QC GAIN The conversion factor in e^{-}/ADU

QC FPNOISE The fixed pattern noise

QC LAMPFLUX The median flux of the difference between the first lamp-on and the first lamp-off flats, divided by the DIT. Used to monitor the lamp evolution.

9.4.4 Parameters

rej_bord Ignored left right bottom and top border for median and RON computation (default (in pixel) "200 200 200 200")

9.5 naco_img_strehl

This recipe estimates the strehl ratio and its error bound.

9.5.1 Input

The frames must comprise at least one pair of files and they must be tagged either CAL_PSF (or TECH_PSF).

9.5.2 Output

For each image pair the difference is computed and saved as a product with the name naco_img_strehl$i$.fits with either PRO CATG = NACO_IMG_STREHL_CAL or NACO_IMG_STREHL_TECH depending on the tag of the input frames.

9.5.3 Quality control

For each computed difference image the following QC parameters are produced:

QC AIRMASS Average airmass

QC STREHL Strehl ratio

QC STREHL FLUX Flux of the source used for strehl computation

QC STREHL PEAK Peak value of the source used for strehl computation

QC STREHL ERROR Error bound on strehl ratio

QC STREHL RMS RMS of background in strehl computation

QC STREHL POSX x position of the reference object

QC STREHL POSY y position of the reference object

9.5.4 Parameters

star_r Star radius in Arcseconds (default is 2.0)

bg_r1 Background ring internal radius in Arcseconds (default is 2.0)

bg_r2 Background ring external radius in Arcseconds (default is 3.0)

9.6 naco_img_zpoint

This recipe computes the Zero Point magnitude, i.e. the magnitude of an object that produces one count per second. The Zero Point magnitude usually exceeds 22.

9.6.1 Input

The recipe expects a set of 5 frames with a standard star exposure for each. The first frame must have the standard star around the center, for the other frames, the star appears usually on the 4 quadrants of the detector. The frames are tagged with CAL_STD_JITTER in non-chopping mode, or CAL_STD_CHOP in chopping mode. The recipe optionally uses a flat field (tagged with MASTER_IMG_FLAT).

9.6.2 Output

One or optionally two files are produced. The first is a FITS-table named naco_img_zpoint.fits (PRO CATG = ZPOINT_TABLE) containing for each standard star the photometry computed as exemplified below.

POSX	POSY	ZPOINT	PEAK	FLUX	BGD_NOISE	STREHL	STREHL_ERR	BGD	FWHMX	FWHMY
462.57	474.98	22.118	5372.1	173295	3.4754	0.5352	0.0023	-6.222	2.9473	3.58018
682.24	692.90	22.106	5984.8	171349	3.5342	0.6030	0.0024	4.6597	4.0332	3.01142
682.22	692.86	22.089	5983.6	168731	3.5262	0.6123	0.0024	-1.838	4.0340	3.00890
243.70	694.82	22.127	5530.6	174735	3.5873	0.5465	0.0024	2.5351	3.2022	3.12375
243.75	694.83	22.196	5528.2	186163	3.6515	0.5127	0.0023	-4.548	3.1945	3.12335
242.93	256.99	22.263	5332.6	198035	3.4772	0.4649	0.0020	3.6361	3.4420	4.08671
243.00	257.08	22.211	5329.7	188770	3.6259	0.4875	0.0022	-8.486	4.0959	2.79183
681.71	255.57	22.070	5160.0	165727	3.5003	0.5376	0.0025	5.2507	4.2373	3.12596

The second, optional product is an image named naco_img_zpoint_check.fits with the extracted stars. It can be used to manually verify that stars have been properly used for the computation (PRO CATG = ZPOINT).

9.6.3 Quality control

QC FILTER OBS The name of the filter from the header of the raw files

QC AMBI RHUM AVG The humidity average

QC ZPOINT The computed zero point

QC ZPOINTRMS The error on the zero point

QC STDNAME The name of the standard star

QC STARMAG The magnitude of the standard star

QC CATNAME The catalog name where the star has been found

QC STREHL Strehl ratio

QC STREHL ERROR Error bound on strehl ratio

QC STREHL FLUX Flux of the source used for strehl computation

QC STREHL PEAK Peak value of the source used for strehl computation

QC STREHL RMS RMS of background in strehl computation

9.6.4 Parameters

star_r Star radius in pixels (default is 30)

bg_r1 Background ring internal radius in pixels (default is 40)

bg_r2 Background ring external radius in pixels (default is 60)

ra RA position (default is 999.0 for unknown)

dec DEC position (default is 999.0 for unknown)

mag star magnitude (default is 99.0 for unknown)

sx Search size in pixels in the x direction (default is 10)

sy Search size in pixels in the y direction (default is 10)

check_im Flag to activate the 'check' image creation (default is FALSE)

pscale Can optionally be used instead of the value found in the value of the FITS-card 'ESO INS PIXSCALE' in the header of the first raw file.

