EURO 3D

Promoting 3D spectroscopy in Europe

www.aip.de/Euro3D

RESEARCH TRAINING NETWORK

Sponsored by the EUROPEAN COMMISSION

Euro3D Data Format

—

Format Definition

Issue 1.2

May 15th, 2003

Authors . . . M.Kissler-Patig, Y.Copin, P.Ferruit, A.Pécontal-Rousset, M.M.Roth

Contact . . . M.Kissler-Patig (mkissler@eso.org)

This page was intentionally left blank

Special Acknowledgements

The definition of the format was developped within the frame of the "3D Spectroscopy Working Group" supported by OPTICON (The Optical Infrared Coordination Network). OPTICON is a project funded by the European Commission as part of its Fifth Framework Programme. OPTICON brings together providers and users of European astronomical infrastructures to identify common approaches and improve access for the benefit of all European astronomy.

The members of the "3D Spectroscopy Working Group" and of the Euro3D Research Training Network are very grateful to OPTICON and the European Commission for supporting their early efforts that helped preparing for the Euro3D Research Training Network.

This page was intentionally left blank

Change Record

Issue	Date	Section affected	Reason/Initiation/Documents/Remarks
0.1	Dec 2001	all	first draft issue, Milano Euro3D meeting
0.2	Jan 2002	all	second draft issue, Marseille Euro3D meeting
0.3	May 14, 2002	all, added Group and Compound table	third draft issue
0.4	June 4, 2002	all	fourth draft issue, base for pratical tests
0.5	June 17, 2002	sect 3.1,3.5	minor corrections to v0.4 after software library decision
1.0	Oct 15, 2002		first release, examples updated from v0.5
1.1	Dec 15, 2002	all	Total restructuring of the format in order to pack the same information into a more flexible binary table, rather than into extensions. Information and philosophy remain unchanged.
1.2	May 15, 2003	all	updates on 1.1 with input from i/o library coding

This page was intentionally left blank

Contents

1 Introduction

1.1 Purpose and Scope

1.1.1 Purpose

This document defines a data format for Integral Field Spectroscopy. The format was proposed and introduced by the OPTICON 3D Spectroscopy Working Group and implemented within the Euro3D Research Training Network.

The general layout of the Data Format was discussed and endorsed by the working group meetings in Lyon (25-26.6.2001), Milano (3-4.12.2001) and Marseille (20-21.03.2002), a restructuring of the information into binary tables occurred on kind recommendation of F.Valdes and P.Grosbøl.

1.1.2 Scope

The goal of the Euro3D research training network is to convert integral field spectroscopy from a technique reserved for experts into a powerful common user tool. To achieve this goal, the objectives are twofold. On the one hand, we need to develop common, easy-to-use tools that will allow the data from 3D instruments to be exploited. On the other hand, these data need to be used to conduct forefront scientific research, in order to attract users.

In order to introduce new, young researchers into using this quickly growing, yet relatively new technique, *a common format for the data of all 3D instruments is clearly the most important milestone.* Such a format, satisfying the requirements of all integral field instruments, was not available and had thus to be agreed upon.

The OPTICON 3D spectroscopy working group debated the nature of this format and decided to introduce a format based on 2D images and associated tables. The 2D images would be used to store the spectra (one spectrum per row) while the tables would serve to store additional information on the spectra (e.g. their position). The arguments for this choice are summarized in the minutes of the Euro3D meeting held in Lyon on June 25-26, 2001 (available through the Euro3D web pages). Briefly, the choice was made between a pure 3D cube, a 2D image+table and a 2D image+cube. The condition that no information shall be lost (i.e. resampling shall not be mandatory) excluded the pure 3D cube, given the non-square pixels of several instruments. Between a 2D image+table and 2D image+cube, the 2D image+table was considered the more flexible for irregular patterns, mosaicing etc... especially in view of the much larger IFUs to come.

The first practical implementation of the Euro 3D data format followed strictly this idea and the Euro 3D fits files contained image extensions for the spectra and table extensions. However, in order to solve various practical issues (flexibility, data access times for large data volumes) and following the advises of F. Valdes and P. Grosbøl, it was decided to to store the spectra as vectors in columns of binary tables instead of stacking theme into an image. Therefore, in the current implementation of the Euro3D format, both the data and the information on the spectra are stored in a single FITS binary table.

The Euro3D format is not originally thought as the format used within an instrument, but rather the format in which the data are stored and circulated once the instrumental signature has been removed. Its main aim is to facilitate the data exchange and inter-instrument data combination of

calibrated data, as well as to allow to share analysis software.

The present document defines the adopted Euro3D FITS format.

1.2 Reference documents

[1]	A User's Guide to the Flexible Image Transport System (FITS)	issue 4.0, 14.04.97
[2]	Definition of the Flexible Image Transport System (FITS)	Standard NOST 100-2.0, 19.03.99
[3]	Representations of world coordinates in FITS	Greisen & Calabretta, 2002, A&A 395, 1061
[4]	Representations of celestial coordinates in FITS	Calabretta & Greisen, 2002, A&A 395, 1077
[5]	Report on the Meeting of the OPTICON 3D Spectroscopy Working Group held in Lyon, June 25-26, 2001	J.Walsh, July 3, 2001
[6]	Binary table extension to FITS	Cotton, Tody & Pence, 1995, A&AS 113, 159

Documents [1] and [2] are available at `http://heasarc.gsfc.nasa.gov/docs/heasarc/fits.html`.
Documents [3] and [4] can be downloaded from `http://fits.gsfc.nasa.gov/documents.html#WCS`.
Document [5] can be found on `http://www.aip.de/Euro3D`.

1.3 Abbreviations and acronyms

AIP	Astrophysikalisches Institut Postdam
DIT	Detector Integration Time
DQ	Data Quality
ESO	European Southern Observatory
FITS	Flexible Image Transport System
FWHM	Full Width at Half Maximum
HDU	Header and Data Unit
IFS	Integral Field Spectroscopy
IFU	Integral Field Unit
MPE	Max-Planck Institut für extraterrestrische Physik
NIR	Near Infra-Red
NOST	NASA / Science Office of Standards and Technology
PA	Position Angle
PSF	Point Spread Function
RD	Reference Document
RON	Read-Out Noise
RMS	Root Mean Square
SPAXEL	SPAtial piXture ELement
S/N	Signal-to-Noise ratio
TBD	To Be Determined
WCS	World Coordinate System

2 Overview of the Euro3D format

The Euro3D format is defined as a FITS file including several extensions, all of them are binary FITS tables. The most important extension is the Data table, that contains in each row the information for a single spectrum, including the position information, information on binning, the spectrum itself as well as the corresponding data quality information and (optionally) statistical information. In addition, an extension describing the shape of the genuine spatial elements on sky must be attached (the group table). Further tables containing results from the scientific analysis can be attached.

