 <i>HSC</i> <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	1


Scientific Mission Planning System

Pointing Modes

Issue 2.4


1 October 2007

Jon Brumfitt

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	2

Document Change Record

- Issue 0.1 15 July 2005 First draft, sent to ICCs for comment
- Issue 0.2 21 July 2005 Second draft, minor updates
- Issue 0.3 20 December 2005 Third draft, minor updates
- updated section 1.2 and added example of next_state function
 - renamed tinithold/tfinalhold to tih/tfh
 - position-switching & nodding: renamed 'tpp' to 'tnod'
 - repeated-raster-with-hold: deleted open issues section; renamed 'tpp' to 'tnod'
 - repeated-raster-with-hold & line-scan-with-hold: corrected syntax description
 - added state tables for each mode
- Issue 1.0 25 July 2006 Major revision in response to new requirements
- Document title changed in line with other MPS documents
 - Introduction completely rewritten and split into two chapters
 - Added 'References' section
 - Added API descriptions which were previously a separate document
 - Added new modes: “nodding_of_raster” and “custom_map”
 - Similar modes combined into one (see table 1)
 - Modes follow new naming convention to avoid name clashes in scripts
 - Repetition is now supported for several more modes
 - Load, holds and/or OFF positions added to some modes
 - Renamed some parameters:
 - 'relative' to 'fixed'
 - 'ncover' to 'nrepeat'
 - tloadslew[*min*] to 'tload[*min*]
 - 'load_slew' to 'load'
 - Returned arrays no longer duplicate input parameters. The positions of other return values may have changed as a result.
 - Added allowed range for parameter 'top'
 - Removed resolution limit of 1E-5 degrees on raoff/decoff
 - All line-scan modes allow d2=0
 - All modes except fine_pointing now support fixed and relative orientation
 - Increased allowed range of some parameters
 - Type of 'chopthrow' changed to double with resolution of 0.5 arcsec
- Issue 1.1 8 August 2006 Corrections and updates following implementation
- Renamed mode fine_pointing to basic_fine_pointing to allow old modes to coexist.
 - yoffset/zoffset ranges now allow negative values.
 - Figure 1: defines 'tobs', the total observation time, excluding slew
 - All modes now return 'tobs' instead of 'tt'
 - Table 4: state names changed to match state tables
 - Table 12: parameters 'patt' and 'd1' renamed 'pattnod' and 'chopthrow'
 - Corrected values returned by pointing function and next_state functions
 - Corrected timing syntax descriptions

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	3

- Table 13: corrected duration of INIT_HOLD state
- Figure 12: corrected to show pattern starting at OFF position
- Tables 15, 17, 19, 21: corrected state tables
- Table 16: increased range of parameters 'nhold' and 'k'
- Table 18: deleted unused parameter 'nnod'
- line_scan_pointing: hold now occurs after deceleration
- Figure 14: updated to show repetition and new position of hold
- Section 3.8.3: deleted comments
- Table 20: increased range of zoffset/yoffset

Issue 1.2 11 August 2006 Minor corrections

- Renamed mode nodding_in_raster_pointing to nodding_raster_pointing
- Table 1: corrected name of basic_fine_pointing
- Figure 13: corrected spelling in caption
- Table 18: updated note 1 following table
- Custom_map_pointing: corrected table 23 and list of returned values
- Table 24: added parameter pattnod

Issue 1.3 13 September 2006 Corrections


- Section 3.8.4: deleted 'tlh' from list of returned values
- Figure 16: added missing label 'END' to diagram
- Table 21: corrected actions of states INIT_HOLD and OFF(1)
- Table 24: corrected type of parameters 'ra' and 'dec'
- Section 3.7: redesigned nodding_of_raster_pointing following clarification of requirements

Issue 1.4 26 September 2006 Corrections + support for SSO tracking

- Section 1.2: added new preface for issue 1.4
- Section 1, acronym list: Added acronym 'SSO'
- Section 1, applicable documents: updated reference AD1
- Section 2.3: added footnote and explanation of 'tend'
- Figure 1: updated to show 'tend'
- Section 2.13: updated section on SSO tracking
- Section 3.7.2 + table 17: corrected variable name 'k' to 'knod'
- Table 24: corrected type of parameter 'tp'
- Listings 1,2,3: added new parameter '0' for naifid
- Listing 2: removed 'debug_print' statement
- Returned timing arrays now include 'tend'
- Tables 6,8,10,12,14,16,18,20,22,24: added parameter 'naifid'
- Tables 8,10: added footnote on 'k', 'raoff' and 'decoff'
- Tables 14,16,22: changed range/units of parameters 'raoff' and 'decoff'
- Syntax descriptions: added 'tend' and 'tobs'

Issue 1.5 28 September 2006 Corrections

- Parameters 'raoff' and 'decoff' for SSO tracking are now in degrees

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	4

- Tables 14 & 16: corrected range of yoffset and zoffset

Issue 1.6 13 October 2006

- Section 1.4.1: updated issue number of references RD1 and AD1
- Section 2.8: added note that boresight IDs are defined in SIAM ICD
- Tables 10, 20, 22: changed minimum value and resolution of 'd1' and deleted corresponding notes below tables

Issue 1.7 17 October 2006

- Table 24: corrected range of parameter 'nnod' and added note on maximum length of arrays

Issue 1.8 4 December 2007

- Section 3.5.4: added note 1
- Section 3.6.4: added note 1
- Section 3.7.4: added note 1
- Section 3.8.4: added note 1
- Section 3.9.4: added note 1
- Table 24: Increased range of yarray and zarray from [0,7200] to [-7440,7440]
- Section 3.10.4, note 2: increased maximum array length to 3000

Issue 1.9 16 January 2007

- Table 24: increased allowed range of parameters yoffset and zoffset

Issue 1.10 10 April 2007


- Section 1.2: added new preface and removed prefaces to earlier issues
- Section 1.3: updated references to other documents
- Section 2.8: minor rewording to clarify use of parameter 'fixed'
- Section 3.8.4: corrected list of values returned by next_state
- Section 3.11: added new pointing mode cross_scan_pointing

Issue 1.11 16 April 2007

- Table 26: Corrected range of d1 for xrepeat=0

Issue 2.0 25 April 2007

- Section 1.2: added preface to issue 2.0
- Section 2.14: new section defining pointing parameters
- Figure 9: corrected diagram (labels were clipped)
- Figure 11: corrected arrow showing length of 'd2'
- Section 3.4: updated to allow nodding to start at point B:
 - Section 3.4.1: updated description and added table 11.1
 - Figure 9: updated caption to specify parameter 'startAtB'

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	5

- Section 3.4.2: modified INIT_HOLD state
- Section 3.4.2: renumbered table 11 to 11.2
- Section 3.4.3: modified syntax description
- Table 12: added new parameter 'startAtB'
- Split 'API' subsections into sections for 'Parameters' and 'Returned Values'
- Moved descriptions of state machine return values into state table sections
- Removed 'TBC' in sections 3.5.2 and 3.8.2

Issue 2.1 1 October 2007

- Replaced document preface by section 1.2 “Summary of Modes”.
- Table 1: added `no_pointing` mode to table
- Section 1.3.2: added RD6 to list of reference documents
- Section 1.4: added AHF, HSC, HSCCOM and MOC to list of acronyms
- Section 2.4: renumbered tables 3 and 4 as tables 2 and 3
- Section 2.8: added note on definition of position angles near the poles
- Section 2.14: added table number to table
- Section 2.15: new section on HSCCOM annotations
- Section 2.14: corrected entry for 'chopthrow' in the table
- Section 3.12: new mode `no_pointing` for engineering observations
- The calibration hold state was previously called both HOLD and CAL_HOLD. The term 'HOLD' has now been adopted throughout the document.

Issue 2.2 2 October 2007

- Section 2.15.3: corrected example of *nodding_raster*

Issue 2.3 30 January 2009

- Section 2.15: Added new HSCCOM types OBS REQ and BB ID. Changed meaning of OBS ID.

Issue 2.4 9 September 2010

- Added section on new gyro-nodding mode

Anticipated changes:


- It may be necessary to impose a minimum duration of one second (TBC) for 'top' of an OFF position.



Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	6


Contents

1	Introduction.....	9
1.1	Purpose and Scope.....	9
1.2	Summary of Modes.....	9
1.3	References.....	10
1.3.1	Applicable Documents.....	10
1.3.2	Reference Documents.....	10
1.4	Acronyms List.....	10
2	Design Overview.....	12
2.1	Overview.....	12
2.2	Features.....	12
2.3	Slew and Initial/Final Holds.....	13
2.4	Finite State Machine.....	14
2.5	Explicit Timing.....	16
2.6	Syntax Notation.....	16
2.7	Scheduling.....	17
2.8	Coordinates.....	18
2.9	Geometry.....	20
2.10	OFF Position.....	21
2.11	Parameter Types.....	21
2.12	Gyro Propagation.....	22
2.13	Solar System Object Tracking.....	22
2.14	Parameter Definitions.....	22
2.15	HSCCOM Records.....	25
2.15.1	INFO Annotations.....	25
2.15.2	OBS Annotations.....	26
2.15.3	BB Annotations.....	26
2.15.4	PMODE Annotations.....	26
3	Pointing Modes.....	29
3.1	basic_fine_pointing.....	29
3.1.1	Description.....	29
3.1.2	State Table.....	29
3.1.3	Timing Syntax.....	29
3.1.4	Parameters.....	29
3.1.5	Returned Values.....	30
3.2	basic_raster_pointing.....	31
3.2.1	Description.....	31
3.2.2	State Table.....	32
3.2.3	Timing Syntax.....	32
3.2.4	Parameters.....	33
3.2.5	Returned Values.....	33
3.3	basic_line_scan_pointing.....	35
3.3.1	Description.....	35
3.3.2	State Table.....	36
3.3.3	Timing Syntax.....	36
3.3.4	Parameters.....	37
3.3.5	Returned Values.....	38

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	7

3.4	nodding_pointing.....	39
3.4.1	Description.....	39
3.4.2	State Table.....	40
3.4.3	Timing Syntax.....	41
3.4.4	Parameters.....	41
3.4.5	Returned Values.....	42
3.5	gyro_nodding_pointing.....	43
3.5.1	Description.....	43
3.5.2	State Table.....	44
3.5.3	Timing Syntax.....	45
3.5.4	Parameters.....	45
3.5.5	Returned Values.....	46
3.6	raster_pointing.....	47
3.6.1	Description.....	47
3.6.2	State Table.....	47
3.6.3	Timing Syntax.....	48
3.6.4	Parameters.....	48
3.6.5	Returned Values.....	49
3.7	nodding_raster_pointing.....	50
3.7.1	Description.....	50
3.7.2	State Table.....	51
3.7.3	Timing Syntax.....	52
3.7.4	Parameters.....	52
3.7.5	Returned Values.....	53
3.8	nodding_of_raster_pointing.....	54
3.8.1	Description.....	54
3.8.2	State Table.....	56
3.8.3	Timing Syntax.....	56
3.8.4	Parameters.....	57
3.8.5	Returned Values.....	57
3.9	line_scan_pointing.....	59
3.9.1	Description.....	59
3.9.2	State Table.....	60
3.9.3	Timing Syntax.....	60
3.9.4	Parameters.....	60
3.9.5	Returned Values.....	61
3.10	line_scan_with_off_pointing.....	62
3.10.1	Description.....	62
3.10.2	State Table.....	63
3.10.3	Timing Syntax.....	63
3.10.4	Parameters.....	64
3.10.5	Returned Values.....	65
3.11	custom_map_pointing.....	66
3.11.1	Description.....	66
3.11.2	State Table.....	67
3.11.3	Timing Syntax.....	67
3.11.4	Parameters.....	67
3.11.5	Returned Values.....	68
3.12	cross_scan_pointing.....	69

3.12.1	Description.....	69
3.12.2	State Table.....	71
3.12.3	Timing Syntax.....	71
3.12.4	Parameters.....	72
3.12.5	Returned Values.....	73
3.13	no_pointing.....	74
3.13.1	Description.....	74
3.13.2	State Table.....	74
3.13.3	Timing Syntax.....	74
3.13.4	Parameters.....	74
3.13.5	Returned Values.....	74

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	9

1 Introduction

1.1 Purpose and Scope

Herschel was originally designed to support three simple pointing modes: fine-pointing, raster map and line scan, as defined in the System Requirements Specification [RD5]. Each of these modes is implemented by a single spacecraft telecommand [AD2].

It was subsequently found that a more complex set of modes were needed to obtain the best scientific results from the instruments. Consequently, the three primitive modes are now used as building blocks from which to construct a set of more complex *composite pointing modes*.


This document describes each of the composite modes in detail, including its behaviour, timing and Application Programming Interface (API). It is intended for use by the instrument teams, so that they can develop observing modes for use with their instruments.

The modes described here have been developed in response to the requirements given in the Pointing Command Requirements document [AD1]. The modes specified in the requirements document may not correspond exactly with the modes described here, since similar modes are sometimes combined into more generic modes.

1.2 Summary of Modes

The modes in the requirements document [AD1] map onto modes in this document as follows:

<i>Requirement</i>	<i>Implementation</i>
Fine pointing	basic_fine_pointing
Raster	basic_raster_pointing
Raster with an off position	basic_raster_pointing
Line scan	basic_line_scan_pointing
Line scan with an off position	basic_line_scan_pointing
Position switching	nodding_pointing
Nodding	nodding_pointing
Composite position switching	nodding_pointing
Composite nodding	nodding_pointing
Repeated raster with hold	raster_pointing
Nodding in raster	nodding_raster_pointing
Nodding in raster with off	nodding_raster_pointing
Nodding of raster	nodding_of_raster_pointing
Line scan with hold	line_scan_pointing
Repeated line scan with off	line_scan_with_off_pointing

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	10

<i>Requirement</i>	<i>Implementation</i>
Chaining of pointings	custom_map_pointing
Orthogonal line scan	cross-scan_pointing
-	no_pointing

Table 1: Relationship between requirements and implementation

The suffix “_pointing” should be avoided in the variable names used in CUS scripts to avoid name clashes when new modes are introduced.

The modes `basic_raster_pointing` and `basic_line_scan_pointing` are primitive spacecraft pointing modes invoked by a single telecommand. It is not envisaged that these will be used, but they have been included for completeness.

1.3 References

1.3.1 Applicable Documents


- AD1 T.Prusti, M.Sanchez, “Pointing Command Requirements for Herschel”, Herschel-HSC-MEM-0616, issue 2.0.
- AD2 “ACMS Telecommand Definition”, H-P-4-DS-TN-024, Dutch Space, issue 4 revision 0, 15 March 2006.

1.3.2 Reference Documents


- RD1 “ICD: Herschel Spacecraft / Instrument Alignment Interface”, DMS# PT-HMOC-FD-ICD-2111-OPS-GFT, HGS# HGS-ICD-026, issue 1.3, 7 February 2006.
- RD2 J.Brumfitt, “Herschel Scientific Mission Planning System – Design Concepts”, HSC-DOC-741, issue 0.2, 6 March 2007.
- RD3 J.Brumfitt, C.Porrett, “Common Uplink System – User Guide”, Herschel-HSC-DOC-0424, issue 1.20, 22 June 2006.
- RD4 J.Brumfitt, C.Porrett, “Common Uplink System – Language Guide”, Herschel-HSC-DOC-0425, issue 1.41, 21 June 2006.
- RD5 Herschel / Planck System Requirements Specification (SRS)”, Annex 1, SCI-PT-RS-05991, issue 3/3, 27 July 2004.
- RD6 J.Brumfitt, “Herschel Planned Observation Sequence (POS) Interface Control Document”, Herschel-HSC-ICD-0377, HGS-ICD-024, issue 1.6, 11 April 2007.