9.7 naco_img_checkfocus

This recipe checks the focus.

9.7.1 Input

The recipe expects a set of at least four frames, all tagged TECH_FOCUS. The first frame is used as a dark frame.

9.7.2 Output

No products, only quality control parameters.

9.7.3 Quality control

QC STREHL Strehl ratio

QC STREHL ERROR Error bound on strehl ratio

QC FWHM PIX The full-width at half-maximum of the detected object.

QC ENERGY Star energy

QC FOCUSOPT The optimal focus

9.7.4 Parameters

star_r Star radius in pixels (default is 30)

bg_r1 Background ring internal radius in pixels (default is 40)

bg_r2 Background ring external radius in pixels (default is 60)

9.8 naco_img_slitpos

This recipe is used to get the precise positions of the edges of the slit.

9.8.1 Input

This recipe expects input frames classified as SLIT_IMG.

Each frame is an image of a horizontal slit and is reduced separately.

9.8.2 Output

For each slit image, numbered i, $i = 01, 02, 03, ...$, a table named naco_img_slitpos_i.fits is created, with PRO CATG = SLITPOS_TABLE.

Each table contains 4 columns as exemplified in Figure 9.8.0.

9.8.3 Quality control

QC SLIT POSANG The slit angle in degrees (180 for perfect horizontal alignment)

QC SLIT XPOS The X position in pixels of the slit center

QC SLIT YPOS The Y position in pixels of the slit center

```
#
# file           naco_img_slitpos_01.fits
# extensions     1
# --------------------------------------------
# XTENSION       1
# Number of    rows 1024
# Number of columns 4
#
SLIT_Y   |SLIT_LEFT          |SLIT_CENTER         |SLIT_RIGHT
1        |546.999965984375   |549.999982992187    |553
2        |546.99996611777    |549.999983058885    |553
3        |546.999966251164   |549.999983125582    |553
4        |546.999966384559   |549.999983192279    |553
5        |546.999966517953   |549.999983258977    |553

...

1020     |547.000101913481   |550.00005095674     |553
1021     |547.000102046875   |550.000051023438    |553
1022     |547.00010218027    |550.000051090135    |553
1023     |547.000102313664   |550.000051156832    |553
1024     |547.000102447059   |550.000051223529    |553
```

Figure 9.8.0: *Table created by naco_img_slitpos.*

9.8.4 Parameters

slit_w The slit width in pixels (default 20)

9.9 naco_img_detlin

This recipe computes the detector linearity.

For each pixel, the DIT is fitted as a function of the intensity using a 4th degree polynomial, with no constant term:

$$D_i(x) = ax + bx^2 + cx^3 + dx^4$$

where $D_i(x)$ is the DIT and x is the intensity of pixel i.

9.9.1 Input

This recipe expects three or more pairs of input frames classified as CAL_DETLIN_LAMP and CAL_DETLIN_DARK.

9.9.2 Output

The following image products are created:

Coefficient a named naco_img_detlin_A.fits with PRO CATG = NACO_IMG_DETLIN_A.

Coefficient b named naco_img_detlin_B.fits with PRO CATG = NACO_IMG_DETLIN_B.

Coefficient c named naco_img_detlin_C.fits with PRO CATG = NACO_IMG_DETLIN_C.

Coefficient d named naco_img_detlin_D.fits with PRO CATG = NACO_IMG_DETLIN_D.

Linearity limit named naco_img_detlin_limit.fits with PRO CATG = NACO_IMG_DARK_LIM.

Map of bad pixels named naco_img_detlin_bpm.fits with PRO CATG = NACO_IMG_DARK_BPM.

9.9.3 Quality control

None.

9.9.4 Parameters

force Compute the results even if the lamp is not considered stable (default false)

10 Algorithms

In this section the data reduction procedures applied by the 9 pipeline recipes are described.

10.1 Recipe Algorithms

10.1.1 naco_img_jitter

The basic steps in reducing jittered data are:

1. Removal of the odd-even column effect

   In most cases, the odd-even column effect is small enough that it can be ignored. However, if the flux level of the exposure is above 10,000 or if the effect was strong during the exposure, users should think seriously about removing it. Currently, the recipe does not support odd-even correction.

2. Dark subtraction and flat fielding.

   If provided, the master dark is subtracted from all images and if provided, a flat field is divided into all images.

3. Flagging bad pixels, removing vignetted regions

   The recipe can replace bad pixels by an average of their valid neighbors if a master bad pixel map is provided (using the CPL function cpl_detector_interpolate_rejected() as described in the CPL reference manual, see [10]).