The proposed format for the FITS file is the following:

Content	Extension
Primary header (with empty primary data array)	[0]
Data table (containing position and spectral information)	[1]
Group table (defining the Spaxel shapes)	[2]
Science table (1)	[3] Optional
Science table (2)	[4] Optional
...	...

Note: FITS does not support a preferred order of the extensions. Accordingly, the extensions in the Euro3D format are not identified by their position in the file, but by their name (FITS keyword: `EXTNAME`) and could in principle appear in any order within the file. The above order is just the suggested one.

Note: We refer to *SPAXEL* or *spaxel* as the SPAtial piXture ELement of the instrument (e.g. fibre, lens, fraction of a slicer, ...), in order to avoid confusion with the detector pixels. Accordingly, we use the term *compound spaxel* for a 'binned' spaxel composed of a number of genuine instrument spaxels. Note, however, that the binning does not need to be regular, thus the use of the term *compound* rather than binned.

3 The Euro3D FITS file

In the following sections, we describe in more detail the different extensions of the Euro3D format.

The Euro3D format strictly follows the standard FITS convention. As such, no subsequent rule defining the Euro3D format shall violate the FITS convention as defined in [1],[2].

3.1 Primary header

3.1.1 The basic FITS header

The primary header, following the FITS standard, *must contain and in this order* the following keywords:

```
SIMPLE
BITPIX
NAXIS
...
END
```

In the Euro3D format, the following primary data array shall be empty, i.e. `NAXIS`, the dimension of the associated data array set to 0.

3.1.2 Announcing (the Euro3D) extensions

In addition to these keywords, the existence of extensions in the Euro3D format must be 'announced' (following the FITS standard) by adding *immediately behind* the last `NAXIS` keyword the keyword:

```
EXTEND = T
```

In order to identify a Euro3D file, a FITS file written in the Euro3D format should contain (anywhere between `EXTEND` and `END`) the following keyword set to logical value True:

```
EURO3D = T
```

This last keyword shall be used if *all mandatory* extensions are present in the file and correspond to the Euro3D format.

3.1.3 Special Euro3D keywords in the primary header

A number of special keywords are used in the Euro3D format. They must be included in the primary header.

• If the data have been corrected for atmospheric refraction, the keyword `E3D_ADC` must be present in the primary header and set to true. This may be applicable if an instrument incorporates an atmospheric dispersion corrector, if the atmospheric refraction was corrected for during the data

reduction, or simply if the effect is negligible (e.g. in the NIR). If none of the latter is the case, i.e. if a correction is needed the keyword must be set to *false*, `E3D_ADC = F`. In this case, further information is required in the group table (see sect. 5).

`E3D_ADC = F or T` : false if the data have *not* been corrected for atmospheric dispersion

• The version number of the format `E3D_VERS` should be recorded in the header. This is to allow later software switches in the data reduction and analysis software, in case significant modifications to the format are performed.

`E3D_VERS= [STRING]` : version number of the format

3.1.4 Summary of mandatory information in the Euro3D primary header

In summary, the following keywords must be present (in this order for the first four, and with these values for the first five) to comply with the Euro3D format. The primary data array must be empty, which is set by `NAXIS = 0`.

```
SIMPLE  =                    T / file conforms to FITS standard
BITPIX  =                    8 / number of bits per pixel
NAXIS   =                    0 / number of image axes
EXTEND  =                    T / file contains extensions
...
EURO3D  =                    T / file conforms with Euro3D standard
...
E3D_ADC =                    T / data corrected for atm dispersion
E3D_VERS=                  1.0 / version number of the Euro3D format
...
END
```

3.1.5 Standard FITS keywords in the Euro3D primary header

Beside the above keywords, the user is encouraged to add the following standard FITS keywords (listed and explained at `http://heasarc.gsfc.nasa.gov/docs/fcg/standard_dict.html`)

```
AUTHOR  : name of the person that created this file
COMMENT : columns 9-80 may contain any ASCII text for comments
DATE    : date on which this file was created
DATE-OBS: date of the observation
EPOCH   : epoch in years for the system in which the position is expressed
EQUINOX : equinox in years for the system in which the position is expressed
HISTORY : columns 9-80 contain any ASCII text describing the history of the file
INSTRUME: instrument used to acquire the data
OBJECT  : name of the object observed
OBSERVER: name of the person who acquired the data
ORIGIN  : name of the institution responsible for creating this file
REFERENC: reference where the data associated with the header are published
TELESCOP: name of the telescope used to acquire the data
```

For DATE and DATE-OBS the FITS format (from 2000 on) is 'yyyy-mm-dd' or 'yyyy-mm-ddTHH:MM:SS[.sss]'.

In addition to these, the user can add any number of keywords complying with the FITS standard.

3.1.6 Example of a Euro3D primary header

Note: this examples includes standard/usual FITS comments after the / sign.

A typical primary header is shown in the following example which will accompany us through this document:

Luke Skywalker observed the Death Star using the 'Ultimate 3D Spectrograph' on the OWL telescope owned by the Jedi Council. The data comply (of course) with the Euro3D format. Obi-Wan Kenobi did the data reduction and analysis (using the Power-auto-reduction software Euro3D), and published the results in the Jedi Journal of Science (Kenobi et al. 2011, JJoS 3475, 237).

```
SIMPLE  =                    T / file conforms to FITS standard
BITPIX  =                    8 / number of bits per pixel
NAXIS   =                    0 / number of axes
EXTEND  =                    T / file contains extensions
EURO3D  =                    T / file conforms with Euro3D standard
E3D_ADC =                    T / data corrected for atm dispersion
E3D_VERS=                  1.0/ version number of the Euro3D format
AUTHOR  ='Kenobi, Obi-Wan (Ben)'/ author of the data
COMMENT = 'Use the Force, Luke!'/descriptive comment
DATE    =           '2010-06-01'/ date of file creation
DATE-OBS=           '2010-04-23'/ date of the observation
EPOCH   =                  2000/ equinox of celestial coordinate system
EQUINOX =                  2000/ equinox of celestial coordinate system
HISTORY = 'Power-auto-reduced with Euro3D' / processing history of the data
INSTRUME=     'Ultimate 3D Spec'/ name of instrument
OBJECT  =           'Death Star'/ name of the observed object
OBSERVER=       'Luke Skywalker'/ observer who acquired the data
ORIGIN  =         'Jedi Council'/ organization responsible for the data
REFERENC='Jedi Journal of Science'/ bibliographic reference
TELESCOP=                 'OWL'/ name of telescope
...
END
```

3.2 Extensions

This section gives some general remarks on the extensions, which are then discussed in more details in the following sections.

The Euro3D format includes at least two extensions in addition to the primary header (see Sect. 2):

- the Data table
- and the Group table.
- Further, it may include Science tables.

All extensions are binary FITS tables.