1.4 Acronyms List

- ACA Attitude Control Axes
- ACMS Attitude Control and Measurement System

 <i>HSC</i> <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	11

AHF	Attitude History File
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CUS	Common Uplink System
DEC	Declination
FSM	Finite State Machine
HCSS	Herschel Common Science System
HSC	Herschel Science Centre
HSCCOM	HSC Comment (used in POS file)
MOC	Mission Operations Centre
MPS	Mission Planning System
N/A	Not Applicable
NAIF	Navigation and Ancillary Information Facility
POS	Planned Observation Sequence
RA	Right Ascension
SIAM	Spacecraft/Instrument Alignment Matrix
SSO	Solar System Object
TBC	To Be Confirmed
TBD	To be Decided

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	12

2 Design Overview

2.1 Overview

The Herschel spacecraft supports three primitive pointing modes: fine-pointing, raster map and line scan. These are used as building blocks from which to construct a set of more complex *composite pointing modes* that meet the needs of the user.

Each observation uses an instance of an *observing mode* that is defined in the CUS language [RD4]. The observing mode invokes a single composite pointing mode, such as a raster, and then issues a sequence of instrument commands to control the instruments. This requires that the script controls the timing of the instrument commands precisely, to correspond with the on-target times of each point in the pattern.

The pointing command returns an array of timing parameters that allow the observing mode to synchronise with the spacecraft manoeuvring. Since the timing calculations are often quite complex (and hence error prone), each pointing mode implements a finite-state-machine that takes care of the timing; the CUS script simply has to define what instrument operations it wishes to perform at each state such as a raster point or OFF position.

2.2 Features

All composite modes support the following features:

- Instrument boresight coordinates are used instead of the spacecraft ACA coordinates.
- ZY offsets may be specified in boresight coordinates.
- Rasters and line-scans are specified using the coordinates of the centre, rather than the first point. However, nodding is specified using the first point.
- Commanding is possible during the initial slew, to prepare the instrument.
- An *initial hold* period may be specified, at the end of the slew, before starting to execute the pointing pattern. This allows instrument preparation that requires a stable pointing.
- A *final hold* may be specified at the end of the observation, after the pointing pattern has completed, to allow any final instrument commanding.

In addition, certain modes support other special features that are not available in the primitive spacecraft pointing modes. These include:

- The orientation of the pattern may be specified relative to the sky as well as relative to the boresight axes.
- It is possible to perform a nodding pattern (2-point raster) that spends unequal times at the two pointing positions.



Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	13

- The number of nods or scan lines may exceed the limit of 32 imposed by the primitive commands.
- It is possible to periodically pause the pattern to perform instrument calibrations. If the spacecraft is required to have a fixed attitude during this time, it is known as a *hold*, otherwise it is known as a *load*¹.
- It is possible to repeat the whole pattern a number of times.
- It is possible to perform nodding at each point in a raster pattern.

2.3 Slew and Initial/Final Holds

An observation is considered to start at the beginning of the slew from the previous observation. Hence, the slew period is available to the observation to prepare the instrument before it reaches the target. As the slew time depends on the sequence of observations within a particular schedule, it may not be long enough to prepare the instrument. Consequently, all the pointing modes have a parameter 'tslewmin' that allows a minimum slew time to be specified.

The pointing mode returns a value 'tslew', which is the greater of the requested 'tslewmin' and the actual slew time. If the actual slew time is less than the requested time, the spacecraft simply waits at the 'hold' point as shown in figure 1(ii) below.

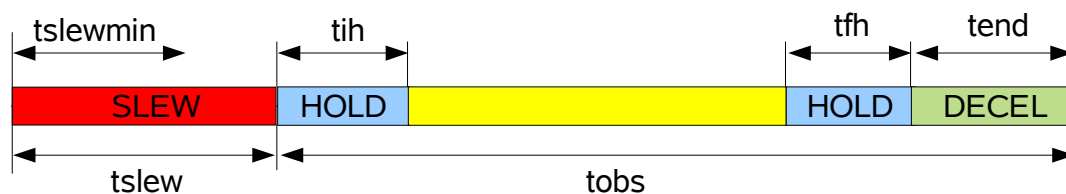


Figure 1(i): $tslewmin < tslew$




Figure 1(ii): $tslewmin > tslew$

Additional time may be requested at the hold point, using the parameter 'tih' (initial hold time) in order to perform instrument calibrations that require a fixed pointing². Similarly, the parameter 'tfh' (final hold time) can be used to request an additional hold at the end of the pattern to complete any necessary instrument commanding.

In the case of a raster, the initial hold point is simply the first raster point (or an OFF position). In the case of a line-scan, it is a point displaced from the start of the first scan

¹ The term *load* is used, rather than *load-slew*, since it does not actually require a slew. For example, if a load occurs at the end of a raster, the spacecraft simply waits at the last raster point.

² In the case of a tracking observation, 'fixed' means fixed with respect to the tracking frame.

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	14

line to allow time for the spacecraft to accelerate to scanning speed before it reaches the start of the line. See the descriptions of the individual modes for further details.

The observation ends with a final period 'tend', which is used by tracking observations to decelerate the spacecraft to rest with respect to the inertial frame. This value is returned by the pointing mode with the other timing information.

2.4 Finite State Machine

Each pointing command returns an array of timing parameters that can be used to synchronise the timing of instrument commands with the manoeuvring of the spacecraft. Although explicit control of timing is possible using these parameters, it becomes extremely complex and hence error prone in all but the simplest cases.

In order to simplify the development of observing modes, each pointing mode implements a finite state machine (FSM) that takes care of the timing. Each call to the `next_state` function [RD4] advances the time and moves to the next state as appropriate. The CUS script simply needs to define the instrument operations to be performed in each state. The explicit timing parameters are still available, if required, to handle special situations within the overall state machine approach.

When a state is entered, the state machine checks that the time is no later than the expected time. Hence, errors in the timing of a CUS script will be detected if they cause the instrument commands to overrun into the next state. A further advantage of using the state machine is that changes to the pointing modes that involve the introduction of a new state, such as a hold, will not affect existing observing modes that do not need to make use of the new state³.

Using the `next_state` function, an observing mode for a raster pointing can be implemented as follows:

```
obs RasterExample {
  double ra = 0.0 in [0.0,360.0];
  double dec = 0.0 in [-90.0,90.0];
  double patt = 0.0 in [0.0,360.0];
  int m = 5 in [2,32];
  int n = 5 in [1,32];
  double d1 = 10.0 in [2.0,480.0];
  double d2 = 20.0 in [2.0,480.0];
  int tp = 100;
}
{
  int tslewmin = duration(H_Prepare());
  int tih = duration(H_Setup());
  int tfh = duration(H_Complete());

  int[] ts = basic_raster_pointing(true,tslew,tih,tfh,"H01_0",0,ra,
                                  dec,true,patt,0.0,0.0,m,n,d1,d2,tp,0,0,0,0);
}
{
  int[] state = [0];
  while(state[0] >= 0) {
    state = next_state();
    if(state[0] == 1) {
      H_Prepare();
    }
  }
}
```

³ However, additional parameters may need to be passed in the call to the pointing mode.

```

    if(state[0] == 2) {
        H_Setup();
    }
    if(state[0] == 3) {
        H_Measure(tp);
    }
    if(state[0] == 5) {
        H_Complete();
    }
}
}

```

Listing 1: Timing using a State Machine

This example also shows how the CUS **duration** command can be used to find the duration of the commands performed during the various phases of the observation, so that they can be used as inputs to the pointing mode.

Each of the mode descriptions includes a *state table* for the finite state machine, as shown by the following simple example for a basic raster:

State	Condition	Duration	Next state	Action
START			SLEW	p=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	POINT	p++
POINT	mi ≠ m	tp + tpp	POINT	p++
	ni ≠ n	tp + tll	POINT	p++
	else	tp	FINAL_HOLD	
FINAL_HOLD		tfh	END	

Table 2: Example of state table

The state machine advances to the next state each time that the `next_state` function is called. Each state may have a number of entries with different conditions, as illustrated by POINT in the example. These conditions are evaluated from top-to-bottom, until one matches. Then the corresponding action is performed, the CUS time is advanced by the amount shown in the 'duration' column and the new state is entered. For further details of usage in the CUS language see the CUS documentation [RD3, RD4].

Each state may be split into a number of *sub-states*, denoted by a number in parentheses after the state name. The set of sub-states is normally one logical state as far as the user is concerned. However, it is often possible to distinguish between sub-states by making use of the array returned by the `next_state` function.

This document refers to the states by name, whereas the CUS API uses integer values which are assigned as follows:

No.	Name
-1	END
0	START
1	SLEW

No.	Name
2	INIT_HOLD
3	POINT
4	OFF
5	FINAL_HOLD
6	HOLD
7	NOD
8	LINE
9	LOAD

Table 3: State names

2.5 Explicit Timing

It is *strongly recommended* that the state machine approach is used in all observing modes, to reduce the complexity. It would become extremely difficult and error prone to code the timing of one of the more complex pointing modes using explicit timing.

The following example illustrates explicit timing for the trivial case of a fine pointing:


```
obs SlewAndHoldTimingExample {
  double ra = 20.0 in [0.0,360.0];
  double dec = 30.0 in [-90.0,90.0];
  int n = 100;
}
{
  int tp = duration(H_Measure(n));
  int tslewmin = duration(H_Prepare());
  int tih = duration(H_Setup());
  int tfh = duration(H_Complete());
  int[] t = basic_fine_pointing(true,tslewmin,tih,tfh,"H01_0",0,ra,dec,
                                0.0,0.0,tp);
}
{
  int tslew = t[1]; // Slew time
  H_Prepare(); // Prepare instrument during slew
  wait_until(tslew);
  H_Setup(); // Calibrate at fixed pointing
  wait_until(tslew + tih);
  H_Measure(tp); // Perform the measurement
  wait_until(tslew + tih + tp);
  H_Complete(); // Perform any final instrument operations
  wait_until(tslew + tih + tp + tfh);
}
```

Listing 2: Explicit timing – not recommended

The CUS `wait_until` command is used to wait until the spacecraft arrives at the appropriate point, before issuing the relevant instrument commands. The pointing mode returns an array of times 't', of which element `t[1]` is always the slew duration 'tslew'.

2.6 Syntax Notation

The state tables define the timing of each mode precisely. However, these tables are often quite complex and difficult to understand. In order to provide a simpler high-level

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	17

summary, a simple syntactic description is also provided.

For example, the raster from the example above may be described as follows:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend              {observation}
      tt   = tx (tll tx)*                  {raster pattern}
      tx   = tp (tpp tp)+                  {one raster line}

```

The following notation is used:

```

t1 t2          t1 followed by t2
t1 | t2        t1 or t2
[t]           optional occurrence of 't'
t*            zero or more repetitions of 't'
t+            one or more repetitions of 't'
(...)         grouping of items
{...}        comment

```

Concatenation has a higher precedence than disjunction. Hence:

```
a b | c d   ≡   (a b) | (c d)
```

2.7 Scheduling

The Herschel scientific Mission Planning System (MPS) performs scheduling in two phases. Initially, it assigns a sequence of observations of known duration to specific start times, allowing for slew times and various scheduling constraints. Finally, this draft schedule is expanded into a sequence of instrument and spacecraft telecommands using the CUS to schedule the telecommands within the time slots allocated for the observations.


This approach allows the scheduler to work efficiently, since it can generate a draft schedule without considering timing at the telecommand level. For this to work, it is important that the duration of an observation is fixed and does not depend (by more than a small margin) on the time at which the observation is scheduled. To avoid problems of this kind, the CUS has been designed to minimise dependence of the duration on the spacecraft orientation.

The following trivial example shows how an observing mode is split into a parameter section, a pointing section and an instrument section:

```

obs RasterExample {
  // Parameter section
  double ra = 0.0 in [0.0,360.0];
  ...
}
{
  // Pointing section
  int[] t = basic_fine_pointing(true,0,0,0,"H01_0",0,ra,dec,
                                0.0,0.0,tp);
}
{
  // Instrument section
  wait_until(t[1]);
  H_Measure(tp);      // Perform the measurement
}

```

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	18

Listing 3: Spacecraft and instrument timing in an observing mode

This shows that the input parameters to a pointing mode may depend on the input parameters of the observing mode, but cannot depend on the times in the array 't' returned by the pointing mode. This prevents circular definitions of timing and ensures proper timing as far as the mission planning system is concerned. For example, the duration of the observation (excluding the slew) cannot depend on the slew time from the previous observation.

2.8 Coordinates

All pointings are described in *boresight coordinates* for a specified boresight. Each instrument has a set of defined boresights, which may be slightly offset and rotated relative to the spacecraft Attitude Control Axes (ACA). The mapping between these coordinate systems is defined by a Spacecraft Instrument Alignment Matrix (SIAM). Each boresight is referenced by an instrument boresight identifier, as defined in the SIAM ICD [RD1].

Each pointing command specifies a *reference point*, with respect to which the pattern is defined. This point is normally defined in equatorial J2000 coordinates by a right ascension (RA) and declination (DEC). In the case of a tracking observation, the target body is specified by an integer NAIF identifier.

The roll angle of the spacecraft is tightly constrained by thermal constraints, which require that the roll angle with respect to the Sun is less than one degree. The observer specifies the reference point (e.g. RA and DEC) of the observation and the Mission Planning System calculates an appropriate roll angle for the time at which the observation is scheduled.

All the pointing modes have YOffset and ZOffset parameters, that allow a small offset of the pattern to be specified in instrument coordinates. This cannot be achieved by simply offsetting the RA and DEC, because the attitude of the spacecraft is unknown when the observation is requested.

Some of the modes allow this offset to be up to 5 degrees, so that a sequence of observations can map a larger area of sky. In other modes the offset is restricted to 3 arc-minutes and is used to make small adjustments to align the boresight with a particular part of the detector.

The orientation of the pattern may be specified in one of two ways, as shown in figures 2 and 3. Although illustrated for a raster, the diagram is also applicable to a line-scan pattern.

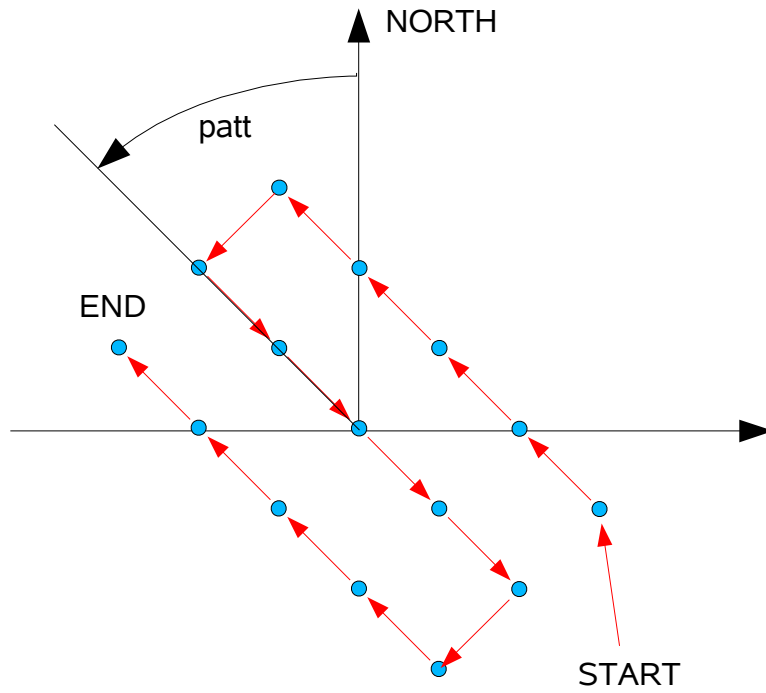


Figure 2: Raster orientation looking at sky, when fixed=true

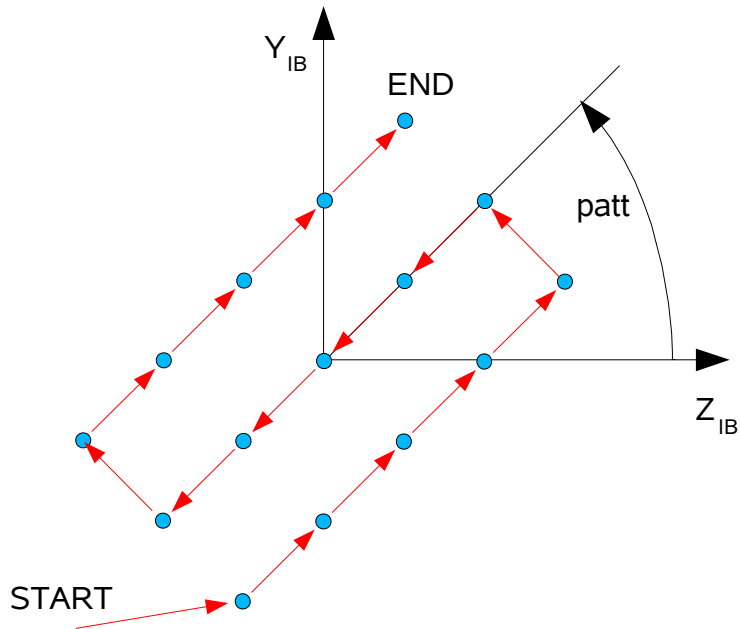



Figure 3: Raster orientation looking at sky, when fixed=false

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	20

The parameter 'fixed' is used to specify the orientation as follows:

- When fixed=true, the pattern is *fixed* with respect to the sky. Consequently, it is independent of the time at which the observation is scheduled. In this case, the orientation is specified as a *position angle* with respect to North.
- When fixed=false, the orientation is specified *relative*⁴ to the instrument boresight axes as an anticlockwise rotation about the X axis. Consequently its orientation on the sky depends on when the observation is scheduled.

The position angle is defined as the rotation between the ACA X-Z plane and the plane defined by the ACA X-axis and celestial North (declination = 90 degrees). The position angle increases in an anticlockwise direction about the positive X-axis.

Irrespective of whether the orientation is fixed or relative, the pattern is always performed with respect to a fixed inertial frame (except for Solar System targets where it is performed with respect to a *tracking frame*). In the case of a relative observation the orientation is fixed at the time when the observation is scheduled.

Note: Whenever a position angle is specified, it refers to the orientation at the reference point. This may differ from the position angle at the raster centre after applying proper motion offsets, ZY offsets, etc. This is especially an issue near the equatorial poles, where a small translation of a pattern may result in large change in position angle. A good way to think about this is to consider the reference position (e.g. RA and DEC) and position angle as a triple that define an attitude on the sky. This is simply translated to the pattern centre by applying the relevant offsets.

The image plane seen looking at the sky has two axes, Z and Y, which are the projections of the instrument boresight Z and Y axis onto the sky. (Z, Y, -X) form a right-handed coordinate system with the -X axis coming out of the plane towards the observer. The pattern rotation in the image plane is measured anticlockwise about the +X axis. This is consistent with the definition of position angle, which increases anticlockwise on the sky.


Further details can be found in the Herschel Scientific Mission Planning System Design Concepts document [RD2].

2.9 Geometry

The primitive rasters and line-scans modes are implemented on-board the spacecraft and are commanded by a single telecommand. The raster points are arranged on a grid whose rows and columns follow great circles. Similarly, scan lines and their end points also follow great circles. As a result, scan lines converge slightly towards the ends and the outermost scan lines are slightly shorter than the line at the centre. The pattern is symmetrical about its centre point.

Most of the composite modes, except for the basic raster and line-scan, are constructed from either fine-pointings or one-line scans. This is necessary in order to accommodate the loads, holds and other features that are not supported by the primitive modes. Rasters

⁴ The parameter name 'relative' was causing confusion because the value was 'false' when the orientation was *relative* to the spacecraft. This parameter has been renamed to 'fixed', but it behaves in exactly the same way as before.

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	21

built this way use the same great-circle geometry as the primitive modes. Line-scans are built from one-line scans and hence all the scan lines are exactly the same length.

The geometry of a pattern such as a raster is symmetrical about its centre point. When ZY offsets are applied, the whole pattern is simply shifted, such that it remains symmetrical about the shifted centre point rather than about the reference point (e.g. RA and DEC). If large offsets (e.g. up to 5 degrees) are used to construct a large map from a number of smaller rasters, note that each sub-raster is symmetrical about its own centre point rather than the centre of the overall pattern. Consequently, the geometry of the individual rasters starts to differ noticeably from that of a single large raster; some overlap should then be allowed to ensure complete coverage.

Further details can be found in the Herschel Scientific Mission Planning System Design Concepts document [RD2].

2.10 OFF Position

Some pointing modes have an OFF position that is used to perform a background measurement. The boresight visits the OFF position after a predefined number of raster points or scan-lines, depending on the stability time of the instrument.

The primitive raster and line-scan commands that are implemented on-board require that the OFF position is not more than 2 degrees from the centre, measured in either the d1 or d2 directions. This imposes a similar constraint on the composite pointing modes that use these primitive modes.

Note that the maximum displacement of the OFF position is measured relative to the *offset* raster centre, rather than the reference point. For pointings that are specified with an orientation relative to the spacecraft, the direction on the sky of the ZY offset depends on when observation is scheduled. Hence an OFF position that is within the 2 degree range one day may be outside it another day. To avoid this problem the maximum zoffset/yoffset values are smaller when an OFF position is used.


Strictly, the maximum displacement is in ACA coordinates whereas the pointing mode arguments are in instrument boresight coordinates. However, the same SIAM correction is applied to both the raster centre and OFF position, so the angle between them remains unchanged.

2.11 Parameter Types

Each pointing mode description includes details of the parameters needed to invoke the pointing from a CUS observing mode.

The following abbreviations are used for parameter types:

b	bool
i	int
d	double
s	string

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	22

Where a parameter is of type 'double', the value will generally be rounded to the nearest integer multiple of the resolution specified for the parameter. To avoid unexpected results, it is recommended that CUS scripts work with integer multiples of the resolution.

The pointing modes return an integer array which contains timing parameters such as the point-to-point slew times. This is often not required since the state machine handles the timing. However, it can be useful in more complex situations.

2.12 Gyro Propagation

The pointing modes do not yet support gyro propagation mode.

2.13 Solar System Object Tracking

The pointing modes support the tracking of moving Solar System Objects (SSO). In this case, the target is specified by a NAIF identifier (naifid) instead of a fixed right ascension (ra) and declination (dec).

As far as the API is concerned, the modes have three parameters, 'naifid', 'ra' and 'dec'. When naifid=0, the 'ra' and 'dec' are used. A non-zero 'naifid' indicates that tracking is required and the 'ra' and 'dec' parameters are not used but should be set to zero.

Tracking has the following effect on timing of the observation:


- The slew time 'tslew' is increased to allow for acceleration to the tracking velocity.
- A deceleration period 'tend' is required at the end of the observation to allow the spacecraft to decelerate to rest with respect to the inertial frame. This occurs after the final hold. This value is included in the timing parameters returned by the pointing mode.

For a normal (non-tracking) observation, the parameters 'raoff' and 'decoff' specify the right ascension and declination of the OFF position. For a tracking observation, this could result in excessively long slews to/from the OFF position, depending on the time of observation. To avoid this problem, the OFF position of a tracking observation is specified as an offset relative to the pointing pattern. The same parameters 'raoff' and 'decoff' are used, but they have a different meaning for tracking observations.


The OFF position is specified as an offset with respect to the pattern axes. As with a normal fixed observation, the orientation of the pattern axis may be specified either with respect to the sky (fixed=true) or the spacecraft (fixed=false). The origin of the pattern axes is offset by 'zoffset' and 'yoffset' (in instrument coordinates) with respect to the reference point. In the case of a tracking observation, the reference point is the moving path of the body specified by the 'naifid'.

2.14 Parameter Definitions

The following table defines the parameters used in the pointing descriptions:

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	23

Name	Description
execute	If false, the CUS pointing command calculates and returns the timing array without commanding the pointing.
tslewmin	The minimum duration of the slew from the previous observation. This is used to ensure enough time to prepare the instrument during the slew. The return value 'tslew' is the greater of 'tslewmin' and the actual slew time.
tih	Duration of the hold at the end of the slew from the previous observation, before starting the pointing sequence (e.g. raster). This is used when the instrument preparation requires a stable pointing.
tfh	Duration of the final hold at the end of the observation used to perform instrument reconfiguration
ib	The instrument boresight identifier (e.g. "H13_0") as defined in the SIAM ICD.
naifid	The NAIF identifier of a Solar System target. This is set to '0' when tracking is not required.
ra	The right ascension of the target. Unused when a 'naifid' is specified.
dec	The declination of the target. Unused when a 'naifid' is specified.
yoffset	Offset of pattern in the direction of the +Y axis of the boresight coordinate frame. The offset is with respect to the reference point specified by 'ra' and 'dec' or 'naifid'.
zoffset	Offset of pattern in the direction of the +Z axis of the boresight coordinate frame. The offset is with respect to the reference point specified by 'ra' and 'dec' or 'naifid'.
fixed	When true, the pattern orientation is specified as a position angle with respect to North. When false, the pattern orientation is specified with respect to the boresight coordinate frame.
patt	Rotation angle of the pattern. The parameter 'fixed' specifies how this angle is defined.
tp	Duration of a fine pointing.
m	Number of points in a raster line.
n	Number of raster lines or scan lines.
d1	Spacing of points along a raster line or the length of a scan line.
d2	Spacing of adjacent raster lines or scan lines.
k	Number of points or scan lines before performing a measurement at the OFF position.
top	Time spent at the OFF position.
raoff	When naifid=0, this is the right ascension of the OFF position. For tracking observations (where naifid ≠ 0), the offset of the OFF position with respect to the pattern axes.

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	24

Name	Description
decoff	When naifid=0, this is the declination of the OFF position. For tracking observations (where naifid \neq 0), the offset of the OFF position with respect to the pattern axes.
rate	Scan rate of a line scan.
nmod	Number of nods (AB or BA switches) in a nodding or position switching sequence.
chopthrow	The nodding distance in the direction specified by 'pattnod'.
pattnod	The direction of the nod position with respect to the un-nodded position. The exact definition depends of the pointing mode.
tpa	Duration of pointing at each nod 'A' position.
tpb	Duration of pointing at each nod 'B' position.
tloadmin	The minimum duration of a load operation.
nload	The number of slews (e.g. nods) before performing a load operation. The definition of slew depends on the pointing mode.
thold	The duration of the hold when nhold>0.
nhold	Number of nods or raster points or scan lines before performing a hold.
nrepeat	Number of repetitions of the pattern. The pattern is performed once when nrepeat=1.
trepeatmin	The minimum duration of the return slew between repetitions of a pattern. The return value 'trep' is the greater of 'trepeatmin' and the actual slew time.
ncycles	See nodding_of_raster_pointing for details.
yarray	See custom_map_pointing for details.
zarray	See custom_map_pointing for details.
xrepeat	Number of repetitions of the cross scan.
pattx	Orientation of cross scan (similar to 'patt' for the nominal scan). See cross_scan_pointing for details.
nx	Number of scan lines in the cross-scan. See cross_scan_pointing for details.
d1x	Length of cross-scan lines.
d2x	Distance between cross-scan lines.
nholdx	Number of cross-scan lines before performing a hold.
startb	If true, the nodding pattern starts at B instead of A.

Table 4.1: Parameter definitions

2.15 HSCCOM Records

The Herschel Scientific Mission Planning System generates a Planned Observation Sequence (POS), which contains the ACMS pointing commands and associated instrument commands for each observation, as described in the POS ICD [RD6].

The pointings in the POS file are annotated using records that are copied into the Attitude History File (AHF) to assist the HSC in processing the attitude history data. As far as the formal POS interface between HSC and MOC is concerned, these records are simply comments. Hence, they are known as HSCCOM (HSC Comment) records. However, as far as the HSC are concerned, these comments have a definite syntax and meaning in relation to the pointing modes, as described below.

Each HSCCOM record in the POS has the following format:

```
<time> HSCCOM <type> <body>
```

For example:

```
2008-10-18T16:12:08Z HSCCOM PMODE POINT 1,2 NOD A,3
```

The type field may be one of the following:

<i>Type</i>	<i>Description</i>
INFO	Information on ACMS pointing commands
OBS	Observation annotation
BB	Building block annotation
PMODE	Pointing mode annotation
DEBUG	Temporary information for testing/debugging purposes

Table 4.2 HSCCOM record types

2.15.1 INFO Annotations


The ACMS pointing commands in the POS file use spacecraft attitudes expressed in the form of quaternions in Attitude Control Axis (ACA) coordinates. For visual inspection purposes it is convenient to annotate these commands with attitudes expressed as right ascension (RA), declination (DEC) and position angle (PA) in degrees, using instrument boresight coordinates. This is done using HSCCOM INFO records.

For example:

```
2008-10-18T20:33:29Z HSCCOM INFO PREQ IB: 319.25135 6.83647 262.33381
```

The allowable INFO record types are as follows:

```
INFO PREQ      IB: <ra> <dec> <pa>
INFO RREQ      IB: <ra> <dec> <pa>
```

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	26

```
INFO RREQ-OFF  IB: <ra> <dec> <pa>
INFO LREQ      IB: <ra> <dec> <pa>
INFO LREQ-OFF  IB: <ra> <dec> <pa>
```

The prefix “IB:” indicates that these are Instrument Boresight coordinates. The suffix “-OFF” indicates that it refers to the OFF position.

Important note: These INFO records must NOT be used in processing the attitude history file as they describe the low-level ACMS commands used to implement a pointing mode. Instead, the AHF processing should use the high-level PMODE annotation that describe the *logical* pointing pattern. The implementation of a pointing mode in terms of low-level ACMS commands may change; for example an N-line scan may be commanded using a single ACMS line-scan with N lines or as N one-line ACMS line-scans.

2.15.2 OBS Annotations

The OBS annotation provides the following information on the observation:

```
OBS NAME <observation name>
OBS MODE <observing mode>
OBS REQ  <observation request ID>
OBS ID   <observation ID>
```

For example:

```
2008-05-18T16:31:19Z HSCCOM OBS NAME PointFastDBS-Band2bWarm
2008-05-18T16:31:19Z HSCCOM OBS MODE HifiPointModeFastDBS
2008-05-18T16:31:19Z HSCCOM OBS REQ  84
2008-05-18T16:31:19Z HSCCOM OBS ID   3221226311
```

2.15.3 BB Annotations

The BB annotation provides the following information on a building block:

```
BB ID <building block ID>
```

For example:

```
2008-05-18T16:35:21Z HSCCOM BB ID 393281537
```

2.15.4 PMODE Annotations

The PMODE (Pointing MODE) annotations provide information on the *logical* pointing pattern generated by a pointing mode.

<i>Name</i>	<i>State</i>	<i>Description</i>
SLEW	SLEW	Initial slew
POINT	POINT	Fine pointing
POINT n	POINT	Fine pointing n (custom_map)

<i>Name</i>	<i>State</i>	<i>Description</i>
POINT n,m	POINT	Raster line n, point m
NOD A,k	POINT	Nodding position A, cycle k
NOD B,k	NOD	Nodding position B, cycle k
POINT n,m NOD A,k	POINT	Nodding (of) raster position A
POINT n,m NOD B,k	NOD	Nodding (of) raster position B
LINE n	LINE	Scan line N
OFF	OFF	OFF position
LOAD	LOAD	Load
HOLD	HOLD	Calibration hold
FINAL_HOLD	FINAL_HOLD	Final hold
CYCLE n		Nodding_of_raster cycle 0,1,...
REPEAT n		Repetition number 0,1,...
CROSS n		Cross-scan 0,1,2
TRACKING <bool>		Tracking enabled

Table 4.3 HSCCOM PMODE annotations

In most cases, the PMODE annotation gives the state name, with arguments indicating the logical raster point number etc. The REPEAT, CROSS and CYCLE annotations provide information on repetitions and do not correspond to additional pointing mode states. Note that the INITIAL_HOLD state is not included.

In the state machine the POINT state may correspond to a fine pointing, a raster point or point A of a nodding cycle, whereas the NOD state correspond to point B of the nodding cycle. In the PMODE annotations, a different convention is used, in which POINT refers to a fine/raster point and NOD A/B refers to the nod position. This allows a more concise description of nodding_raster and nodding_of_raster pointings.

For example, a nodding_raster with nnod=2 proceeds as follows:

```

PMODE POINT 1,1 NOD A,1
PMODE POINT 1,1 NOD B,1
PMODE POINT 1,1 NOD B,2
PMODE POINT 1,1 NOD A,2
PMODE POINT 1,2 NOD A,1
PMODE POINT 1,2 NOD B,1
PMODE POINT 1,2 NOD B,2
PMODE POINT 1,2 NOD A,2
...


```

whereas a nodding_of_raster with k=2 proceeds as follows:

```

PMODE POINT 1,1 NOD A,1
PMODE POINT 1,2 NOD A,1

```

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	28

```

PMODE POINT 1,1 NOD B,1
PMODE POINT 1,2 NOD B,1
PMODE POINT 1,3 NOD B,2
PMODE POINT 1,4 NOD B,2
PMODE POINT 1,3 NOD A,2
PMODE POINT 1,4 NOD A,2
. . .

```

Raster points are described by two numbers n and m. Alternate raster lines are normally traversed in the reverse direction to reduce the slew time between raster lines. Hence, a 3x3 raster pattern is as follows:

```

PMODE POINT 1,1
PMODE POINT 1,2
PMODE POINT 1,3
PMODE POINT 2,3
PMODE POINT 2,2
PMODE POINT 2,1
PMODE POINT 3,1
PMODE POINT 3,2
PMODE POINT 3,3

```

3 Pointing Modes

3.1 basic_fine_pointing

3.1.1 Description

A `basic_fine_pointing` is a single fixed pointing at a given RA and DEC. The spacecraft maintains a fixed inertial attitude for a duration 'tp', as shown in figure 4.

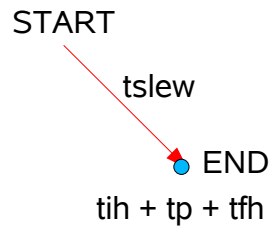


Figure 4: Basic fine pointing

3.1.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	POINT	
POINT		tp	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 5: State table for basic_fine_pointing

The `next_state` function returns the following array of integer values:

{state, time}

3.1.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend              {observation}
      tt   = tp                            {raster point}

```

The parameters 'tih' and 'tfh' allow additional hold times to be introduced before and after the pointing, respectively. These parameters are not strictly necessary as the same effect could be achieved by extending the time 'tp'. However, they are included for consistency with the other modes.

3.1.4 Parameters

The CUS command `basic_fine_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
yoffset	d	[-180,180]		arcsec	
zoffset	d	[-180,180]		arcsec	
tp	i	[1,50000]	1	s	

Table 6: Parameters for basic_fine_pointing

3.1.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tend}

3.2 basic_raster_pointing

3.2.1 Description

A basic_raster_pointing is a regular grid of fine pointings, implemented using a primitive raster telecommand, as shown by the example in figure 5.

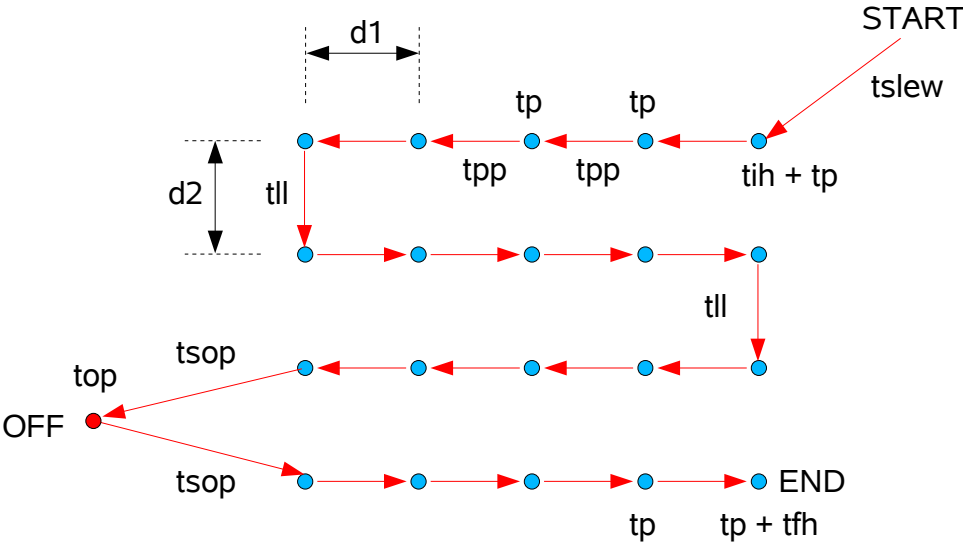


Figure 5: Basic_raster_pointing with $m=5$, $n=4$, $k=15$

After every 'k' raster points, the boresight slews to a predefined OFF position, after which it continues at the next raster point. If $k=0$, no OFF positions occur. If 'm x n' is exactly divisible by 'k', the pattern finishes at the OFF position, as shown in figure 6. In this case, the final hold occurs at the OFF position.

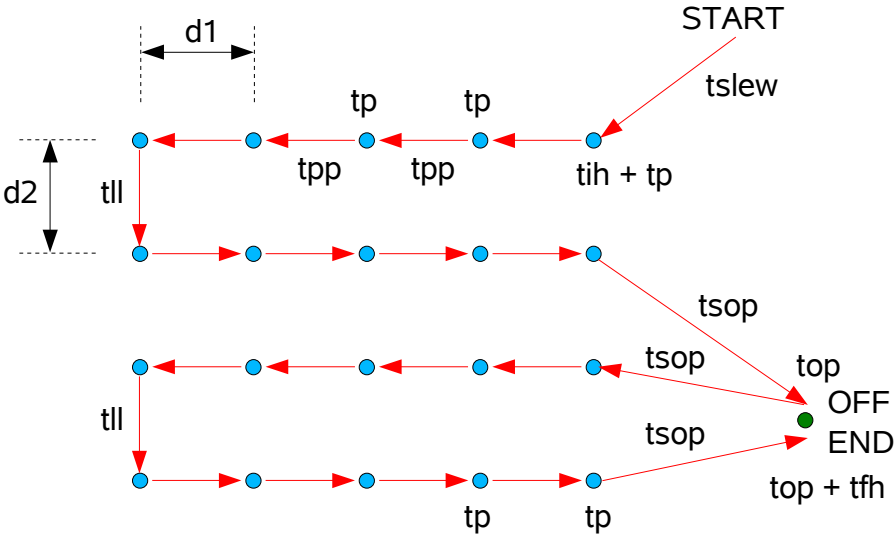


Figure 6: Basic_raster_pointing ending on OFF, $m=5$, $n=4$, $k=10$

The RA and DEC coordinates, with any ZY offset applied, specify the position of the raster centre. The raster orientation is specified by an angle 'patt', as follows:

- If the parameter 'fixed' is true, 'patt' is a position angle on the sky. When patt=0, the direction of the first raster line is North. See figure 2 for an example.
- If the parameter 'fixed' is false, 'patt' is measured anticlockwise about the boresight X axis. When patt=0, the first raster line is in the +Z direction and the raster lines proceed in the +Y direction. See figure 3 for an example.

3.2.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	p=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	POINT	p++
POINT	k>0 & p%k=0	tp + tsop	OFF	
	mi ≠ m	tp + tpp	POINT	p++
	ni ≠ n	tp + tll	POINT	p++
	else	tp	FINAL_HOLD	
OFF	ni=n & mi=m	top	FINAL_HOLD	
	else	top + tsop	POINT	p++
FINAL_HOLD		tfh + tend	END	

Table 7: State table for basic_raster_pointing

The next_state function returns the following array of integer values:

{state, time, p}

3.2.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + obs}
where tobs = tih tt tfh tend                {obs}
      tt   = tx ((tll | (tsop top tsop)) tx)* [tsop top] {raster}
      tx   = tp ((tpp | (tsop top tsop)) tp)+         {line}

```

Notes:

- The slews to and from the OFF position do not all take equal time. The value 'tsop' is the worst-case value corresponding to the longest slew; the spacecraft simply waits at the next stable pointing position if the actual slew time is less than this.
- If fixed=true, the slew times to/from the OFF position do not vary with the time at which the observation is scheduled. If fixed=false, the worst-case slew time may be assumed, so that the duration of the observation is independent of when it is scheduled.

3.2.4 Parameters


The CUS command `basic_raster_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	(1)
zoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	(1)
m	i	[2,32]			
n	i	[1,32]			
d1	d	[2,480]	0.5	arcsec	
d2	d	0 or [2,480]	0.5	arcsec	
tp	i	[10,50000]	1	s	
k	i	0 or [2,m*n], naifid=0 0, naifid>0			(2)
top	i	[0,50000]	1	s	
raoff	d	[0,360)		deg	(2)
decoff	d	[-90,90]		deg	(2)

Table 8: Parameters for basic_raster_pointing

Notes:

1. The offsets `yoffset` and `zoffset` have maximum absolute values of 3 arc-minutes (TBC) when an OFF position is used. The 2 degree limit on the ZY coordinates of the OFF position relative to the nominal raster centre is reduced by 3 arc-minutes to ensure that the OFF position is always within in allowed range relative to the actual (offset) raster centre, irrespective of the spacecraft roll angle.
2. This mode does not support OFF positions with SSO tracking.

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	34

3.2.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tpp, tll, tsop, tend}



3.3 basic_line_scan_pointing

3.3.1 Description

A basic_line_scan_pointing is a regular array of scan lines, implemented using a primitive line-scan telecommand, as shown by the example in figure 7.

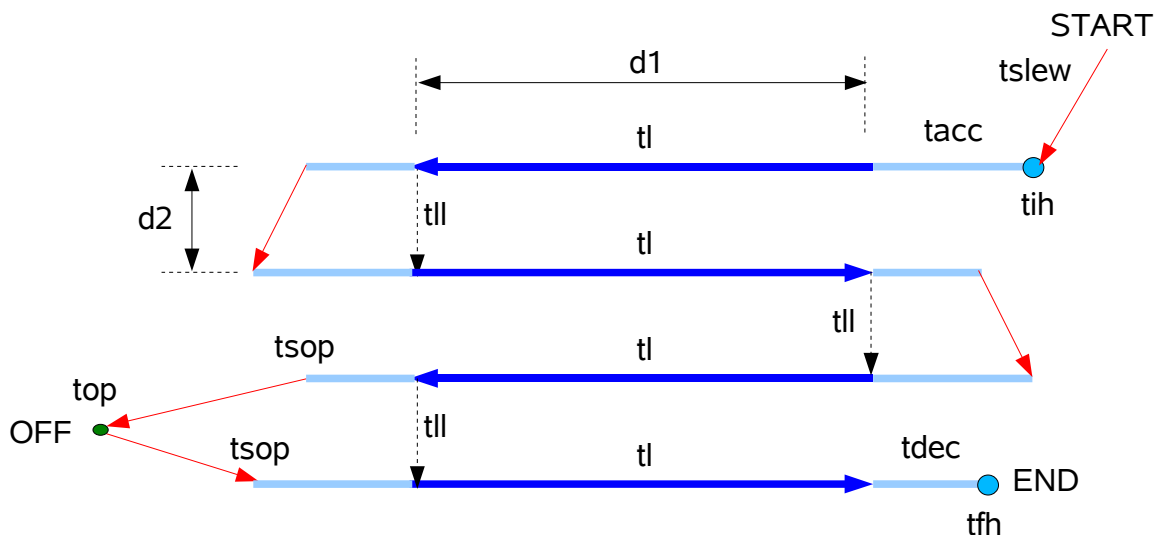


Figure 7: Basic_line-scan_pointing with $n=4$, $k=3$
(Note that 'tsop' includes acceleration/deceleration time)

After every 'k' lines, the boresight slews to a predefined OFF position, after which it continues with the next scan line. If $k=0$, no OFF positions occur. If 'n' is divisible by 'k', the pattern ends at the OFF position, as shown below:

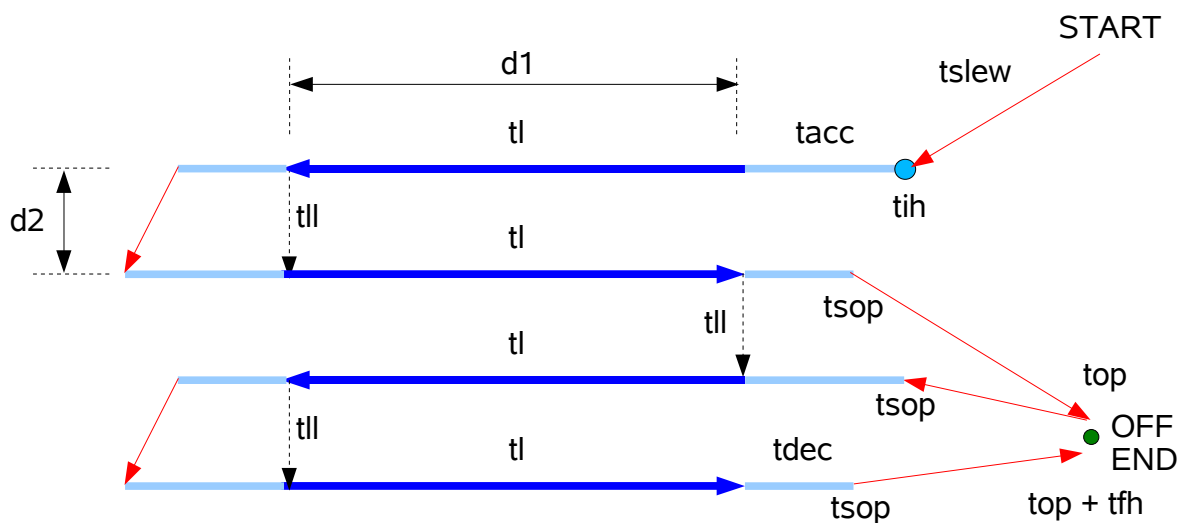



Figure 8: Basic_line-scan_pointing ending on OFF with $n=4$, $k=2$
(Note that 'tsop' includes acceleration/deceleration time)

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	36

The pattern centre and orientation are specified in the same way as for a `basic_raster`, using the parameters, 'ra', 'dec', 'fixed' and 'patt'.

3.3.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	line=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih + tacc	LINE	line++
LINE	k>0 & line%k=0	tl + tsop	OFF	
	line ≠ n	tl + tll	LINE	line++
	else	tl + tdec	FINAL_HOLD	
OFF	line ≠ n	top + tsop	LINE	line++
	else	top	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 9: State table for basic_line_scan_pointing

The `next_state` function returns the following array of integer values:

{state, time, line}

3.3.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend                {observation}
      tt   = tacc tx* tl (tdec | (tsop top)) {line-scan pattern}
      tx   = tl (tll | (tsop top tsop))     {one line}

```

Comments:

- The slew time 'tsop' is measured to/from the start/end of the scan period 'tl' and includes the acceleration/deceleration time.
- The slews to and from the OFF position do not all take equal time. The value 'tsop' is the worst-case value corresponding to the longest slew; the spacecraft simply waits at the next stable pointing position if the actual slew time is less than this.
- If `fixed=true`, the slew times to/from the OFF position do not vary with the time at which the observation is scheduled. If `fixed=false`, the worst-case slew time may be assumed, so that the duration of the observation is independent of when it is scheduled.