4. Sky subtraction

   This is the most important step and great care and a good understanding of the technique are necessary if good results are required.

   For each input object image, a sky frame is created and subtracted to the object frame. If in the input set of frame, there are some sky frames provided, the sky estimation is simply the median of those sky frames. This median is then used for all the object frames.

   In most cases, there are no sky frames taken, and the sky estimation has to be computed from the object frames. In this case, the median of the object frames is used.

5. Alignment and stacking

   To align the sky-subtracted images, it is necessary to precisely estimate the offsets between them. `naco_img_jitter` applies a 2D cross-correlation routine (using the CPL function cpl_geom_img_offset_combine() as described in the CPL reference manual, see [10]) to determine the offsets, align and stack the images. An initial estimate of the offsets between frames is taken from the FITS headers, unless the initial estimate is provided as an option to the recipe.

Figure 10.1.1 shows an example of a raw image and the result obtained with the *naco_img_jitter* recipe.

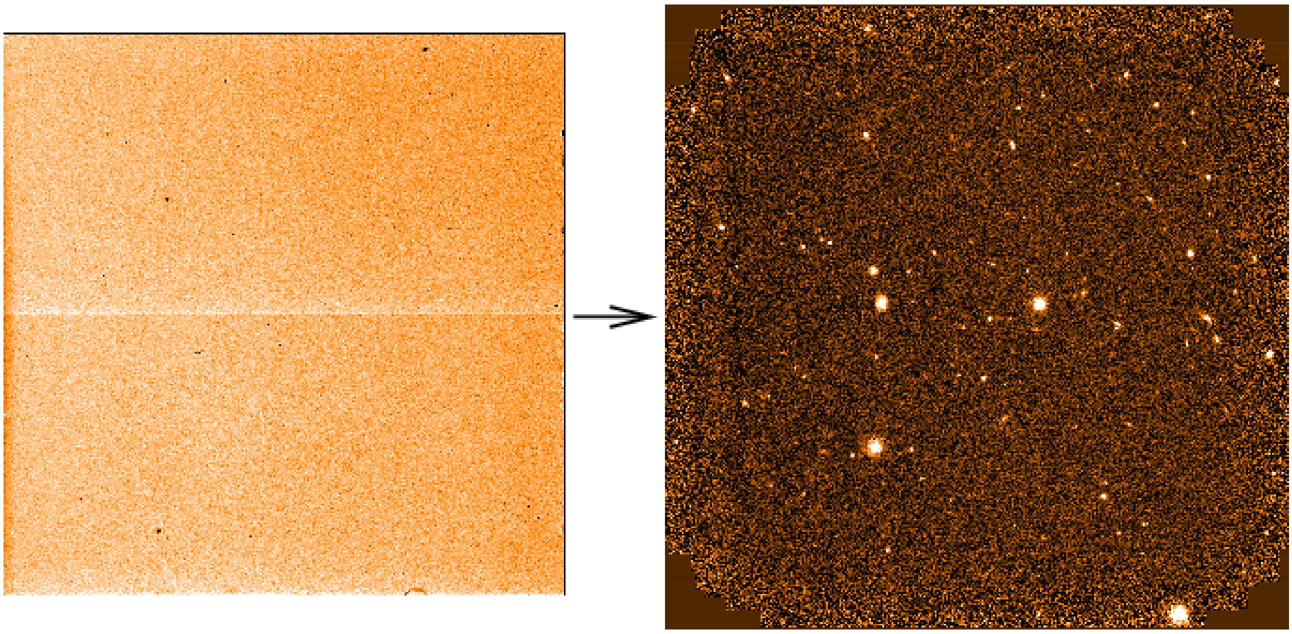

Figure 10.1.1: *Example of jitter result.*

10.1.2 naco_img_dark

Dark frames are exposures without detector illumination.

The darks are reduced with the `naco_img_dark` recipe. This recipe first groups the input list of frames by settings, see 9.2 on page 30.

For each setting, the recipe produces one master dark that is simply the average of the input files. Additionally, the dark level and Read-Out Noise (RON) is measured and written as QC parameter.

From the n_{set} images in each of the settings, $n_{\mathrm{set}} - 1$ difference images are formed. The following measurement is applied to each difference image:

- Generate 100 4×4 windows on the input pixel surface. These windows are optimally scattered using a Poisson distribution to make sure they sample the whole area with as little overlap as possible.
- Compute the pixel standard deviation in each window.
- The readout noise is the median of all these measured standard deviations multiplied by $\sqrt{\frac{NDIT}{2}}$, (computed using the CPL function cpl_flux_get_noise_window() as described in the CPL reference manual, see [10]).

The results are written as QC parameters. For example, an input data set containing 9 frames of 3 different settings will generate 3 master darks (1 per setting) and 2 RON measurements per setting (6 in total).