Following the FITS standard, each extension must include the following keywords (similar to the primary header but replacing `SIMPLE` by `XTENSION`):

```
XTENSION
BITPIX
NAXIS
NAXISn
...
PCOUNT
GCOUNT
...
END
```

The keywords `PCOUNT` and `GCOUNT` specify the size of the extension. For binary tables, `GCOUNT` is 1 and the total number of bytes is PCOUNT + NAXIS1×NAXIS2. The value of `PCOUNT` is equal to the number of bytes in the binary table extension `following` the main table (i.e. usually data are stored in the table only, PCOUNT=0; but in certain cases data can follow the table within the extension in which case PCOUNT>0). When no table entries of variable length are defined (as it is the case in the Euro3D format), `PCOUNT` will in general be equal to 0.

In addition, the following keywords are reserved for extension headers and should be used as described in the sections below:

`EXTNAME` = <`STRING`>, gives a unique name to the extension
`EXTVER` = 1 (default), gives the version of the extension
`EXTLEVEL` = 1 (default), gives the level of the extension

Note that extensions can appear in any order within the FITS file, after the primary header. As such, no order is imposed but we recommend to use the order proposed in Sect. 2. We discuss the extensions in turn.

4 The Data extension

This extension is **mandatory**.

This extension contains all positional and spectral information within a binary table. Remember that binary tables accept vectors and arrays as table element entries. The Euro3D format makes heavy use of this option.

The Data table contains 1 row for each spectrum. In this row, the first entries serve to define an ID, the corresponding position on the sky, and eventual binning. The next entries in the row contain the spectrum (as 1 dimensional vector in wavelength), a data quality (DQ) spectrum (1D vector), and optionally a statistical error spectrum (1D vector).

Some important notes:

- All columns hosting spectra (and that means *ALL* incl. data, data quality and statistical error spectra in all rows) have the same length in number of vector elements. However, the actual useful spectra within these columns can be shorter (i.e. use only a fraction of the column length).
- All spectra share the same wavelength unit (CTYPES), reference value (CRVALS) and step (vector element) size (CDELTS) defined as keywords in the spectra. Note, however, that no CRPIXS is **not** defined, i.e. the spectra do not need to all start with the same wavelength, as long as they start with an integer number of steps (CDELTS) from the reference value (CRVALS). This integer number of steps in given in column 5 (SPEC_STA).
- For a given spectrum ID (column 1), the data, the data quality and the statistical error (if present) spectra **must** start with the same wavelength and have an identical number of useful vector elements.
- All data spectra and statistical error spectra share the same pixels (vector element) units, defined as in the keyword CUNITS in the header.

4.1 The structure of the table

The Data extension is a FITS binary table which, for each spectrum (i.e. row) contains 11 or 12 entries (Note that column 12 is optional even if strongly recommended), namely:

- Column 1: SPEC_ID, a spectrum identifier [long integer].
- Column 2: SELECTED, a selection flag [logical]
- Column 3: NSPAX, the number of instrument spaxels composing the spectrum [integer]
- Column 4: SPEC_LEN, the length in vector elements of the useful spectrum [positive long integer]
- Column 5: SPEC_STA, defines the starting wavelength (wavelength of the first vector element) by an integer number of steps (CDELTS) from the reference value (CRVALS) [long integer]

- Column 6: XPOS, the spaxel coordinate(s) along spatial axis 1 (a vector when the data are spatially binned) [(vector of) doubles]
- Column 7: YPOS, the spaxel coordinate(s) along spatial axis 2 (a vector when the data are spatially binned) [(vector of) doubles]
- Column 8: GROUP_N, the group number [(vector of) long integer]
- Column 9: SPAX_ID, a spaxel identifier [character string] (a vector of ASCII strings when the data are spatially binned)
- Column 10: DATA_SPE, the data spectrum [vector of short/integers/long integers/floats/doubles]
- Column 11: QUAL_SPE, the associated data quality spectrum [vector of long integers]
- Column 12 (optional): STAT_SPE, the associated statistical error spectrum [type must be identical to the one of column 10]

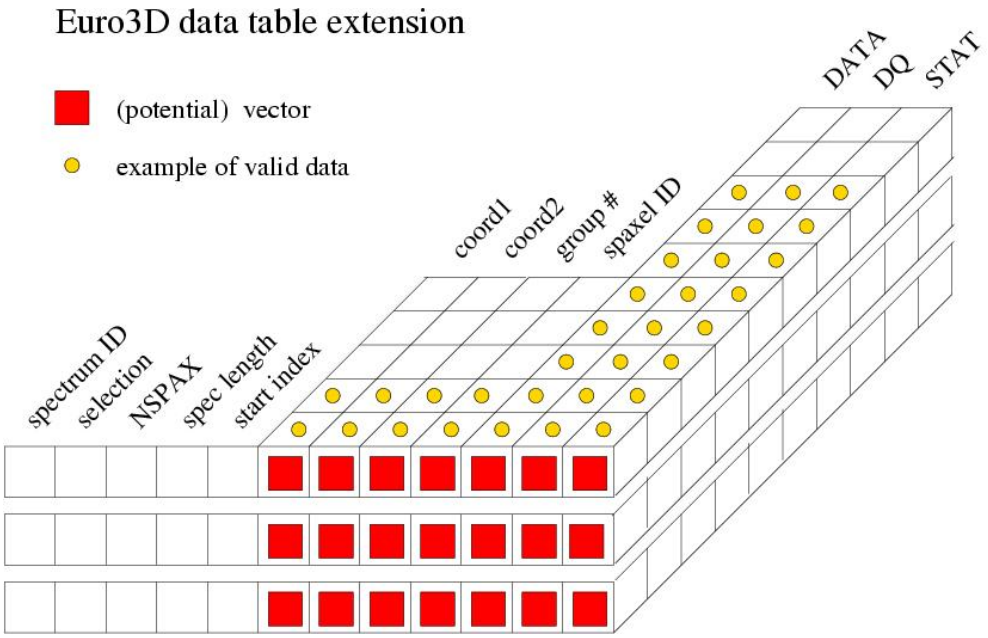

Figure 1: Schematic overview of the data extension. Reach row describes one (1) spectrum. Note that column 12 is optional. The details on each column are given below. The squares mark columns that can be vectors, in which case columns 6,7,8,9 have the same dimension, as well as columns 10,11,12. Filled dots mark an example of valid data for the first spectrum and illustrate that not all vector elements need to contain valid data.

Each column is described briefly below and in more details in the next sections.

- Column 1: the spectrum ID is used to associate an (long) integer value to each spectrum in the file. It must be unique in the data table. It does not need to start with a given value nor to have increments of 1, nor to be sorted in the table (although all these would be good practice). It is **the only** reference for the science tables in order to associate the science information with the right spectra and shall therefore be treated rigorously.

- Column 2: the selection flag (as a logical value) allows to store a selection performed during the data analysis directly into the file. It has proven to be very useful to the Lyon group in the past during the data analysis and was thus included in the format. The flag is set, i.e. the spectrum selected, on TRUE. By default, all spectra should be selected, i.e. all rows in column 2 set to TRUE.