3.3.4 Parameters

The CUS command `basic_line_scan_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	(1)
zoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	(1)
n	i	[1,32]			
d1	d	[20,72000], k=0 [20,7200], k>0	5	arcsec	(2)
d2	d	0 or [2,480]	0.5	arcsec	
rate	d	[0.1,60]	0.1	arcsec/s	
k	i	[0,n], naifid=0 0, naifid>0			(3)
top	i	[0,50000]	1	s	
raoff	d	[0,360)		deg	(3)
decoff	d	[-90,90]		deg	(3)


Table 10: Parameters for basic_line_scan_pointing

Notes:

1. The offsets yoffset and zoffset have maximum absolute values of 3 arc-minutes (TBC) when an OFF position is used. The 2 degree limit on the ZY coordinates of the OFF position relative to the nominal centre is reduced by 3 arc-minutes to ensure that the OFF position is always within in allowed range relative to the actual (offset) centre, irrespective of the spacecraft roll angle.
2. The length of scan line is limited to 2 degrees when an OFF position is used, in order to avoid excessively long slews to/from the OFF position. This limitation is imposed by the ACMS.
3. This mode does not support OFF positions with SSO tracking.

3.3.5 Returned Values

The following array of times, in seconds, is returned:

 <i>HSC Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	38

{tobs, tslew, tacc, tdec, tl, tll, tsop, tend}

3.4 nodding_pointing

3.4.1 Description

The `nodding_pointing` mode switches the boresight repeatedly between two positions A and B on the sky. This can be used in one of two ways:

- to make repeated background measurements, without the use of a chopper. This is known as *position switching*.
- in conjunction with an instrument chopper in *chop/nod* mode. The spacecraft 'nods' to offset the telescope by exactly the same amount as the chopper throw, so that the target remains at the same point on the detector.

When used in chop/nod mode, the nod direction and amplitude must exactly match the chopper movement, such that the two cancel. Hence, the parameter 'fixed' must be set to false, so that the 'pattnod' parameter is with respect to the instrument boresight coordinates. In this case, the 'pattnod' nod angle should be set to the nominal chopping direction. The SIAM is then used to describe any misalignment of the chopper direction with respect to the spacecraft axes.

The pattern consists of 'nmod' AB or BA pairs, where 'A' is the on-target position and 'B' is the off-target position. Each pair counts as one nod. If the parameter 'startAtB' is 'false', the pattern starts at the ON position (A), otherwise it starts at the OFF position (B). This pattern is illustrated by the following table:

nmod	startAtB=false	startAtB=true
1	AB	BA
2	ABBA	BAAB
3	ABBAAB	BAABBA
4	ABBAABBA	BAABBAAB

Table 11.1: Nodding pattern

This pattern ensures that the total integration times on the two points is equal. However, the mode allows different A and B times, 'tpa' and 'tpb', to be specified if required. There is a brief pause of duration 'tss' (typically 2 seconds) between each pair.

This mode supports both holds and load slews:

- Every `nload`'th nod is a load slew to allow load calibrations to take place during the slew. If `nload=0`, no loads occur. A minimum duration 'tloadmin' can be specified; the spacecraft slews to the next position and then waits, if necessary, until the requested time has elapsed.



- After every 'nhold' nods, a hold of duration 'thold' occurs at a fixed pointing for calibration. This occurs between the pair of points AA (or BB). If nhold=0, no hold occur.

The RA and DEC coordinates, with any ZY offset applied, specify the position of point A. Point B is then defined relative to A by a direction 'pattnod' and a distance 'chopthrow'. If the parameter 'fixed' is true, 'pattnod' is a position angle on the sky; otherwise it is measured anticlockwise about the X axis from Z.

The pattern is illustrated by the following diagram. The pattern ends on a hold because 'n' is divisible by 'nhold':

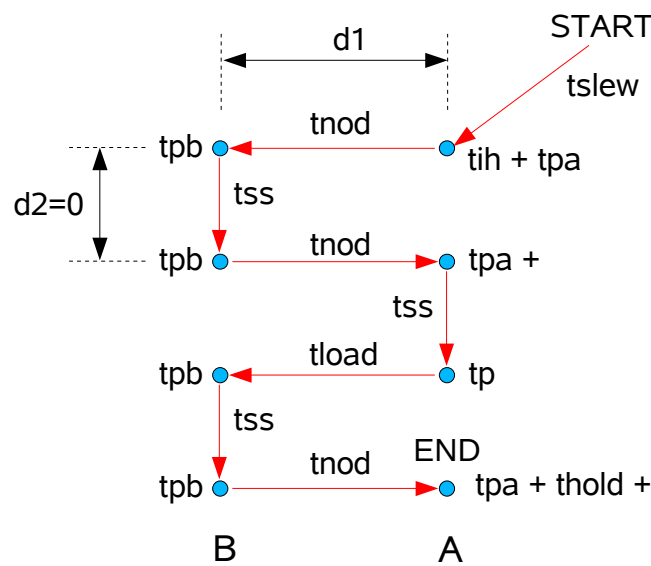



Figure 9: Nodding pointing with $n=4$, $nhold=2$, $nload=3$, $startAtB=false$ (ABBAABBA)

3.4.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	j=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD	startAtB else	tih tih	NOD(2) POINT(1)	j++ j++
POINT(1)	nload>0 & j%nload=0 else	tpa tpa + tnod	LOAD(1) NOD(1)	
NOD(1)	nhold>0 & j%nhold=0 j<nnod else	tpb tpb + tss tpb	HOLD(1) NOD(2) FINAL_HOLD	j++
NOD(2)	nload>0 & j%nload=0 else	tpb tpb + tnod	LOAD(2) POINT(2)	
POINT(2)	nhold>0 & j%nhold=0 j<nnod else	tpa tpa + tss tpa	HOLD(2) POINT(1) FINAL_HOLD	j++

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	41

State	Condition	Duration	Next state	Action
LOAD(1)		tload	NOD(1)	
LOAD(2)		tload	POINT(2)	
HOLD(1)	j<nnod	thold + tss	NOD(2)	j++
	else	thold	FINAL_HOLD	
HOLD(2)	j<nnod	thold + tss	POINT(1)	j++
	else	thold	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 11.2: State table for nodding_pointing.

The next_state function returns the following array of integer values:

{state, time, j}

3.4.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih [tt1 | tt2] tfh tend      {observation}
      tt1 = txa (tss txb tss txa)* [tss txb] {nodding from A}
      tt2 = txb (tss txa tss txb)* [tss txa] {nodding from B}
      txa = tpa (tnod | tload) tpb [thold]  {one forward nod}
      txb = tpb (tnod | tload) tpa [thold]  {one backward nod}

```

The slew-to-self time 'tss' is expected to be one second.

The time 'tload' is the greater of the actual slew and the value requested by 'tloadmin'.

3.4.4 Parameters

The CUS command nodding_pointing has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
pattnod	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	
zoffset	d	[-18000,18000]		arcsec	

Parameter	Type	Range	Resolution	Units	Notes
nmod	i	[1,1200]			
chopthrow	d	[2,7200]	0.5	arcsec	
tpa	i	[10,50000]	1	s	
tpb	i	[10,50000]	1	s	
tloadmin	i	≥ 0	1	s	
nload	i	[0,1200]			
thold	i	≥ 0	1	s	
nhold	i	[0,1200]			
startAtB	b				

Table 12: Parameters for nodding_pointing

3.4.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tnod, tss, tload, tend}



Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	43

3.5 gyro_nodding_pointing

3.5.1 Description

The `gyro_nodding_pointing` mode is a nodding mode that uses on-board gyro propagation. The boresight repeatedly switches between two positions A and B on the sky, starting in a predefined OFF position which is the same as the A nodding position, as shown in figure 10:

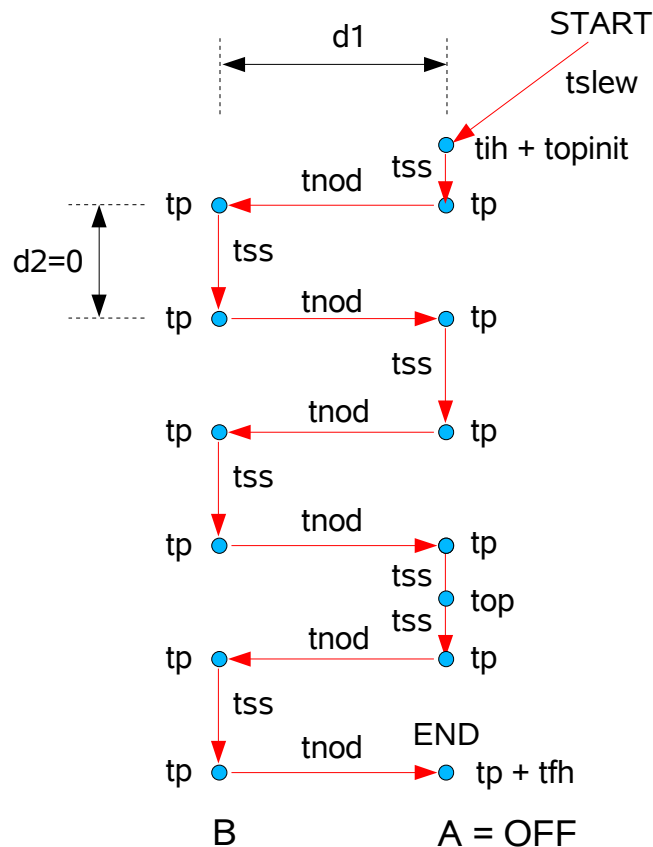



Figure 10: *Gyro_nodding_pointing* with $n=6$, $koff=4$

This mode is similar to the `nodding_pointing` mode but it does not support holds or load slews. It is implemented an ACMS raster command with a line spacing of zero. Therefore, the maximum number of AB switches corresponds to the maximum number of raster lines, i.e. 32.

The pointing starts at the OFF position then proceeds to do ABBAAB... nods. It is necessary to return to OFF (not simply A) at least every 'tmax' seconds, where this is the maximum period for which the SRPE requirement can be achieved.

This is achieved by calculating a value 'koff', which is the maximum number of switches before returning to the OFF position:

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	44

$$koff = \left(\frac{k}{4}\right) * 2$$

where

$$k = \text{floor} \left(\frac{t_{max} - t_{ss} + \max(t_{nod}, t_{ss})}{t_p + \max(t_{nod}, t_{ss})} \right)$$

Note this formula is for information only and should not be used. CUS scripts should not rely on it as future optimizations could change the calculation.

Hence, the boresight slews to the predefined OFF position after every 'koff' switches:

koff=2: OFF ABBA OFF ABBA OFF...
koff=4: OFF ABBAABBA OFF...

Odd values for 'koff' are not allowed, as this would involve two extra slews.

However, the total number of nods 'nnod' may be ODD, in which case the pattern does not finish at the OFF position:

koff=2, nnod=3: OFF ABBA OFF AB

If nnod is divisible by koff, the pattern ends on the OFF position except when nnod = koff, in which case the last OFF is omitted, because it is unnecessary. E.g:

koff=2, nnod=4: OFF ABBA OFF ABBA OFF
koff=4, nnod=4: OFF ABBAABBA

This mode does not support SSO tracking.

3.5.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	j=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	OFF1	
OFF1		topinit + tss	POINT(1)	j++
OFF	j < nnod else	top + tss top	POINT(1) FINAL_HOLD	j++
POINT(1)		tp + tnod	NOD(1)	
NOD(1)	j < nnod else	tp + tss tp	NOD(2) FINAL_HOLD	j++
NOD(2)		tp + tnod	POINT(2)	
POINT(2)	j % koff == 0 & nnod != koff j < nnod else	tp + tss tp + tss tp	OFF POINT(1) FINAL_HOLD	j++
FINAL_HOLD		tfh + tend	END	

Table 11.2: State table for gyro_nodding_pointing.

The `next_state` function returns the following array of integer values:

```
{state, time, j}
```

3.5.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                               {slew + obs}
where tobs = tih topinit tt tfh tend      {observation}
      tt  = (tss ty [tss top])* [tss tx]  {nodding sequence}
      ty  = tx tss tx                     {pair of nods}
      tx  = tp tnod tp                     {one nod}

```

The slew-to-self time 'tss' is the time taken to perform a zero length slew.

3.5.4 Parameters


The CUS command `gyro_nodding_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
pattnod	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	
zoffset	d	[-18000,18000]		arcsec	
nnod	i	[2,32]			
chopthrow	d	[2,7200]	0.5	arcsec	
tp	i	[10,50000]	1	s	
top	i	[0,1000]	1	s	(1)
topinit	i	[0,1000]	1	s	(2)
tmax	i	[0,1000]	1	s	(3)

Table 12: Parameters for gyro_nodding_pointing

Notes:

1. The recommended value for top is 60 seconds.
2. The recommended value for topinit is 60 seconds.

 <i>HSC</i> <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	46

3. The recommended value for tmax is 600 seconds.

3.5.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tnod, tss, koff, tend}

Note that 'koff' is a dimensionless number and not a time.

3.6 raster_pointing

3.6.1 Description

A `raster_pointing` is similar to a basic raster, but has less restrictions as it is not implemented by a single telecommand. For example:

- The raster may be up to 100 x 100 points.
- The whole pattern may be repeated 'nrepeat' times.
- When an OFF position is used, the pattern starts with an OFF. This is more symmetrical than the basic raster, where the time between the first raster point and an OFF is larger than for any other point in the raster.
- After every 'nhold' raster points, a hold of duration 'thold' occurs at a fixed pointing for calibration. If `nhold=0`, no holds occurs.

The raster centre and orientation are described in the same way as a `basic_raster`, by the parameters 'ra', 'dec', 'fixed' and 'patt'.