10.1.3 naco_img_twflat

NACO imaging data may be flat fielded with twilight flats. Such flats are derived by imaging a region of the twilight sky. Between 10 to 25 exposures with constant DIT and NDIT are taken for each filter and a robust linear fit between the flux in each pixel and the median flux of all pixels is used to produce the flat field.

The implemented algorithm is the following:

- The recipe divides the data into groups with identical settings and processes each group separately.
- If provided, the dark frames are subtracted from the flat frames.
- For each pixel on the detector, a curve is plotted of the median plane value against the individual pixel value in this plane. This curve shows the pixel response from which a robust linear regression provides a pixel gain.
- The image showing all pixel gains (i.e. the flat-field) is normalized to have an average value of 1. This image is the master flat produced.

A master flat is produced for each group of frames with the same instrument settings.

Other products are: a map of the zero-intercepts and an error map of the fit. Both can be used to visually check the validity of the fit. A bad pixel map can also be produced by declaring all pixels whose value after normalization is too far from 1 as bad.

10.1.4 naco_img_lampflat

Flats can also be produced by using an internal lamp. For each instrument setting the recipe takes a number of lamp-on/off pairs of exposures.

The implemented algorithm is the following:

- The data are categorised by instrument settings, and each setting is separately reduced. The following steps apply for each setting.
- The lamp-off frames are subtracted from the lamp-on frames.
- The average of these difference-images is computed.
- This average is normalized so the central part of the image has a mean value of 1.
- The gain is computed as

  $$g = \sqrt{\frac{2\mathrm{mean}(L_1)}{\mathrm{RON}^2(L_1 - L_2) - \mathrm{RON}^2(D_1 - D_2)}},$$

  where L_i is the ith lamp frame, D_i is the ith dark frame, and $\mathrm{RON}()$ is the readout noise (computed using the CPL function cpl_flux_get_noise_window() as described in the CPL reference manual, see [10]).

- The fixed pattern noise is computed as

$$n_{\text{fp}} = \sqrt{\text{RON}^2(L_1 - D_1) - \text{RON}^2(L_1 - L_2) - \text{RON}^2(D_1 - D_2)}.$$

- The lamp flux is computed as

$$F_l = \frac{1}{\text{DIT}}\text{median}(L_1 - D_1).$$

The normalized average of dark-subtracted flats is the main product, while the gain, fixed pattern noise and lamp flux are quality control parameters.

10.1.5 naco_img_strehl

The recipe reduces pairs of images of a bright point source, with no overlap of the source between the images in a pair. For each pair the following steps are done:

- The two images are subtracted, thus reducing the noise and producing a difference image with one positive and one negative source. Each difference image is a recipe product.
- The computation of the Strehl ratio is carried out on the difference image in a circle with a radius $R =$ 3 arcseconds (around the center of the positive source. The recipe assumes that the extent of the star is limited by a circle with radius $R_{\text{star}} = 2$ arcseconds.
- The background flux, F_{bg} is estimated as the flux of the pixels located between R_{star} and R whose intensities are in the 90th percentile and not in the 10th percentile.
- The flux of the star, F_{star} is computed as the flux within R_{star} corrected for F_{bg}, and $I_{\text{star,max}}$ is the peak intensity of the star.
- The ideal Point Spread Function is computed as the inverse Fourier Transform of the ideal Optical Transfer Function, which is based on the telescope and instrument characteristics. F_{psf} is the flux of the PSF and $I_{\text{psf,max}}$ is its peak intensity.
- The Strehl ratio is then

$$\frac{I_{\text{star,max}}}{F_{\text{star}}} / \frac{I_{\text{psf,max}}}{F_{\text{psf}}}.$$

The error bound on the Strehl ratio is

$$c \cdot \pi \cdot \sigma_R \cdot w_{\text{pscale}} \cdot R^2_{\text{star}} / F_{\text{star}},$$

where $c = 0.007/0.0271$ is determined empirically, where w_{pscale} is the pixel scale (obtained from the FITS card with key `ESO INS PIXSCALE`), and where σ_R is the estimated noise on the pixels located between R_{star} and R, (using the CPL function cpl_flux_get_noise_ring() as described in the CPL reference manual, see [10]).

10.1.6 naco_img_zpoint

Standard stars are observed every night in the J, H and Ks filters. For the Narrow Band filters, standards are observed as required.

Standard stars are imaged over a grid of five positions, one just above the center of the array and one in each quadrant. The recipe finds the standard (it assumes that the star in the first image is near the center), computes the instrumental magnitude, and then uses the standard star database to determine the Zero Point.

The standard star database contains about 1000 stars with magnitudes in the J, H, K, Ks, L and M bands, although most stars only have magnitudes in a subset of these filters.