- Column 3: `Nspax` is the number of *genuine* spaxels (real instrumental spatial resolution elements) from which the spectrum in this row was obtained, i.e. the number of instrument spaxels that compose this spaxel. Its value will be 1 when the spectrum in this row was obtained from 1 spatial resolution element of the instrument (typically when the data are unbinned), or alternatively when the user does not want to carry along the information on individual spaxels at the origin of the spectrum. `Nspax` is greater than 1 when several spaxels got combined to a compound spaxel, i.e. when spectra of several spatial resolution elements were combined, or the spectra were spatially binned. Then `Nspax`> 1 allows to carry the information on the individual spaxels at the origin of the spectrum.

  The maximum `Nspax` in the table defines the dimension of columns 6,7,8,9. For `Nspax`> 1, these columns host then vectors as explained below. The value in column 3 defines the number of useful elements in the columns 6,7,8,9.

- Columns 4 and 5: In order to keep the format compact and the access on disk fast, it was decided to avoid heaps (data physically stored below the table when a column contains elements of different length, see [1]). This assumes that in most cases all spectra have roughly the same length in terms of spectral resolution elements. Thus, the dimension of columns 10,11,12 is fixed for all rows and corresponds to the length of the longest spectra (in terms of resolution elements). All other spectra will fill all or only part of the columns 10,11,12.

  For a given spectrum in row SPEC_ID, the wavelength associated with the n^{th} element (where $1 \leq n \leq \mathrm{SPE\_LEN}_{\mathrm{SPEC\_ID}}$) in column 10, 11 or 12 is:

$$\lambda^{n}_{\mathrm{SPEC\_ID}} = \mathrm{CRVALS} + ((\mathrm{SPE\_STA}_{\mathrm{SPEC\_ID}} + (n-1)) \times \mathrm{CDELTS})$$

  Let's assume that your file includes 100 spectra of different length, the longest has 300 resolution elements. In that case, the columns 10,11,12 will be dimensioned to host a vector of 300 elements. And the longest spectrum will use the full length of the column and have an entry in column 4, SPE_LEN, of 300. Another spectrum that covers only $2/3$ of that wavelength range (has a length of 200 elements) will only fill the first 200 elements in columns 100,11,12. Its 'useful' length, the entry in column 4, SPE_LEN will be 200. It fill always fill the first `n` elements of columns 10,11,12. So how will we know the wavelength of its first element that can vary from spectrum to spectrum? To do this, we use column 5, defining the starting index, SPE_STA.

  In the header, there is defined a reference value (CRVALS) and a step size (CDELTS). Column 5 gives the (integer) number of steps away from the reference value at which the spectra in columns 10,11,12 start.

  For example, let's assume that CTYPES is set to 'nanometers', CRVALS to 550 (nm), and CDELTS to 0.5 (nm). The longest spectrum spans 300 elements (steps), i.e. 150nm. For that spectrum, the value in column 5 is 0, i.e. it starts 0 steps from the reference values, i.e. at the reference value 550nm, and thus covers 550nm to 700nm. The other spectrum spans the range 570nm to 670nm (indeed 100nm that is 200 steps, remember the step size is fixed for all spectra). It starts 20nm, i.e. 40 steps above the reference values, its entry in

column 5 is (+)40. The first 200 entries in columns 10,11,12 are useful, the last 100 will be ignored (set in the data quality spectrum to NaN). To extend the example, another spectrum spanning 450nm to 580nm will have the entries 260 in column 4 (length is 130nm / (0.5 nm/steps)= 260 steps), and -200 in column 5 (starting at the reference 550nm - (200×0.5 nm/step)).

- Column 6 and 7: this column gives the coordinates along the two axis of the (list of) spaxel(s) to which the spectrum is associated. Typically, `Nspax`$= 1$ and it will contain only one value in each column (e.g. X in column 6 and Y in column 7). However, if the spaxel is a compound one (`Nspax`> 1), the two columns will each list the X and Y coordinates of a number `Nspax` of genuine spaxels, i.e. the coordinates of all individual spaxels that were used to produce this one spectrum are listed.

  For example, for 2×2 spatially binned spaxels, the coordinates of the compound spaxel will be defined by 4 pairs of coordinates (X,Y)s, and column 6 and 7 will each contain a vector of dimension 4 (i.e. in column 6 the 4 Xs, in column 7 the 4 Ys).

- Column 8: the group number identifies to which group the spaxel(s) belong. The group information is then to be taken from the group table (see Sect. 5). Again, in the case of compound spaxels, the same applies as for the coordinates: the column contains a *list* of `Nspax` group numbers.

- Column 9: the spaxel ID allows each genuine spaxel to carry an ID as ASCII character string(s). This allows to carry along a spaxel ID coming directly from the instrument, and to identify the physical origin of each spaxel. And again, in the case of compound spaxels, the same applies as for the coordinates: the column contains a *list* of `Nspax` spaxel IDs.

- Column 10: contains the data spectrum as a vector. It can contain a vector of short/long integers, floats or doubles. The wavelength unit (CTYPES), the unit of the vector element (CUNITS) and fixed increment between each vector element (CDELTS) are defined as header keywords. The dimension of the column is set as the number of spectral resolution elements of the longest spectrum in the file. The wavelength of the first element and number of valid elements for a given spectrum are given in columns 4 and 5 (see above).

- Column 11: contains as a vector the associated data quality spectrum. The data quality spectrum allows to attach a specific data quality code/flag (long integer, see Sect. 4.6) to each pixel of the data spectrum. The wavelength of the first element and number of valid elements are the same as for the data spectrum.

- Column 12 (optional): contains as a vector the associated statistical error spectrum. The statistical error spectrum allows to carry along the different reduction steps the statistical error attached to each pixel in the data spectrum. The units of the vector elements (CUNITS), as well as the wavelength of the first element and number of valid elements are the same as for the data spectrum.

4.1.1 Mandatory header keywords for the Data extension

As all columns will be described in more detail in the following sections, we will progressively built up a full header for the Data table extension. Here we start with the main header.

The following keywords (and values when indicated) are mandatory at the beginning of the Data table extension header:

`XTENSION = 'BINTABLE'` : type of extension (binary table)
`BITPIX = 8` : binary table
`NAXIS = 2` : table with rows and columns
`NAXIS1 = [INTEGER]`: number of bytes per row
`NAXIS2 = [INTEGER]`: number of rows (equal to the number of spectra)
`PCOUNT = [INTEGER]`: number of bytes in the extension after the binary table
`GCOUNT = 1`: always 1 for binary tables
`TFIELDS = 11 or 12` : number of columns
`EXTNAME = E3D_DATA` : extension name for Euro3D position table
...
here follow all the column definitions, etc (see next sections)
...
`END`

4.1.2 Beginning the example of a data table header

The are starting here the example of a data table header. The example will be continued in the next sections as we describe the content of the data table. The full header is given in the Appendix.