The diagram below illustrates a simple case, where $m*n$ is divisible by 'nhold' so that the holds occur at the same raster points on each repetition:

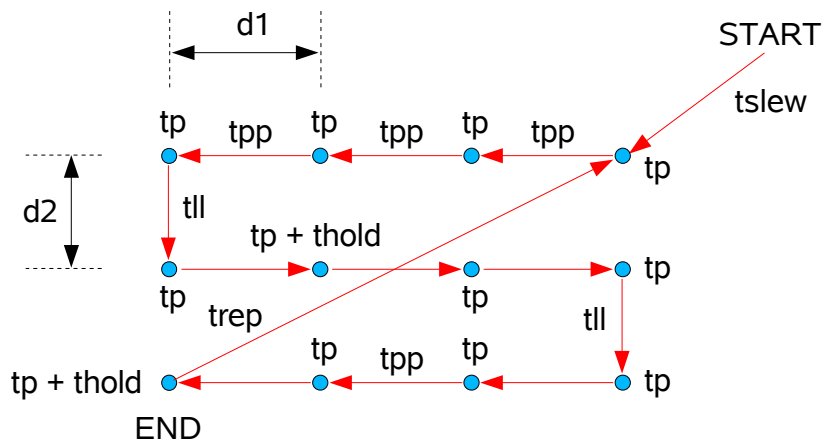


Figure 10: `Raster_pointing` with $m=4$, $n=3$, $k=0$, `nhold=6`, `nrepeat>1`
tih and tfh are omitted for clarity

3.6.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	$p=0, r=0, i=0$
SLEW		tslew	INIT_HOLD	
INIT_HOLD	$k > 0$ else	tih	OFF(1)	
OFF(1)		top + tsop	POINT	$p=1, r++, i++$
POINT	$nhold > 0$ & $i \% nhold = 0$ $k > 0$ & $i \% k = 0$	tp	HOLD	
		tp + tsop	OFF(2)	

State	Condition	Duration	Next state	Action
	mi ≠ m	tp + tpp	POINT	p++, i++
	ni ≠ n	tp + tll	POINT	p++, i++
	r ≠ nrepeat	tp + trep	POINT	p=1, r++, i++
	else	tp	FINAL_HOLD	
HOLD	k>0 & i%k=0	thold + tsop	OFF(2)	
	mi ≠ m	thold + tpp	POINT	p++, i++
	ni ≠ n	thold + tll	POINT	p++, i++
	r ≠ nrepeat	thold + trep	POINT	p=1, r++, i++
	else	thold	FINAL_HOLD	
OFF(2)	!(ni=n & mi=m)	top + tsop	POINT	p++, i++
	r ≠ nrepeat	top + tsop	POINT	p=1, r++, i++
	else	top	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 13: State table for raster_pointing

The next_state function returns the following array of integer values:

{state, time, p, r}

3.6.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend                {observation}
    tt      = [top tsop] (tx [tsop top])+    {repetitions}
    tx      = ty ((tll | tsop top tsop) ty)* {raster}
    ty      = tp ((tpp | tsop top tsop) tp)+ {one raster line}
    tz      = tp [thold]                     {one raster point}

```

The basic raster mode is effectively a special case of this mode, with nhold=0 and nrepeat=1, although it is implemented by a single raster telecommand.

3.6.4 Parameters

The CUS command raster_pointing has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	



Parameter	Type	Range	Resolution	Units	Notes
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	
zoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	
m	i	[2,100]			
n	i	[1,100]			
d1	d	[2,480]	0.5	arcsec	
d2	d	[2,480]	0.5	arcsec	
tp	i	[10,50000]	1	s	
nrepeat	i	[1,100]			
thold	i	≥ 0		s	
nhold	i	[0,m*n]			
k	i	[0,m*n]			
top	i	[0,50000]		s	
raoff	d	[0,360), naifid=0 [2,2], naifid>0		deg deg	
decoff	d	[-90,90], naifid=0 [-2,2], naifid>0		deg deg	

Table 14: Parameters for raster_pointing

Notes:

1. The maximum size of the raster is 14880 x 14880 arcsec⁵

3.6.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tpp, tll, trep, tsop, tend}

⁵ This is the same as maximum size of a basic_raster_pointing (31 x 480 arcsec)



Figure 11: Nodding_raster_pointing with $m=3, n=2, nnod=2, nload=0, nhold=0, k=0$. Multiple lines passing through a point are displaced to show the path.

When 'nnod' is odd, a slew occurs from the 'nod' position to the 'unnodded' next raster position. The following diagram illustrates this for the case where $nnod=1$, with the pattern finishing at the OFF position:

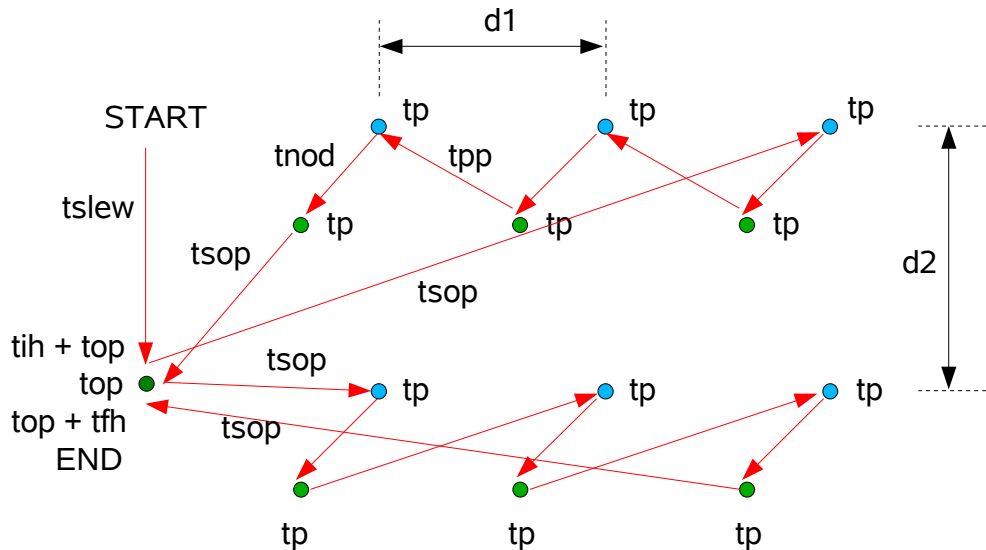


Figure 12: Nodding_raster_pointing ending on the OFF position, with $m=3, n=2, nnod=1, nload=0, nhold=0, k=3$

3.7.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	$p=0, r=0, i=0, j=0$
SLEW		tslew	INIT_HOLD	
INIT_HOLD	$k>0$	tih	OFF(1)	
INIT_HOLD	else	tih	POINT(1)	$p=1, r=1, i=1, j++$
OFF(1)		$top + tsop$	POINT(1)	$p=1, r=1, i=1, j++$
POINT(1)	$nload>0 \ \& \ itot\%nload=0$	tp	LOAD(1)	
POINT(1)	else	$tp + tnod$	NOD(1)	
NOD(1)	$j<nnod$	$tp + tss$	NOD(2)	$j++$
NOD(1)	$nhold>0 \ \& \ j\%nhold=0$	tp	HOLD	
NOD(1)	$k>0 \ \& \ i\%k=0$	$tp+tsop$	OFF(2)	
NOD(1)	$mi \neq m$	$tp + tpp$	POINT(1)	$p++, i++, j=1$
NOD(1)	$ni \neq n$	$tp + tll$	POINT(1)	$p++, i++, j=1$
NOD(1)	$r \neq nrepeat$	$tp + trep$	POINT(1)	$p=1, r++, i++, j=1$
NOD(1)	else	tp	FINAL_HOLD	
LOAD(1)		tload	NOD(1)	
NOD(2)	$nload>0 \ \& \ j\%nload=0$	tp	LOAD(2)	
NOD(2)	else	$tp + tnod$	POINT(2)	
POINT(2)	$j<nnod$	$tp + tss$	POINT(1)	$j++$
POINT(2)	$nhold>0 \ \& \ itot\%nhold=0$	tp	HOLD	
POINT(2)	$k>0 \ \& \ i\%k=0$	$tp + tsop$	OFF(2)	

State	Condition	Duration	Next state	Action
	mi ≠ m	tp + tpp	POINT(1)	p++, i++, j=1
	ni ≠ n	tp + tll	POINT(1)	p++, i++, j=1
	r ≠ nrepeat	tp + trep	POINT(1)	p=1, r++, i++, j=1
	else	tp	FINAL_HOLD	
LOAD(2)		tload	POINT(2)	
HOLD	k>0 & i%k=0	thold+tsop	OFF(2)	
	mi ≠ m	thold + tpp	POINT(1)	p++, i++, j=1
	ni ≠ n	thold + tll	POINT(1)	p++, i++, j=1
	r ≠ nrepeat	thold + trep	POINT(1)	p=1, r++, i++, j=1
	else	thold	FINAL_HOLD	
OFF(2)	!(ni=n & mi=m)	top + tsop	POINT(1)	p++, i++, j=1
	r ≠ nrepeat	top + tsop	POINT(1)	p=1, r++, i++, j=1
	else	top	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 15: State table for nodding_raster_pointing

The next_state function returns the following array of integer values:

{state, time, p, j, r}

3.7.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend                {observation}
    tt = [top tsop] (tw [tsop top]) +       {repetitions}
    tw = tx ((tll | tsop top tsop) tx) *   {raster}
    tx = ty ((tpp | tsop top tsop) ty) +   {one raster line}
    ty = tz (tss tz) * [thold]              {nods at one point}
    tz = tp (tnod | tload) tp               {one nod}

```

When 'nnod' is odd, the actual slew time 'tpp' differs between odd and even lines, as can be seen from the diagrams above. In addition, when the raster orientation is specified relative to the sky, the direction of the nods relative to the raster depends on the spacecraft roll angle and hence on the time at which the observation is performed. The value 'tpp' returned by the CUS pointing mode is a worst-case value, so that the observation has a fixed duration.

3.7.4 Parameters

The CUS command nodding_raster_pointing has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	



Parameter	Type	Range	Resolution	Units	Notes
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	
zoffset	d	[-18000,18000], k=0 [-180,180], k>0		arcsec	
m	i	[2,100]			
n	i	[1,100]			
d1	d	[2,480]	0.5	arcsec	
d2	d	[2,480]	0.5	arcsec	
tp	i	[10,50000]	1	s	
thold	i	≥ 0	1	s	
nhold	i	[0,m*n*nrepeat]			
pattnod	d	[0,360)		deg	
chopthrow	d	[2,960]	0.5	arcsec	
nnod	i	[1,32]			
k	i	[0,m*n*nrepeat]			
top	i	[0,50000]	1	s	
raoff	d	[0,360), naifid=0 [-2,2], naifid>0		deg deg	
decoff	d	[-90,90], naifid=0 [-2,2], naifid>0		deg deg	
nrepeat	i	[1,100]			
trepeatmin	i	≥ 0		s	
tloadmin	i	≥ 0		s	
nload	i	[0,nnod]			


Table 16: Parameters for nodding_raster_pointing

Notes:

1. The maximum size of the raster (excluding the nods) is 14880 x 14880 arcsec.

3.7.5 Returned Values

The following array of times, in seconds, is returned:

 <i>HSC Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	54

{tobs, tslew, tpp, tss, tll, tnod, tload, tsop, trep, tend}



Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	55

3.8 nodding_of_raster_pointing

3.8.1 Description

This mode is similar to `nodding_raster_pointing`, except that nodding is performed at a much slower rate, such that there are several raster points per nod phase, rather than several nods per raster point.

It consists of a raster 'map-1' and a second identical raster 'map-2' that is offset by the chopper throw. The pattern starts with 'k' points of map-1, then nods to map-2 for $2*k$ points, then back to map-1 for $2*k$ points, and so on, finally finishing with 'k' points in the last map. This is illustrated in figure 13.

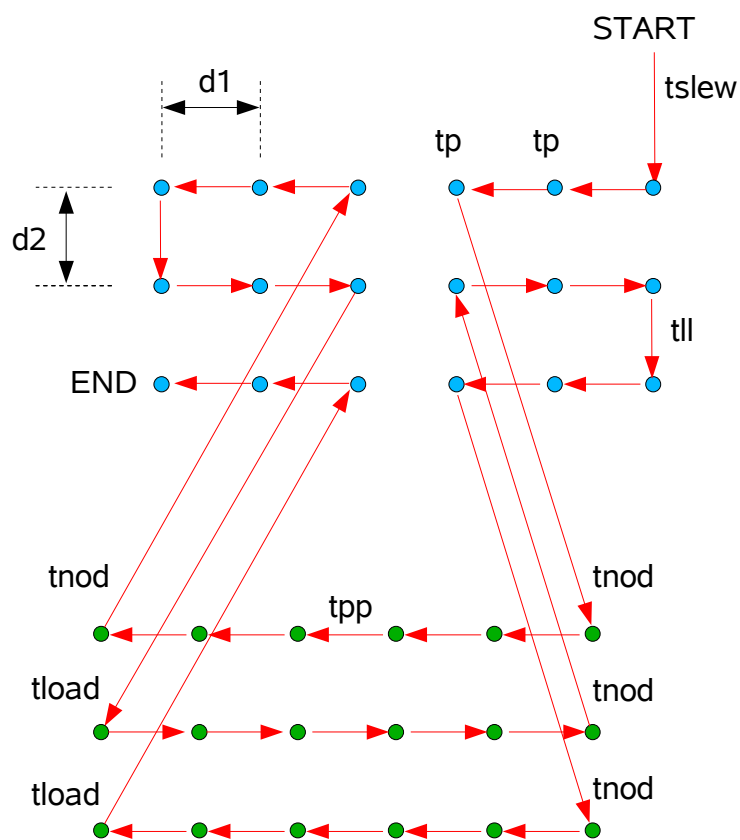



Figure 13: *Nodding_of_raster_pointing*, with $m=6$, $n=3$, $knod=3$, $nload=3$, $ncycles=1$, $nrepeat=1$ (*tih* and *tjh* are omitted for clarity).

The raster centre and orientation, as well as the nod distance and direction are defined in the same way as for `nodding_raster_pointing`.

Holds and load-slews are supported as follows:

- Every $nload$ 'th nodding slew is a load. If $nload=0$, no loads occur.
- The raster is repeated ' $ncycles$ ' times. This whole pattern is repeated ' $nrepeat$ ' times, with a hold of duration ' $thold$ ' between repetitions to allow retuning of the HIFI local oscillator.

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	56

The parameter 'knod' must be chosen such that the number of raster points $m*n$ is divisible by 'knod'. This results in a pattern that has the following properties:

- A cycle may start and end in different maps.
- The next cycle always starts in the same map as the end of current cycle. Hence a repeat slew (trep) cannot involve a nod.
- A HOLD is never followed directly by a nod to the other map.

The following examples illustrates various patterns. The raster points are denoted by A_1, A_2, A_3, \dots and the corresponding nod positions by B_1, B_2, B_3 . The symbols 'c' and 'r' are the cycle count and repeat count, respectively.

Example 1: $m*n=8$, $knod=2$, $ncycles=3$, $nrepeat=2$:

$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=1, c=1$
$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=1, c=2$
$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=1, c=3$
hold	
$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=2, c=1$
$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=2, c=2$
$A_1A_2B_1B_2B_3B_4A_3A_4A_5A_6B_5B_6B_7B_8A_7A_8$	$r=2, c=3$

Example 2: $m*n=3$, $knod=3$, $ncycles=3$, $nrepeat=2$

$A_1A_2A_3B_1B_2B_3$	$r=1, c=1$
$B_1B_2B_3A_1A_2A_3$	$r=1, c=2$
$A_1A_2A_3B_1B_2B_3$	$r=1, c=3$
hold	
$B_1B_2B_3A_1A_2A_3$	$r=2, c=1$
$A_1A_2A_3B_1B_2B_3$	$r=2, c=2$
$B_1B_2B_3A_1A_2A_3$	$r=2, c=3$

Example 3: $m*n=3$, $knod=1$, $ncycles=2$, $nrepeat=1$

$A_1B_1B_2A_2A_3B_3$	$r=1, c=1$
$B_1A_1A_2B_2B_3A_3$	$r=1, c=2$
$A_1B_1B_2A_2A_3B_3$	$r=1, c=3$

3.8.2 State Table

In the following table, the variable 'nod' is given by:

$$\text{nod} = (i + \text{knod}) \% (2 \text{ knod}) = 0$$

State	Condition	Duration	Next state	Action
START			SLEW	p=0, i=0, j=0, c=0, r=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	POINT	p=1, i=1, j=1, c=1, r=1
POINT	nod & nload>0 & j%nload=0	tp	LOAD1	p=p+knod-1, i++, j++
	nod	tp + tnod	NOD	p=p+knod-1, i++, j++
	mi ≠ m	tp + tpp	POINT	p++, i++
	ni ≠ n	tp + tll	POINT	p++, i++
	c ≠ ncycles	tp + trep	POINT	p=1, i++, c++
	c=ncycles & r ≠ nrepeat	tp	HOLD1	
	else	tp	FINAL_HOLD	
NOD	nod & nload>0 & j%nload=0	tp	LOAD2	p=p+knod-1, i++, j++
	nod	tp + tnod	POINT	p=p+knod-1, i++, j++
	mi ≠ m	tp + tpp	NOD	p++, i++
	ni ≠ n	tp + tll	NOD	p++, i++
	c ≠ ncycles	tp + trep	NOD	p=1, i++, c++
	c=ncycles & r ≠ nrepeat	tp	HOLD2	
	else	tp	FINAL_HOLD	
HOLD1		thold + trep	POINT	p=1, i=1, c=1, r++
HOLD2		thold + trep	NOD	p=1, i=1, c=1, r++
LOAD1		tload	NOD	
LOAD2		tload	POINT	
FINAL_HOLD		tfh + tend	END	

Table 17: State table for nodding_of_raster_pointing

The next_state function returns the following array of integer values:

{state, time, p, c, r}

3.8.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                               {slew + observation}
where tobs = tih tt tfh tend               {complete mode}
    tt    = tv (thold trep tv)*           {repetitions}
    tv    = tw (trep tw)*                 {raster cycles}
    tw    = tx (tll tx)*                  {n raster lines}
    tx    = ty (tpp ty)*                  {one raster line}
    ty    = tz (tnod | tload) tz          {k-nod-k}
    tz    = tp ((tpp | tll) tp)*         {k points}

```

Note: No 'tss' delay is required for this mode because there are no zero-length slews.

3.8.4 Parameters

The CUS command `nodding_of_raster_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	
zoffset	d	[-18000,18000]		arcsec	
m	i	[2,100]			
n	i	[1,100]			
d1	d	[2,480]	0.5	arcsec	
d2	d	[2,480]	0.5	arcsec	
tp	i	[10,50000]	1	s	
thold	i	≥ 0	1	s	
pattnod	d	[0,360)		deg	
chopthrow	d	[2,480]	0.5	arcsec	
ncycles	i	[1,100]			
knod	i	[1,m*n]			(1)
nrepeat	i	[1,10]			
tloadmin	i	≥ 0		s	
nload	i	[0,ncycles*m*n/knod]			


Table 18: Parameters for nodding_of_raster_pointing

Notes:

1. $n*m$ should be divisible by knod.
1. The maximum size of the raster (excluding the nods) is 14880 x 14880 arcsec.

3.8.5 Returned Values

The following array of times, in seconds, is returned:

 <i>HSC</i> <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	59

{tobs, tslew, tpp, tll, tnod, tload, trep, tend}

3.9 line_scan_pointing

3.9.1 Description

A `line_scan_pointing` is similar to a `basic_line_scan_pointing`, but has less restrictions as it is not implemented by a single telecommand. For example:

- There may be up to 1500 lines.
- The whole pattern may be repeated 'nrepeat' times.
- After every 'nhold' lines, a hold of duration 'thold' occurs at a fixed pointing for calibration. If `nhold=0`, no holds occur.

The centre and orientation are described in the same way as a `basic_raster`, by the parameters 'ra', 'dec', 'fixed' and 'patt'.

The following example illustrates a pattern that ends with a 'hold':

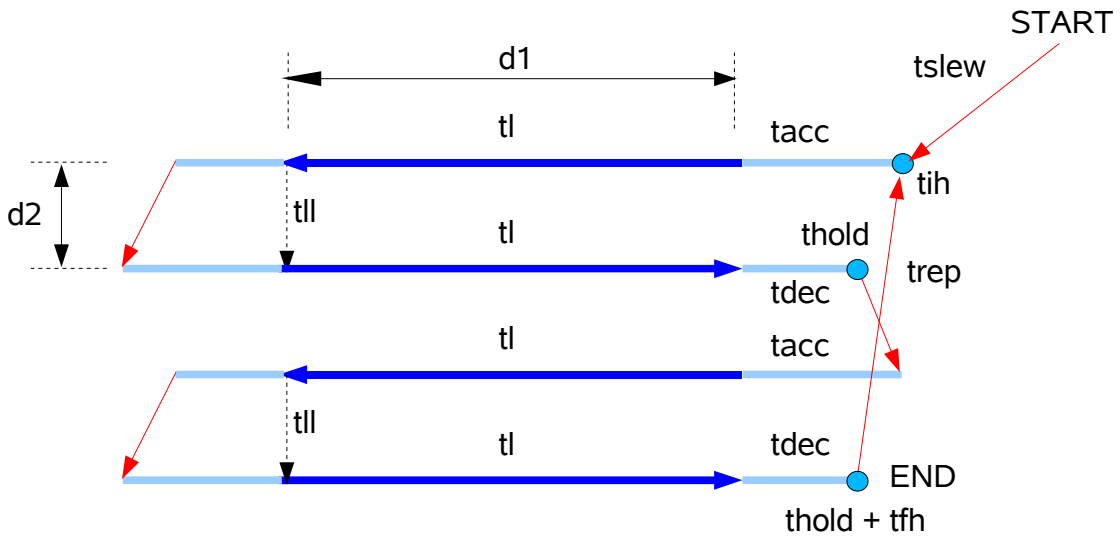


Figure 14: `Line_scan_pointing`, `n=4`, `nhold=2`, `nrepeat>1`
 The pattern ends with a 'hold'

3.9.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	line=0, i=0, r=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih + tacc	LINE	line=1, i=1, r=1
LINE	nhold>0 & i%nhold=0	tl + tdec	HOLD	
	line ≠ n	tl + tll	LINE	line++, i++
	r ≠ nrepeat	tl + tdec + trep + tacc	LINE	line=1, i++, r++
	else	tl + tdec	FINAL_HOLD	
HOLD	line ≠ n	thold + tll - tdec	LINE	line++, i++
	r ≠ nrepeat	thold + trep + tacc	LINE	line=1, i++, r++
	else	thold	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 19: State table for line_scan_pointing

The next_state function returns the following array of integer values:

{state, time, line, r}

3.9.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend              {observation}
      tt  = (tx trep)*                     {repetitions}
      tx  = (ty tz)* ty                    {one repetition}
      ty  = tacc tl tdec [thold]           {one line}
      tz  = (tll - tacc - tdec)            {slew between lines}

```

3.9.4 Parameters

The CUS command line_scan_pointing has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	

Parameter	Type	Range	Resolution	Units	Notes
zoffset	d	[-18000,18000]		arcsec	
n	i	[1,1500]			
d1	d	[20,72000]	5	arcsec	
d2	d	0 or [2,480]	0.5	arcsec	
rate	d	[0.1,60]	0.1	arcsec/s	
thold	i	≥ 0	1	s	
nhold	i	[0,n]			
nrepeat	i	[1,100]			

Table 20: Parameters for line_scan_pointing

Notes:

1. The maximum height of the map [$(n - 1) * d2$] is 14880 arcsec.

3.9.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tacc, tdec, tl, tll, trep, tend}



Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	63

3.10 line_scan_with_off_pointing

3.10.1 Description

This is a special mode intended for use with the HIFI instrument. It is similar to a `line_scan_pointing`, with the addition of an OFF position.

The line-scan pattern is repeated 'nrepeat' times. Each repetition starts and ends at the OFF position as follows, where 'map' is the sequence of 'n' scan lines:

OFF map OFF map OFF map OFF

After every 'nhold' repetitions, a hold occurs so that the local oscillator frequency can be adjusted. Since, each repetition starts and ends at OFF, the resulting pattern is:

OFF map OFF map OFF HOLD OFF map OFF map OFF

Load slews are supported as follows:

- Every nload'th slew to the OFF position is a load. If nload=0, no loads occur.

The following diagram illustrates the pattern:

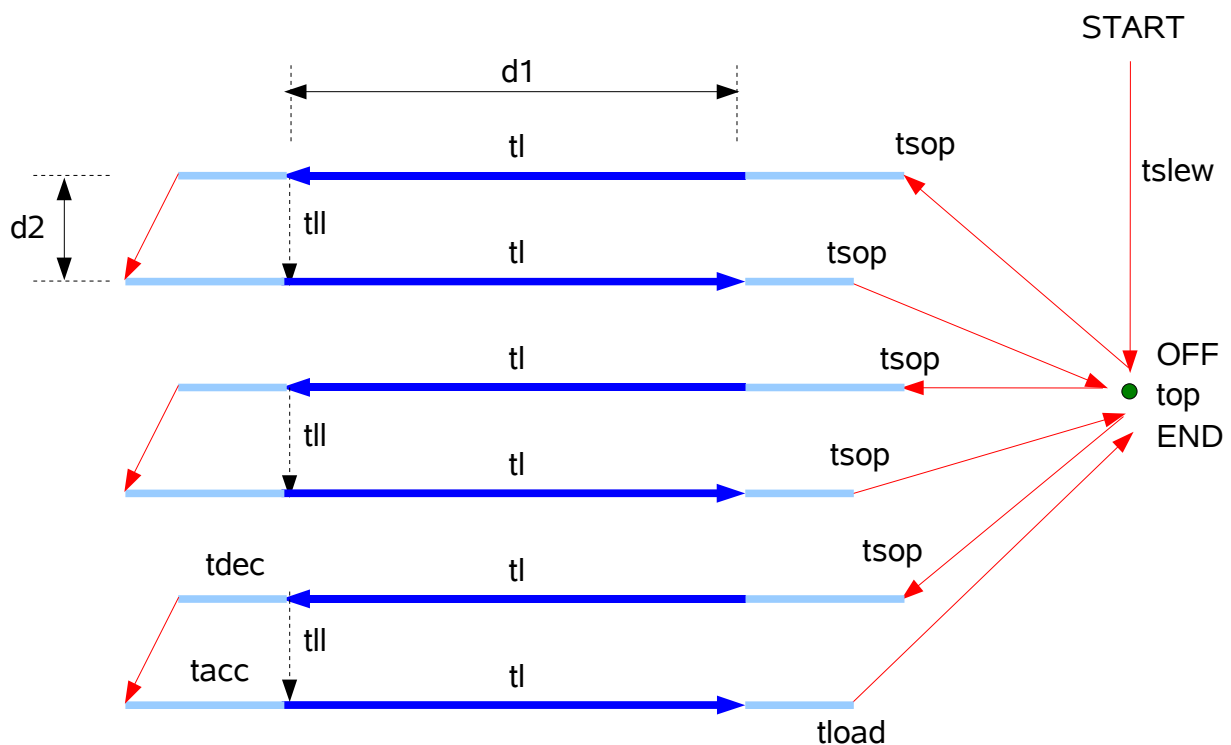


Figure 15: `line_scan_with_off_pointing` with $n=6$, $k=2$, $nload=3$, $nrepeat=2$.
' t_{sop} ' includes the acceleration/deceleration time.
' t_{load} ' includes the deceleration time. t_{ih} and t_{fh} are omitted for clarity.



The use of holds is illustrated by the following diagram:

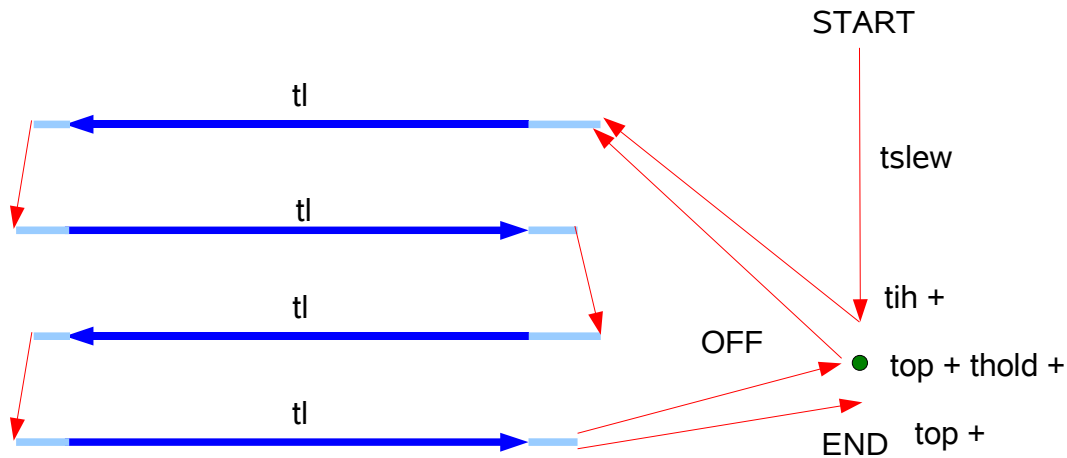


Figure 16: *Line_scan_with_off_pointing*, showing the use of hold, with $n=4$, $k=4$, $nload=0$, $repeat=2$, $nhold=1$. Paths through the OFF position are shown separated for clarity.

3.10.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	line=0, r=0, off=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	OFF(1)	line=1, r++
OFF(1)		top + tsop	LINE	
LINE	line%k ≠ 0 & line ≠ n	tl + tll	LINE	line++
	nload>0 & (off+1)%nload=0	tl	LOAD	
	else	tl + tsop	OFF(2)	off++
OFF(2)	line ≠ n	top + tsop	LINE	line++
	r = nrepeat	top	FINAL_HOLD	
	nhold>0 & r%nhold=0	top	HOLD	
	else	top + tsop	LINE	line=1, r++
HOLD	r ≠ nrepeat	thold	OFF(1)	r++, line=1
	else	thold	FINAL_HOLD	
LOAD		tload	OFF(2)	off++
FINAL_HOLD		tfh + tend	END	

Table 21: *State table for line_scan_with_off_pointing*

The next_state function returns the following array of integer values:

{state, time, line, r}

3.10.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```
tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend                {observation}
```




Doc. No	HERSCHEL-HSC-DOC-624
Issue	2.4
Date	10.09.10
Page	65