The implemented recipe is the following:

1. Compute the difference of the successive images (8 differences for 5 input images).
2. In each difference, locate the star around the expected pixel position.
3. Compute the background around the star, and the star flux.
4. Store the flux result in an output table.

This yields $2(N-1)$ measurements for N input frames. From this statistical set, the highest and lowest values are removed, then an average and standard deviation are computed. The conversion formula from ADUs to magnitudes is:

$$\text{zmag} = \text{mag} + 2.5 \cdot log_{10}\frac{\text{flux}}{\text{DIT}}.$$

where:

- $zmag$ is the computed zero-point.
- mag is the known magnitude of the standard star in the observed band.
- $flux$ is the measured flux in ADUs in the image.
- DIT is the Detector Integration Time.

Note that neither the extinction nor the colour correction are included in the Zero Point. The average airmass is given in the output result file, together with individual airmass values for each frame.

10.1.7 naco_img_checkfocus

The input raw frames must be exposures of a bright point source made with varying focus values ("ESO AOS INS FOCU ABSPOS"). For each frame the strehl ratio and its error estimate is computed (see 10.1.5 on the facing page). For frames with a strehl error estimate less than 0.1, the strehl ratio S as a function of the focus value f is fitted with a second degree polynomial, $S = af^2 + bf + c, a < 0$.

The optimal focus, that maximizes the strehl ratio, is computed from this second degree polynomial as

$$f\mathrm{max} = -\frac{b}{2a}.$$

10.1.8 naco_img_slitpos

As a health check the observatory takes daily images of using different slits. These images are processed with the `naco_img_slitpos` recipe which measures the slit position.

For every slit image, the slit is detected, and precisely analysed to get its precise position. The slit borders positions are stored in a FITS table.

Each image is flipped around the $x = y$-axis before the analysis, thus a perfectly aligned horizontal slit has a slit angle of 180 degrees.

To isolate the slit from other features in the image, a threshold is first applied. Then, successive morphological erosions (with a vertical kernel) are performed until there is one object left. The slit is then reconstructed with dilations (see Figure 10.1.2).

Once the slit is precisely isolated, the edges can easily be detected with a simple thresholding. For each pixel along the slit, the profile is analysed to get the left and the right positions on the slit.

Assuming that the slit is not curved, the left (resp. right) positions are linear-fitted and the fitted positions are written in a table.

10.1.9 naco_img_detlin

To estimate the linearity of the detector, series of images are taken with a lamp where the integration time is increased. Additionally dark frames (lamp off) are taken with the same integration times as for the 'lamp on' frames. To verify the stability of the lamp during the process, control images are taken (typically after every 5 frames) with a reference short integration time.

The recipe algorithm is the following:

- The dark frames are associated (and subtracted) to the 'lamp on' frames by using the DIT.
- The frames with the same DIT as the first frame are used to check the lamp stability over the whole observation: the average of the images should not vary more than 1 percent.
- The rest of the frames are used to compute the linearity: For each pixel the DIT as a function of the detector count, I, is fitted with a 4th degree polynomial

  $$t_{\mathrm{DIT}} = aI + bI^2 + cI^3 + dI^4,$$

  that has a zero-valued constant term. 5 images are produced: One for each polynomial coefficient and one for the error of the fit.