Recall that Luke Skywalker observed the Death Star using the 'Ultimate 3D Spectrograph' on the OWL telescope owned by the Jedi Council. The data comply (of course) with the Euro3D format. Obi-Wan Kenobi did the data reduction and analysis (using the Power-auto-reduction software Euro3D), and published the results in the Jedi Journal of Science (Kenobi et al. 2011, JJoS 3475, 237).

Luke's 'Ultimate 3D Spectrograph' has 64×64 spaxels, each spectrum occupying 2048 detector pixels. He observed the Death Star in the optical light (4197.80 Å$< \lambda <$9604.21 Å).

The corresponding header of the data table of Luke's 'Ultimate 3D Spectrograph' exposure looks the following:

```
XTENSION=             BINTABLE / type of extension (binary table)
BITPIX  =                    8 / must be 8 for a binary table
NAXIS   =                    2 / table with rows and columns
NAXIS1  =                32784 / number of bytes per row 16+(8+4+4)x2048
NAXIS2  =                 4096 / number of rows (spectra) here 64x64
PCOUNT  =                    0 / no data in the extension after the binary table
GCOUNT  =                    1 / always 1 for binary tables
TFIELDS =                   12 / 12 and only 12 columns in the data table
EXTNAME =             E3D_DATA / extension name for Euro3D position table
...
```

4.2 The IDs and flags

4.2.1 The spectrum ID

Column 1 defines an ID independently of the spectrum position in the spectra extension. It must be unique in the data table.

This ID is used as reference for the science tables in order to associate the science information with the right spectra. It can further be used to carry along the ID from the raw data while working on a subset of processed spectra, etc... It identifies the spectrum as opposed to the ID in column 9 that identify the spaxel(s) from which the spectrum originated.

4.2.2 The selection Flag

Column 2 is used by the Euro3D software to store a selection together with the data. It consist of a logical value (set to True if that particular spectrum is selected, False if not). By default it shall initially be set to True for all spectra.

4.2.3 The spaxel ID

Column 9, the spaxel ID allows to carry along a user ID, independently of the ID in column 1. While the ID in column 1 is associated with the spectrum in that row, the ID in column 9 is used to identify the spaxel itself. Thus, when a compound spaxel is made out of several genuine spaxels, the ID is allowed to be a vector carrying IDs for all genuine spaxels composing the new compound spaxel.

4.2.4 Mandatory header keywords for IDs and Flags

The following header keywords are mandatory to define these columns in the header of the extension.

```
...
TFORM1 = 1J : field 1 contains one signed 32-bit integer
TTYPE1 = 'SPEC_ID ': spectrum ID
TFORM2 = 1L : field 2 contains one logical value
TTYPE2 = 'SELECTED ': Selection flag
...
TFORM9 = rA : field 9 contains one 8-bit ASCII character string
TTYPE9 = 'SPAX_ID': identifier for each spaxel
...
```

where r is fixed for the entire table and is equal to the largest Nspax in the table. Nspax defines the number of genuine spaxels that compose the compound spaxel to which the data spectrum is associated. In the simplest case, all spectra will come from 1 instrument resolution element only (i.e. 1 spaxel and Nspax= 1) and only one 8-bit ASCII ID will be present in column 9 (TFORM9=1A). However, if one or more spectra are associated with a compound spaxel, and the largest compound is made of e.g. 12 spaxels (Nspax=12, i.e. each columns 6,7,8,9 have vectors of dimension 12), then r=12 and the column 9 entry will host a vector for twelve 8-bit ASCII strings (TFORM9=12A).

4.2.5 Example of the data table header continued

Luke's 'Ultimate 3D Spectrograph' just complies will all the above and the spectra are all from 1 spaxel (no binning applied yet), so that NSPAX$= 1$ *throughout the table.*

```
...
TFORM1  =                    1J / spectrum ID expressed a one signed 32-bit integer
TTYPE1  =              SPEC_ID / spectrum ID
TFORM2  =                    1L / selection flag is a logical value (1bit)
TTYPE2  =             SELECTED / selection flag\\
...
TFORM9  =                    1A / NSPAX=1 thus the entry is only one 8-bit ASCII char strin
TTYPE9  =              SPAX_ID / user ID for each spaxel
...
```

4.3 The position information

The position information is given by four columns (3, 6, 7, and 8).

• Column 3 is used to identify compound (binned) data. It gives the number of *genuine* spaxels that were used to compose the spectrum of this spaxel. In the simplest case the described spaxel was obtained from a single instrument spaxel, and this number will be 1. However, if the spectrum in that row was obtain from several instrumental spaxels, it gives the number of spaxels that were combined in order to obtain the spectrum (e.g. a 2×2 spatial binning will have the number 4 associated with it, or e.g. a spectrum obtained from 12 instrument spaxels will have the number 12). **If column 3 hosts values > 1 the entries in column 6,7,8,9 will be vectors**, otherwise, if NSPAX$= 1$, columns 6,7,8,9 each host a single 64-bit float.

The cases of Nspax> 1 are discussed in more detail in Sect. 4.4 below.

Note that the Euro3D definition of compounds, unlike the familiar understanding of regular binning (e.g. of CCD pixels), allows one to create any spatially heterogeneous combinations of spaxels. Compound spaxels allow adaptive binning, synthetic apertures, creating annuli for sky subtraction, ... (see for further discussion Sect. 4.4).

• Columns 6 and 7 give the position of the associated spectrum in instrumental coordinates. If in column 3 Nspax> 1, the values of column 6 and 7 are vectors.

The coordinates refer to the geometric center of the (genuine) spaxel(s). The instrumental coordinates are being converted into the world coordinate system as described below (see Sect. 4.3.2).

The exact shape of the spaxels is defined in the Group table (see Sect. 5).

• Column 8 allows to group in the same file spectra from different exposures (e.g. mosaic) performed with the same IFU, or even merge spectra from different instruments or IFUs. Column 8 contains a group index (1,...,*n*) which identifies *n* groups, each of spectra belonging to the same IFU/exposure. **Note**, however, that in the case of multiple groups *i)* the wavelength sampling, units, reference value and vector element units must be the same for all groups (only 1 set of CDELTS, CUNITS, CRVALS, CTYPES keywords is allowed in the header) and *ii)* that spatial re-sampling will be required to bring all groups into the same reference system since only one WCS is allowed.

Note further that in the case of compound spaxels, column 8 will be a vector (of same dimension as columns 6 and 7) with one group number per genuine spaxel composing the compound spaxel (see Sect. 4.4).

Keeping track of the group identification is used to allow a visualization tool or data analysis software to distinguish between different deployable IFUs, or to separate the main IFU spectra from special purpose spectra (e.g. calibration spectra).