```

tt  = top tx* [thold top] tx          {repetitions}
tx  = tsop ty* tl (tsop | tload) top  {one repetition}
ty  = tl (tll | tsop top tsop)       {one line}

```


Each repetition of the map finishes on the OFF position and consequently the complete pattern also finishes on the OFF position. For a regular pattern, 'n' should be divisible by 'k', otherwise the last OFF in each repetition will occur after fewer than 'k' lines.

The repetitions are not necessarily exactly the same, since the loads may not occur on the same lines each time.

3.10.4 Parameters

The CUS command `line_scan_with_off_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-180,180]		arcsec	
zoffset	d	[-180,180]		arcsec	
n	i	[1,240]			
d1	d	[20,7200]	5	arcsec	
d2	d	0 or [2,480]	0.5	arcsec	
rate	d	[0.1,60]	0.1	arcsec/s	
k	i	[1,n]			
top	i	[0,50000]	1		
raoff	d	[0,360), naifid=0 [-2,2], naifid>0		deg deg	
decoff	d	[-90,90], naifid=0 [-2,2], naifid>0		deg deg	
nrepeat	i	[1,1200]			
thold	i	≥ 0	1	s	
nhold	i	[0, nrepeat]			
tloadmin	i	≥ 0	1	s	

 HSC Development	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	66

Parameter	Type	Range	Resolution	Units	Notes
nload	i	[0,32]			

Table 22: Parameters for line_scan_with_off_pointing

Notes:

1. The maximum height of the map [$(n - 1) * d2$] is 14880 arcsec.

3.10.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tacc, tdec, tl, tll, tsop, tload, tend}

3.11 custom_map_pointing

3.11.1 Description

This mode consists of a sequence of fine pointings and nodding pointings, specified by an array of coordinates.

A Cartesian coordinate frame is specified by the parameters 'ra', 'dec', 'fixed' and 'patt', in the same way as other modes such as basic_raster. The positions of the pointings are specified by two arrays, 'yarray' and 'zarray', with respect to this coordinate system.

By default, all the pointings are fine pointings of duration 'tp'. However, the parameter 'knod' can be used to specify that the first and every 'knod' positions thereafter are nodding, with 'nnod' nods. In this case, the coordinates specify the A position. The durations of the A and B positions are both 'tp'.

Holds and load slews are supported as follows:

- On every nload'th nod, a 'load' occurs. If nload=0, no loads occur. The counting is reset to zero at the start of each new pointing position.
- After every 'nhold' array positions, a hold of duration 'thold' occurs at a fixed pointing for calibration.

The following diagram shows an example:

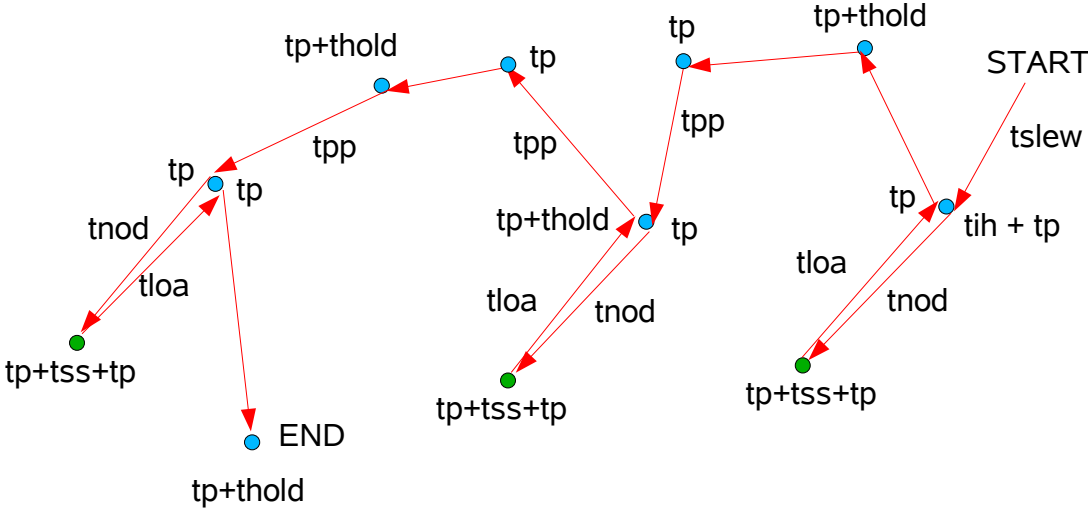


Figure 17: Custom_map_pointing, with 8 points, knod=3, nmod=2, nload=2, nhold=3

3.11.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	p=0, j=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD	knod>0 & p%knod=0 else	tih tih	POINT(1) POINT(0)	p++, j=1 p++
POINT(0)	nhold>0 & p%nhold=0 p = n knod>0 & p%knod=0 else	tp[p] tp[p] tp[p]+tpp[p] tp[p]+tpp[p]	HOLD FINAL_HOLD POINT(1) POINT(0)	 p++, j=1 p++, j=0
POINT(1)	nload>0 & j%nload=0 else	tp[p] tp[p]+tnod	LOAD(1) NOD(1)	
LOAD(1)		tload	NOD(1)	
NOD(1)	j<nnod nhold>0 & p%nhold=0 p = n knod>0 & p%knod=0 else	tp[p]+tss tp[p] tp[p] tp[p]+tpp[p] tp[p]+tpp[p]	NOD(2) HOLD FINAL_HOLD POINT(1) POINT(0)	j++ p++, j=1 p++, j=0
NOD(2)	nload>0 & j%nload=0 else	tp[p] tp[p]+tnod	LOAD(2) POINT(2)	
LOAD(2)		tload	POINT(2)	
POINT(2)	j<nnod nhold>0 & p%nhold=0 p = n knod>0 & p%knod=0 else	tp[p]+tss tp[p] tp[p] tp[p]+tpp[p] tp[p]+tpp[p]	POINT(1) HOLD FINAL_HOLD POINT(1) POINT(0)	j++ p++, j=1 p++, j=0
CAL__HOLD	p = n knod>0 & p%knod=0 else	thold thold+tpp[p] thold+tpp[p]	FINAL_HOLD POINT(1) POINT(0)	 p++, j=1 p++, j=0
FINAL_HOLD		tfh + tend	END	

Table 23: State table for custom_map_pointing

The next_state function returns the following array of integer values:

{state, time, p, j}

3.11.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                               {slew + observation}
where tobs = tih tt tfh tend              {observation}
      tt   = tx (tpp tx)*                  {sequence of points}
      tx   = ((tp | ty) [thold])*          {one point}
      ty   = tz (tss tz)*                  {nods at one point}
      tz   = tp (tnod | tload) tp         {one nod}

```

3.11.4 Parameters

The CUS command custom_map has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	
zoffset	d	[-18000,18000]		arcsec	
yarray	d (*n)	[-7440, 7440]		arcsec	(1, 2)
zarray	d (*n)	[-7440, 7440]		arcsec	(1, 2)
tp	i (*n)	[10,50000]	1	s	(2)
pattnod	d	[0,360)		deg	
chopthrow	d	[2,960]	0.5	arcsec	
nmod	i	[1,1200]			
knod	i	[0,n]			
tloadmin	i	≥ 0	1	s	
nload	i	[0,nmod]			
thold	i	≥ 0			
nhold	i	[0,n]		s	

Table 24: Parameters for custom_map_pointing


Notes:

1. The parameters ra, dec, fixed, yoffset and zoffset define a Cartesian coordinate system. The arrays yarray and zarray, both of length 'n', specify a list of points with respect to this coordinate system.
2. The maximum length of the arrays is 3000.

3.11.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tnod, tss, tload, tend, n, tpp[1], tpp[2], ..., tpp[n-1]}

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	70

3.12 cross_scan_pointing

3.12.1 Description

A `cross_scan_pointing` is similar to a `line_scan_pointing`, but with scans in two directions that cross one another. When the number of cross scans is zero, the pattern is the same as `line_scan_pointing` and consequently this mode may be used as a replacement for `line_scan_pointing`.

The basic pattern consists of a normal line scan with 'n' lines, followed by 'xrepeat' (0 to 2) cross scans, each with 'nx' lines. This whole pattern is repeated 'nrepeat' times. When `xrepeat=2`, the second cross scan follows the path of the first in reverse to minimize slew time. Figures 18 and 19 illustrate the pattern with 1 and 2 cross scans, respectively.

In addition:

- In the nominal scan, a hold of duration 'thold' occurs at a fixed pointing for calibration after every 'nhold' lines. Similarly, in the cross-scan the hold occurs after every 'nholdx' lines. The hold count is set to zero when switching between the nominal scan and cross scan, or vice-versa, as the values of 'nhold' and 'nholdx' may differ.

If the nominal scan is denoted by N , the cross-scan by C and resetting of the hold count by 'r', the pattern is of the form:

$$\begin{array}{ll}
 r N_1 N_2 \dots N_{nrepeat} & (xrepeat=0) \\
 (rN rC)_0 (rN rC)_1 \dots (rN rC)_{nrepeat} & (xrepeat=1) \\
 (rN rCC)_0 (rN rCC)_1 \dots (rN rCC)_{nrepeat} & (xrepeat=2)
 \end{array}$$

The centre and orientation are described in the same way as a `basic_raster`, by the parameters 'ra', 'dec', 'fixed' and 'patt'. An additional parameter 'pattx' is used to specify the orientation of the cross scan map.

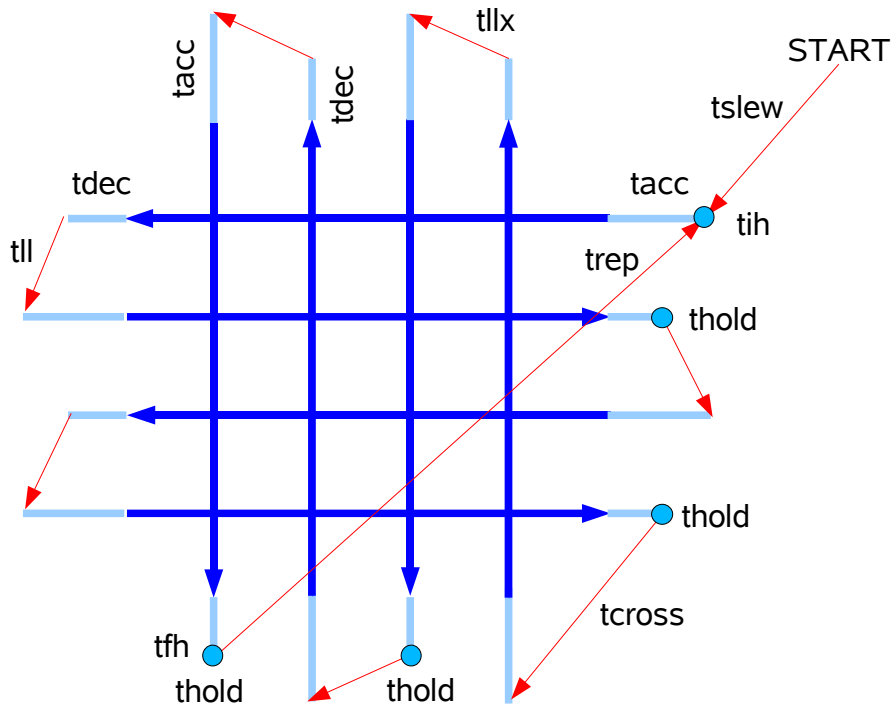


Figure 18: *Cross_scan_pointing*, $n=4$, $nx=4$, $xrepeat=1$, $nrepeat>1$

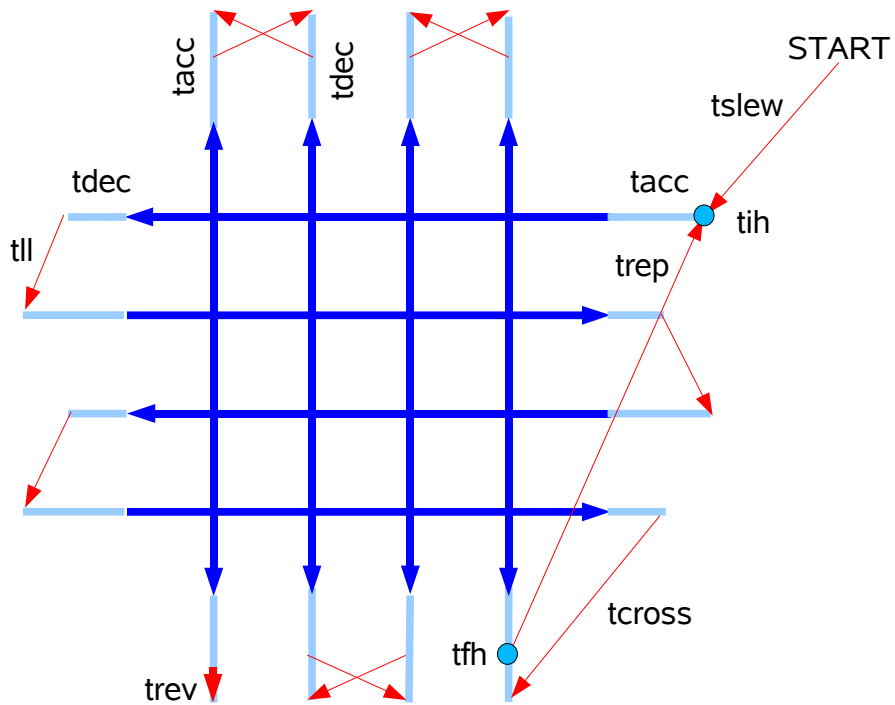


Figure 19: *Cross_scan_pointing*, $n=4$, $nx=4$, $xrepeat=2$, $nrepeat>1$

3.12.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	line=0, i=0, r=0, x=0
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih + tacc	LINE(1)	line=1, i=1, r=1
LINE(1)	nhold>0 & i%nhold=0	tl + tdec	HOLD(1)	
	line ≠ n	tl + tll	LINE(1)	line++, i++
	xrepeat ≠ 0	tl + tdec + tcross + tacc	LINE(2)	line=1, i=1, x++
	r ≠ nrepeat	tl + tdec + trep + tacc	LINE(1)	line=1, i++, r++
	else	tl + tdec	FINAL_HOLD	
LINE(2)	nholdx>0 & i%nholdx=0	tlx + tdec	HOLD(2)	
	line ≠ nx	tlx + tllx	LINE(2)	line++, i++
	x ≠ xrepeat	tlx + tdec + trev + tacc	LINE(2)	line=1, i++, x++
	r ≠ nrepeat	tlx + tdec + trep + tacc	LINE(1)	line=1, i=1, r++, x=0
	else	tlx + tdec	FINAL_HOLD	
HOLD(1)	line ≠ n	thold + tll - tdec	LINE(1)	line++, i++
	xrepeat ≠ 0	thold + tcross + tacc	LINE(2)	line=1, i=1, x++
	r ≠ nrepeat	thold + trep + tacc	LINE(1)	line=1, i++, r++
	else	thold	FINAL_HOLD	
HOLD(2)	line ≠ nx	thold + tllx - tdec	LINE(2)	line++, i++
	x ≠ xrepeat	thold + trev + tacc	LINE(2)	line=1, i++, x++
	r ≠ nrepeat	thold + trep + tacc	LINE(1)	line=1, i=1, r++, x=0
	else	thold	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 25: State table for cross_scan_pointing

The next_state function returns the following array of integer values:

{state, time, line, r, x}

3.12.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```

tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend                {observation}
      tt  = (tx trep)*                       {repetitions}
      tx  = tn [tcross tnx [trev tnx]]       {one repetition}

tn  = (ty tz)* ty                            {nominal map}
ty  = tacc tl tdec [thold]                   {nominal line}
tz  = (tll - tacc - tdec)                    {slew between nom. lines}

tnx = (tyx tzx)* tyx                        {cross map}
tyx = tacc tlx tdec [thold]                 {cross line}
tzx = (tllx - tacc - tdec)                  {slew between cross lines}

```


3.12.4 Parameters

The CUS command `cross_scan_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tslewmin	i	≥ 0	1	s	
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
ib	s				
naifid	i	0, use ra,dec >0, tracking			
ra	d	[0,360), naifid=0		deg	
dec	d	[-90,90], naifid=0		deg	
fixed	b				
patt	d	[0,360)		deg	
yoffset	d	[-18000,18000]		arcsec	
zoffset	d	[-18000,18000]		arcsec	
n	i	[1,1500]			1
d1	d	[20,72000], xrepeat=0 [20,14400], xrepeat>0	5 5	arcsec	
d2	d	0 or [2,480]	0.5	arcsec	
rate	d	[0.1,60]	0.1	arcsec/s	
thold	i	≥ 0	1	s	
nhold	i	[0,n]			
nrepeat	i	[1,100]			
xrepeat	i	[0,2]			
pattx	d	[0,360)		deg	
nx	i	[1,1500]			1
d1x	d	[20,14400]	5	arcsec	
d2x	d	0 or [2,480]	0.5	arcsec	
nholdx	i	[0,nx]			


Table 26: Parameters for cross_scan_pointing

Notes:

1. The map size is constrained by:

$$(n - 1) \times d2 \leq 14880$$


$$(nx - 1) \times d2x \leq 14880$$

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	74

3.12.5 Returned Values

The following array of times, in seconds, is returned:

{tobs, tslew, tacc, tdec, tl, tll, trep, tlx, tllx, tcross, trev, tend}

 HSC <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	75

3.13 no_pointing

3.13.1 Description

The `no_pointing` is used for engineering 'observations', such as cooler recycling, where no specific pointing direction is required.

3.13.2 State Table

State	Condition	Duration	Next state	Action
START			SLEW	
SLEW		tslew	INIT_HOLD	
INIT_HOLD		tih	POINT	
POINT		tp	FINAL_HOLD	
FINAL_HOLD		tfh + tend	END	

Table 27: State table for no_pointing

The `next_state` function returns the following array of integer values:

{state, time}

3.13.3 Timing Syntax

The timing of the pattern is described by the following syntax:

```
tslew tobs                                {slew + observation}
where tobs = tih tt tfh tend              {observation}
      tt    = tp                            {dummy pointing}
```

The parameters 'tih' and 'tfh' allow additional hold times to be introduced at the beginning and end, respectively. These parameters are not strictly necessary as the same effect could be achieved by extending the time 'tp'. However, they are included for consistency with the other modes.

3.13.4 Parameters


The CUS command `no_pointing` has the following parameters:

Parameter	Type	Range	Resolution	Units	Notes
execute	b				
tih	i	≥ 0	1	s	
tfh	i	≥ 0	1	s	
tp	i	[1,50000]	1	s	

Table 6: Parameters for basic_fine_pointing

3.13.5 Returned Values

The following array of times, in seconds, is returned:

 <i>HSC</i> <i>Development</i>	Doc. No	HERSCHEL-HSC-DOC-624
	Issue	2.4
	Date	10.09.10
	Page	76

{tobs, tslew, tend}

The value `tslew` is included for consistency with the other modes, although it is zero.