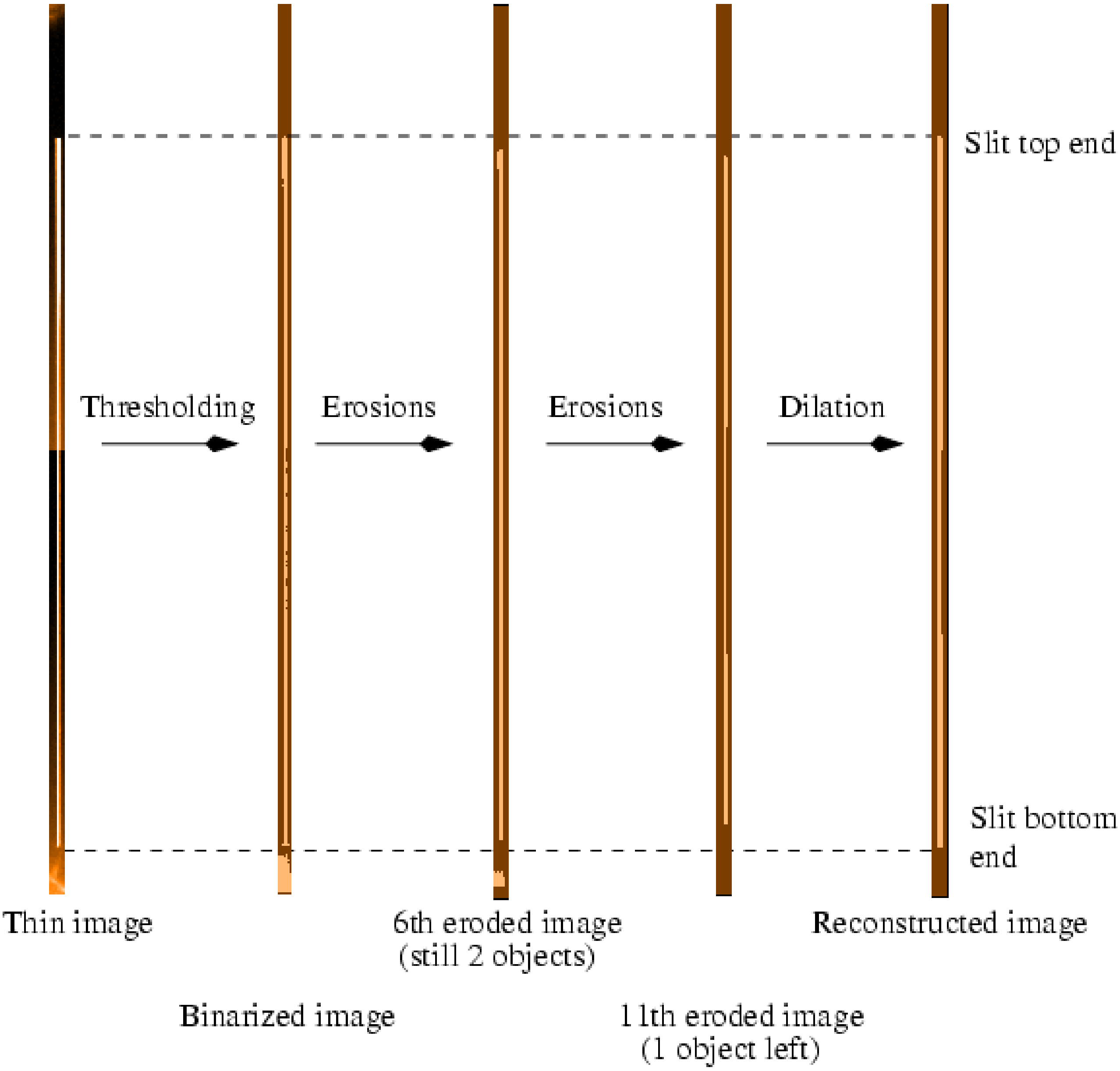

Figure 10.1.2: *Slit detection.*

A Installation

This chapter gives generic instructions on how to obtain, build and install the NACO pipeline version 4.4.12. Even if this chapter is kept as up-to-date as much as possible, it may not be fully applicable to future releases. For instructions on installation of future releases, the reader is therefore advised to check the installation instructions delivered with such future releases. The supported platforms are listed in Section A.1. It is recommended reading through Section A.2.2 before starting the installation.

A bundled version of the NACO pipeline with all the required tools and an installer script is available from *http://www.eso.org/pipelines*, for users who are not familiar with the installation of software packages.

A.1 Supported platforms

The NACO pipeline version 4.4.12 is verified and supported on the VLT target platforms:

- Scientific Linux 4.3 (Athlon), using gcc v. 3.3.4
- Linux Red Hat 9 (P4), using gcc v. 3.2.2
- Sun Solaris 8 (SPARC), using gcc 3.3.

The usage of the GNU build tools should allow to build and run the NACO pipeline on a variety of platforms. As such, the NACO pipeline version 4.4.12 is known to build and to produce correct output with esorex on these platforms:

- Scientific Linux 4.0 (Athlon), using gcc v. 3.3.4
- Linux Ubuntu 6.10 (AMD64), using gcc v. 4.1.2
- Linux XUbuntu 6.10 (P4 Mobile, 128MB RAM), using gcc v. 4.1.2
- Linux Fedora Core 6 (P4), using gcc v. 4.1.1
- Linux SuSE 10.2 (AMD64), using gcc v. 4.1.2
- Linux Mandrake 9.0 (P3), using gcc v. 3.2
- Linux Mandrake 8.0 (P3), using gcc v. 2.96
- Mac OS X 10.4.4 (G4), using gcc 3.3

A.2 Building the NACO pipeline

This section shows how to obtain, build and install the NACO pipeline from the official source distribution.

A.2.1 Requirements

To compile and install the NACO pipeline one needs:

- one of the C compilers listed above,
- a version of the `tar` file-archiving program,
- the GNU software: `gzip`, `perl`, `make`.

On the above mentioned systems a Java Development Kit (JDK) version 1.5.0 will additionally allow the usage of Gasgano (part of this distribution).