4.3.1 Mandatory header keywords for the position information

The following keywords (and values when indicated) are mandatory for the column definitions:

...
TFORM3 = 1J : field 3 contains one signed 32-bit integer
TTYPE3 = 'NSPAX ': number of genuine spaxel composing this spaxel
TFORM6 = rD : field 6 contains a number r of 64-bit floats
TTYPE6 = 'XPOS ': spatial element X position
TUNIT6 = [STRING] : units of the X coordinates
TFORM7 = rD : field 7 contains a number r of 64-bit floats
TTYPE7 = 'YPOS ': spatial element Y position
TUNIT7 = [STRING] : units of the Y coordinates
TFORM8 = 1J : field 8 contains one signed 32-bit integer
TTYPE8 = 'GROUP_N ': IFU group number (see also group table extension)
...

Note: If each spectrum originated from one genuine instrument spaxel (no binning occurred) then all values of column 6 are set to N_group$= 1$.

4.3.2 The World Coordinate System

The instrument coordinates in columns 6 and 7 get transformed into the world coordinate system using the information stored in header keywords. These keywords are following the FITS convention as described in [3]. In that particular paper, our case is treated in their Sect. 3.2, as the two spatial dimensions (that we will refer to as x and y) are not stored as different dimensions in the extension but as table columns, that can be vectors.

Note that only one set of keywords is allowed in the headers, i.e. that all coordinates must share the same reference system.

4.3.3 Mandatory header keywords for WCS transformation

For the transformation into world coordinate system (WCS), the following keywords must be specified in the data table header:

...
TCTYP*n* = [STRING] : axis type
TCUNI*n* = [STRING] : axis unit
TCRVL*n* = [FLOAT] : reference value
TCDLT*n* = [FLOAT] : coordinate increment
TCRPX*n* = [FLOAT] : reference point
...
TP*n_k* = [FLOAT] : transformation matrix
TC*n_k* = [FLOAT] : transformation matrix
TV*n_m* = [FLOAT] : coordinate parameter
TS*n_m* = [FLOAT] : coordinate parameter
TWCS*n* = [FLOAT] : coordinate name
TCRD*n* = [FLOAT] : random error
TCSY*n* = [FLOAT] : systematic error
TCROT*n* = [FLOAT] : coordinate rotation
...
END

Note: For the transformation into WCS, *n* and *k* denote the column number (6 or 7 for the two spatial axes), *m* is an integer between 0 and 99. For further details on the transformation we refer to reference [3].

4.3.4 Atmospheric Dispersion

The Euro3D format allows to stored information on atmospheric dispersion in the group table (see Sect. 5). This information shall be used to correct the coordinates prior to transform them into the WCS.

In the most general case the data will be affected *i)* by atmospheric dispersion in the sense that a position is only valid for a given wavelength (e.g. monochromatic images in a data cube will be shifted spatially over a large wavelength range); and *ii)* by differential atmospheric refraction (e.g. over a wide field the image will be distorted since passing through different parts of the atmosphere). If the data have been corrected for atmospheric effects (either within the instrument if it has an atmospheric dispersion compensator, or by the reduction software, or it simply does not require to be corrected as e.g. usually in the near-infrared), the keyword `E3D_ADC` is set to *true* in the primary header. If this is not the case, i.e. if a correction is needed the keyword shall be set to *false*, `E3D_ADC = F`. In this latter case, the transformation to coordinates (for each group) must be specified for a fixed reference wavelength and the keywords to derive the correction (airmass, parallactic angle, pressure and temperature) are given in the group table for each group.

4.3.5 Example of a data table header continued

Remember: Luke's 'Ultimate 3D Spectrograph' just complies with all the above and the spectra are all from 1 spaxel (no binning applied yet), so that NSPAX$= 1$ *throughout the table and THUS columns 4 and 5 host only 1 scalar each (no vectors as in the case of binned or compound spaxels).*

```
...
TFORM3  =                   1J / field 3 contains one signed 32-bit integer
TTYPE3  =               'NSPAX' / number of spaxels associated with the spectrum
TFORM6  =                   1D / field 6 contains a 32-bit float
TTYPE6  =                 'XPOS' / spatial element X position
TUNIT6  =              'ARCSEC' / units of the X coordinates
TFORM7  =                   1D / field 7 contains a 32-bit float
TTYPE7  =                 'YPOS' / spatial element Y position
TUNIT7  =              'ARCSEC' / units of the X coordinates
TFORM8  =                   1J / field 8 contains one unsigned 8-bit integer
TTYPE8  =             'GROUP_N' / IFU group number
...
TCTYP6  =                  'RA' / axis type
TCTYP7  =                 'DEC' / axis type
TCUNI6  =              'ARCSEC' / axis unit
TCUNI7  =              'ARCSEC' / axis unit
TCRVL6  =                   1D / reference value
TCRVL7  =                   1D / reference value
TCDLT6  =                   1D / coordinate increment
TCDLT7  =                   1D / coordinate increment
TCRPX6  =                   1D / reference point
TCRPX7  =                   1D / reference point
TP66    =                   1D / transformation matrix
TP67    =                   1D / transformation matrix
TP76    =                   1D / transformation matrix
TP77    =                   1D / transformation matrix
TC66    =                   1D / transformation matrix
TC67    =                   1D / transformation matrix
TC76    =                   1D / transformation matrix
TC77    =                   1D / transformation matrix
TWCS6   =          'Rest frame' / coordinate name
TWCS7   =          'Rest frame' / coordinate name
TCRD6   =                   1D / random error
TCRD7   =                   1D / random error
TCSY6   =                   1D / systematic error
TCSY7   =                   1D / systematic error
...
```

4.4 Spatial binning / Compound spaxels

You can safely skip over this section if it is your first lecture of this document.

It will happen that, during the data reduction and/or analysis, spectra of different spaxels will get combined to produce a single spectrum. Actually, experience showed that often, e.g. to define a background spectrum, spectra of different regions and non-adjacent spaxels will get combined to one. Alternatively, to increase the signal to noise, spaxel can get binned e.g. 2 by 2. In these cases, the spectrum defined in 1 data table *row* comes from several spatial positions, i.e. spaxels.

In some cases, the user will not care to carry along the exact knowledge of what happen in the data file, in others the spatial information is no longer of any importance. However, in many cases the user will want to be able to trace back what happened and also be able to associate a reliable position information to the spectra. For these cases, columns 6, 7, 8 and 9 are allowed to host vectors of floats. These vectors carry then the position information for all instrument spaxels that went into building the spectrum of that row.

A few things to remember when using compound spaxels:

- only columns 6, 7, 8, and 9 are affected and the information is only used for the spatial aspect
- since each 'x' must have an associated 'y' and each of these genuine spaxels must have an associated group number and spaxel ID, the vectors in columns 6, 7, 8 and 9 must have the same dimension.
- **important** in the Euro3D case, we do not allow elements in columns to vary their size from row to row. I.e. the header keyword on the vector dimension in each of the columns 6, 7, 8 and 9 is set only once in the header (to the maximum length in the table) and all vectors in the different rows will have this dimension. Vectors are filled with IEEE NaN values (all bits set) for the unused elements. E.g. if in a data table a single spectrum in the whole file comes from a compound spaxels (made of *j* genuine single spaxels), all others come from single spaxels, the header keyword TFORM will nevertheless have the value *j*D for columns 6 and 7 and in all rows the position will be stored as vectors.

4.4.1 Examples of a data table with compound spaxel table

Example 1: *Obi-Wan used the first of Luke's exposures to apply a 2×2 spatial binning and achieve a higher signal to noise in the spectra (used to study the Death's Star composition). The following shows the header of the data table and the associated content of the columns.*

The header of the data table of a single 'Ultimate 3D Spectrograph' exposure binned 2 by 2 looks as follows:

```
XTENSION=             BINTABLE / type of extension (binary table)
BITPIX  =                    8 / must be 8 for a binary table
NAXIS   =                    2 / table with rows and columns
NAXIS1  =                32784 / number of bytes per row 16+(8+4+4)x2048
NAXIS2  =                 4096 / number of rows (spectra) here 64x64
PCOUNT  =                    0 / no data in the extension after the binary table
GCOUNT  =                    1 / always 1 for binary tables
TFIELDS =                   12 / 12 and only 12 columns in the data table
EXTNAME =             E3D_DATA / extension name for Euro3D position table
...
TFORM3  =                   1J / field 3 contains one signed 32-bit integer
TTYPE3  =               'NSPAX' / number of spaxels associated with the spectrum
TFORM6  =                   4D / field 6 contains four (2x2) 64-bit float
TTYPE6  =                 'XPOS' / spatial element X position
TUNIT6  =              'ARCSEC' / units of the X coordinates
TFORM7  =                   4D / field 7 contains four (2x2) 64-bit float
TTYPE7  =                 'YPOS' / spatial element Y position
TUNIT7  =              'ARCSEC' / units of the X coordinates
TFORM8  =                   4J / field 8 contains four signed 32-bit integer
TTYPE8  =            'GROUP_N' / IFU group number
TFORM9  =                   4A / NSPAX=4 thus the entry is four 8-bit ASCII character stri
TTYPE9  =              SPAX_ID / user ID for each spaxel
...
```

Note that columns 6,7,8,9 host now vectors of 4 elements (of the 2 by 2 binning).

The relevant columns in the table could have the following content:

NSPAX	XPOS	YPOS	GROUP_N	SPAX_ID
4	(1,1,2,2)	(1,2,1,2)	(1,1,1,1)	(a1,a2,b1,b2)
4	(3,3,4,4)	(1,2,1,2)	(1,1,1,1)	(a3,a4,b3,b4)
4	(5,5,6,6)	(1,2,1,2)	(1,1,1,1)	(a5,a6,b5,b6)
...	...	...	...	...

Note that all genuine spaxel come from a single exposure and are thus belonging to the same group. That group number in column 8 refers to **the group table** (see Sect. 5) which could look like this:

GROUP_N	G_SHAPE	G_SIZE1	G_ANGLE	G_SIZE2	G_POSWAV	G_AIRMAS	G_PARANG	G_PRESSU	G_TEMPER	G_HUMID
1	HEXAGON	0.453	0	NaN	5432.78	1.21	17.4	926	12.3	13

Example 2: *Chewbacca attempts to extract some information from Han Solo's Death Star mosaic (see Sect. 5). He leaves the survey 3D images untouched, but bins the short range 3D camera image in lines, i.e. collapses each row (10 elements) to one spaxel (somewhat to the surprise of Obi-Wan who, however, wisely avoids any harsh criticism in the presence of Chewbacca).*

The header of the data table of the new mosaic exposure looks as follows:

```
...
TFORM3  =                  1J / field 3 contains one signed 32-bit integer
TTYPE3  =              'NSPAX' / number of spaxels associated with the spectrum
TFORM6  =                 10D / field 6 contains ten 64-bit float
TTYPE6  =               'XPOS' / spatial element X position
TUNIT6  =             'ARCSEC' / units of the X coordinates
TFORM7  =                 10D / field 7 contains ten 64-bit float
TTYPE7  =               'YPOS' / spatial element Y position
TUNIT7  =             'ARCSEC' / units of the X coordinates
TFORM8  =                 10J / field 8 contains ten signed 32-bit integer
TTYPE8  =           'GROUP_N' / IFU group number
...

TFORM9  =                 10A / NSPAX=1 thus the entry is ten 8-bit ASCII character strin
TTYPE9  =            SPAX_ID / user ID for each spaxel
...
```

Note that the largest compound spaxel ($\sim$ binning) determines the number of elements in each vector of columns 6,7,8,9: here 10 genuine spaxels are binned.

The content of the data table could look as follows:

Each survey 3D Spectrograph covers 20 by 20 circular spaxels on the sky and the two sit next to each other (with a gap equivalent to 4 spaxels). They remain unbinned, thus `NSPAX = 1` and form groups 1 and 2. The short range 3D camera has originally a field of view of 10 by 10 rectangular spaxels, reduced now to 10 spaxels (each composed of 10 genuine instrument spaxels). Note also, that all three exposures share the same WCS, but that the short range 3D camera has spaxels that are roughly 6.6 times smaller on the sky than the ones of the survey 3D cameras (see Example 2 in Sect. 5). This explains the X and Y values in the third block below.

NSPAX	XPOS	YPOS	GROUP_N	SPAX_ID
1	(1.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(1,NaN,...,NaN)	(survey1_spec1,NaN,NaN,...,NaN)
1	(2.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(1,NaN,...,NaN)	(survey1_spec2,NaN,NaN,...,NaN)
1	(3.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(1,NaN,...,NaN)	(survey1_spec3,NaN,NaN,...,NaN)
...	...	...	...	...
1	(25.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(2,NaN,...,NaN)	(survey2_spec1,NaN,NaN,...,NaN)
1	(26.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(2,NaN,...,NaN)	(survey2_spec2,NaN,NaN,...,NaN)
1	(27.0,NaN,NaN,...,NaN)	(1.0,NaN,NaN,...NaN)	(2,NaN,...,NaN)	(survey2_spec3,NaN,NaN,...,NaN)
...	...	...	...	...
10	(23.55,23.70,...,25.05)	(31.00, 31.00,...,31.00)	(3,...,3)	(short_spec1,short_spec11,...,short_spec91)
10	(23.55,23.70,...,25.05)	(31.15, 31.15,...,31.15)	(3,...,3)	(short_spec2,short_spec12,...,short_spec92)
...	...	...	...	...

The first 2 blocks of 20$\times$20=400 spaxels are unbinned and have each one associated group value. The last 10 spaxels are composed of 10 by 1 genuine spaxels.

4.5 The data spectrum

The data spectra are given as vectors of short or long integers, floats or doubles.