A.2.2 Compiling and installing the NACO pipeline

The NACO pipeline distribution kit 4.4.12 contains:

naco-manual-1.0.pdf	The NACO pipeline manual
install_pipeline	Install script
cfitsio2510.tar.gz	CFITSIO 2.510
cpl-4.0.1.tar.gz	CPL 4.0.1
esorex-3.6.5.tar.gz	esorex 3.6.5
gasgano-2.2.7.tar.gz	GASGANO 2.2.7
naco-3.7.0.tar.gz	NACO 3.7.0
naco-calib-3.7.0.tar.gz	NACO calibration files 3.7.0

Here is a description of the installation procedure:

1. Change directory to where you want to retrieve the NACO pipeline recipes 4.4.12 package. It can be any directory of your choice but not:

   ```
   $HOME/gasgano
   $HOME/.esorex
   ```

2. Download from the ESO ftp server, *http://www.eso.org/pipelines*, the latest release of the NACO pipeline distribution.

3. Unpack using the following command:

   ```
   gzip -dc naco-kit-3.7.0.tar.gz | tar xf -
   ```

4. Install: after moving to the top installation directory,

   ```
   cd naco-kit-3.7.0
   ```

it is possible to perform a default installation using the available installer script (*recommended*):

```
./install_pipeline
```

(beware: the execution may take a few minutes on Linux and several minutes on SunOS).

By default the script will install the NACO recipes, *Gasgano*, *EsoRex*, all the necessary libraries, and the static calibration tables, into a directory tree rooted at `$HOME`. A different path may be specified as soon as the script is run.

The only exception to all this is the *Gasgano* tool, that will always be installed under the directory `$HOME/gasgano`. Note that the installer will move an existing `$HOME/gasgano` directory to `$HOME/gasgano.old` before the new *Gasgano* version is installed.

Important: the installation script would ensure that any existing *Gasgano* and *EsoRex* setup would be inherited into the newly installed configuration files (avoiding in this way any conflict with other installed instrument pipelines).

Alternatively, it is possible to perform a manual installation (*experienced users only*): the `README` file located in the top installation directory contains more detailed information about a step-by-step installation.

B QC Parameters

This appendix describes the QC Parameters created by the NACO pipeline:

```
Parameter Name:    QC FWHM PIX
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.2f
Unit:              pix
Comment Field:     (pixels) median FWHM in image
Description:       Median value of the full-width at half-maximum of all
                   detected objects in the final reduced frame.

Parameter Name:    QC FWHM ARCSEC
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.2f
Unit:              arcsec
Comment Field:     (arcsec) median FWHM in image
Description:       Median value of the full-width at half-maximum of all
                   detected objects in the final reduced frame, multiplied
                   by the pixel size in arcsec/pixel.

Parameter Name:    QC IQ
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.2f
Unit:              arcsec
Comment Field:     Estimated image quality (arcsec)
Description:       Estimated value of the image quality, based on
                   the median FWHM of all star-like objects in the
                   final product frame.

Parameter Name:    QC AMBI RHUM AVG
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.f
Unit:
Comment Field:     (percent) ambient relative humidity @ 30/2 m
Description:       Temporal mean of site ambient relative humidity
```

```
                    measured at sensor position i = 1/2 in 30m/2m height.
                    This parameter is delivered by the ASM at intervals
                    of typically 1 minute. This keyword stores the average
                    of all measured values during the zero-point
                    measurement.

Parameter Name:     QC AIRMASS
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:
Comment Field:      Average airmass
Description:        Average airmass during the zero-point acquisition.

Parameter Name:     QC ZPOINT
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:               mag
Comment Field:      Measured zero-point.
Description:        Measured zero-point.

Parameter Name:     QC ZPOINTRMS
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:               mag
Comment Field:      zero-point RMS
Description:        RMS of the zero-point measurement.

Parameter Name:     QC FILTER OBS
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
Unit:
Comment Field:      Observation filter
Description:        This is the filter used to observe the star. It might
                    be different from the one used to get the star
                    magnitude in the database.

Parameter Name:     QC FILTER REF
```

```
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
Unit:
Comment Field:      Reference filter
Description:        If the filter used for observation corresponds to an
                    unknown magnitude value in the database, an association
                    rule is used to see which other magnitude could be
                    used. In that case, this value differs from the
                    actually used filter for observation.

Parameter Name:     QC FILTER NDENS
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
Unit:
Comment Field:      Neutral density filter
Description:        Neutral density filter (INS.OPTI4.NAME)

Parameter Name:     QC FILTER POL
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
Unit:
Comment Field:      Polarizer
Description:        Polarizer (INS.OPTI3.NAME)

Parameter Name:     QC STDNAME
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
Unit:
Comment Field:      Standard star name
Description:        Name of the standard star used for ZP computation

Parameter Name:     QC CATNAME
Class:              header|qc-log
Context:            process
Type:               string
Value Format:       %20s
```

```
Unit:
Comment Field:     Standard star catalogue name
Description:       Name of the catalogue from which the standard star
                   has been extracted.

Parameter Name:    QC SPECTYPE
Class:             header|qc-log
Context:           process
Type:              string
Value Format:      %s
Unit:
Comment Field:     Standard star spectral type
Description:       Standard star spectral type

Parameter Name:    QC STARMAG
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %g
Unit:
Comment Field:     Standard star magnitude
Description:       Standard star magnitude

Parameter Name:    QC STREHL
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              none
Comment Field:     Strehl ratio
Description:       Strehl ratio

Parameter Name:    QC STREHL FLUX
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              ADU
Comment Field:     Source flux
Description:       Flux of the source used for strehl computation

Parameter Name:    QC STREHL PEAK
Class:             header|qc-log
Context:           process
```

```
Type:                double
Value Format:        %f
Unit:                ADU
Comment Field:       Source peak
Description:         Peak value of the source used for strehl computation

Parameter Name:      QC STREHL RMS
Class:               header|qc-log
Context:             process
Type:                double
Value Format:        %f
Unit:                mag
Comment Field:       RMS of background in strehl computation.
Description:         RMS of background in strehl computation.

Parameter Name:      QC STREHL ERROR
Class:               header|qc-log
Context:             process
Type:                double
Value Format:        %f
Unit:                none
Comment Field:       Error bound on strehl ratio
Description:         Error bound on strehl ratio

Parameter Name:      QC STREHL POSX
Class:               header|qc-log
Context:             process
Type:                double
Value Format:        %f
Unit:                pixels
Comment Field:       x position of the reference object
Description:         x position of the reference object

Parameter Name:      QC STREHL POSY
Class:               header|qc-log
Context:             process
Type:                double
Value Format:        %f
Unit:                pixels
Comment Field:       y position of the reference object
Description:         y position of the reference object

Parameter Name:      QC GAIN
Class:               header|qc-log
```

```
Context:            process
Type:               double
Value Format:       %f
Unit:               electron/ADU
Comment Field:      Gain
Description:        Gain is the conversion factor in e-/ADU

Parameter Name:     QC FPNOISE
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:               ADU
Comment Field:      Fixed pattern noise
Description:        Largest noise component in raw images

Parameter Name:     QC LAMPFLUX
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:               ADU/s
Comment Field:      The median flux of the first lampflat difference per DIT
Description:        The median flux of the difference between the first
                    lamp-on and the first lamp-off flats, divided by the
                    DIT. Used to monitor the lamp evolution.

Parameter Name:     QC RONi
Class:              header|qc-log
Context:            process
Type:               double
Value Format:       %f
Unit:               ADU
Comment Field:      Read-out noise of the ith pair of the set
Description:        Measured read-out noise on the whole array.

Parameter Name:     QC NBDEVPIX
Class:              header|qc-log
Context:            process
Type:               integer
Value Format:       %d
Unit:               nb of pixels
Comment Field:      Number of deviant pixels
Description:        Number of deviant pixels
```

```
Parameter Name:    QC NBHOTPIX
Class:             header|qc-log
Context:           process
Type:              integer
Value Format:      %d
Unit:              nb of pixels
Comment Field:     Number of hot pixels
Description:       Number of hot pixels

Parameter Name:    QC NBCOLPIX
Class:             header|qc-log
Context:           process
Type:              integer
Value Format:      %d
Unit:              nb of pixels
Comment Field:     Number of cold pixels
Description:       Number of cold pixels

Parameter Name:    QC DARKMED
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              ADU
Comment Field:     Dark current
Description:       Median of the dark produced image

Parameter Name:    QC ENERGY
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              ADU
Comment Field:     energy
Description:       Star energy

Parameter Name:    QC FOCUS
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              ADU
Comment Field:     focus
Description:       focus
```

```
Parameter Name:    QC FOCUSOPT
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %f
Unit:              ADU
Comment Field:     optimal focus
Description:       optimal focus

Parameter Name:    QC SLIT XPOS
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.3f
Unit:              pix
Comment Field:     Average slit x position
Description:       Provides a measurement of the average position of
                   the slit (in x) on the detector.

Parameter Name:    QC SLIT YPOS
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.3f
Unit:              pix
Comment Field:     Average slit y position
Description:       Provides a measurement of the average position of
                   the slit (in y) on the detector.

Parameter Name:    QC SLIT POSANG
Class:             header|qc-log
Context:           process
Type:              double
Value Format:      %.3f
Unit:              degree
Comment Field:     Slit horizontal angle
Description:       Gives the angle in degrees made by the slit.
```