Euro3D (as FITS) only allows a single value for the reference wavelength and step size of the spectra. In 2D representations, the wavelength sampling would be described by the keywords `CRPIXn, CRVALn, CDELTn, CTYPEn`, where `n` is the axis number in the dispersion direction. These keywords, however, are meaningless here as the spectra are single entries in table rows. Instead, in the Euro3D format, we use the keywords `CRVALS, CDELTS, CTYPES, CUNITS` where `S` stands for Spectra; these keywords are unique and apply to all spectra, including science, data quality and statistical error spectra.

The meaning of each keyword was described above. We remind here that CRVALS and CDELTS are used together with columns 4 and 5 to define the wavelength of each vector element. The wavelength is express in units CTYPES, and each vector element is in units CUNITS.

For a given spectrum in row SPEC_ID, the wavelength associated with the n^{th} element (where $1 \leq n \leq \mathrm{SPE\_LEN_{SPEC\_ID}}$) in column 10, 11 or 12 is:

$$\lambda^{n}_{\mathrm{SPEC\_ID}} = \mathrm{CRVALS} + ((\mathrm{SPE\_STA_{SPEC\_ID}} + (n-1)) \times \mathrm{CDELTS})$$

Regions without data in the science spectra must have the value NaN and be flagged as such in the data quality spectra (see below).

4.5.1 Mandatory header keywords for the data spectrum

The following keywords (and their values when indicated) are mandatory:

`CTYPES = [STRING]` : wavelength unit (e.g.ANGSTROM, NM, MICRON, ...)
`CRVALS = [DOUBLE]`: reference wavelength
`CDELTS = [DOUBLE]` : wavelength increment per pixel
...
`TFORM4 = 1J` : field 4 contains one signed 32-bit integer
`TTYPE4 = 'SPEC_LEN'`: length of useful spectrum (in number of vector elements)
`TFORM9 = 1J` : field 5 contains one signed 32-bit integer
`TTYPE9 = 'SPEC_STA'`: starting vector element of the useful spectrum
`TFORM10 = rI/J/E/D` : field 10 contains a number r of shorts/longs/floats/doubles
`TTYPE10 = 'DATA_SPE'`: Data spectrum

Note that TUNIT10 is substituted by CTYPES and CUNITS.

4.5.2 Example of the data table header continued

Luke's 'Ultimate 3D Spectrograph' has 64×64 spaxels, each spectrum occupying 2048 detector pixels. He observed the Death Star in the optical light (4197.80 Å< λ <9604.21 Å).

```
CTYPES  =             'ANGSTROM'/ wavelength scale (along NAXIS1)
CRVALS  =               4197.80 / reference wavelength
CDELTS  =               2.63985 / wavelength increment per pixel
...
TFORM4  =                    1J / field 4 contains one signed 32-bit integer
```

```
TTYPE4  =           'SPEC_LEN' / length of useful spectrum
TFORM5  =                  1J / field 5 contains one signed 32-bit integer
TTYPE5  =           'SPEC_STA' / starting vector element of the useful spectrum
TFORM10 =               2048D / each spectrum covers 2048 detector pixels
TTYPE10 =           'DATA_SPE' / Data spectrum
...
```

4.6 The data quality spectrum

This entry is **mandatory**, since it contains, the information on the valid part of each spectrum, i.e. whether its wavelength range spans the entire row or whether a portion at the beginning, in the middle or the end of a row can be ignored. Note that this spectrum allows the user to leave "wavelength holes" in the data, without changing the pixel values in the data.

The data quality spectrum has identical dimensions and wavelength sampling as the data spectrum, but is described by a 32-bit integer values. The data quality is used to set a data quality flag for each pixel. It serves two purposes: *i)* to mask out pixels that do not contain any data (e.g. in the case that not all spectra span the same wavelength range) and *ii)* to flag any pixel affected by one or more anomalies (such as detector flaws, cosmic ray hits, ...).

4.6.1 Data quality convention

The Euro3D format uses the following convention (recognized by the software when the `EURO3D` keyword is set to True). The higher the value, the more severe is the problem :

Bit #	Flag value	Quality condition
0	0	Good pixel – no flaw detected
1	1	Affected by telluric feature (corrected)
2	2	Affected by telluric feature (uncorrected)
3	4	Ghost/stray light at $> 10\%$ intensity level
4	8	Electronic pickup noise
5	16	Cosmic ray (removed)
6	32	Cosmic ray (unremoved)
7	64	low QE pixel ($< 20\%$ of the average sensitivity; e.g. defective CCD coating, vignetting...)
8	128	Calibration file defect (if pixel is flagged in any calibration file)
9	256	Hot pixel ($> 5\sigma$ median dark)
10	512	Dark pixel (permanent CCD charge trap)
11	1024	Questionable pixel (lying above a charge trap which may have affected it)
12	2048	Detector potential well saturation (signal irrecoverable, but known to exceed the max. e^- number)
13	4096	A/D converter saturation (signal irrecoverable, but known to exceed the A/D full scale signal)
14	8192	Permanent camera defect (such as blocked columns, dead pixels)
15	16384	Bad pixel not fitting into any other category
...	...	User defined
31	2^{30}	missing data (pixel was lost)
32	2^{31}	outside data range (outside of spectral range, inactive detector area, mosaic gap, ...)

Since each condition is linked to a bit, several simultaneous conditions can be directly expressed as the sum of all corresponding flag values. Example: a pixel with calibration defects, known as a hot pixel and saturated in this spectrum would have a flag value of $128 + 256 + 4096 = 4480$.

4.6.2 Mandatory header keywords for the data quality spectrum

The following keywords (and their values when indicated) are mandatory:

`TFORM11 = rJ` : field 11 contains a number r of signed 32-bit integers
`TTYPE11 = 'QUAL_SPE'`: Data Quality spectrum

Note that TUNIT11 is substituted by CTYPES.

4.6.3 Example of a data quality info in the header

Obi-Wan computed the data quality image for Luke's observations and attached it as the DQ spectrum in the Euro3D format. Note that Obi-Wan (in his infinite wisdom) correctly sampled the data quality spectrum to the same dimensions and identical CTYPES, CRVALS, CDELTS values as the data spectrum — an otherwise current source of mistake of younger Jedis.

```
...
CTYPES  =             'ANGSTROM'/ wavelength scale (along NAXIS1)
CRVALS  =               4197.80 / reference wavelength
CDELTS  =               2.63985 / wavelength increment per vector element
...
TFORM11 =                 2048J / each spectrum covers 2048 detector pixels
TTYPE11 =            'QUAL_SPE' / Data quality spectrum
...
```

Note that the CTYPES, CRVALS, CDELTS cannot be repeated explicitly for the DQ spectrum. Note further that the length of the DQ spectrum (here 2048 detector pixels) *must* be identical to the length of the data spectrum.

4.7 The (optional) statistical error spectrum

This spectrum is **optional**. Note, however, that it is required by some software packages in order to produce meaningful results.

The statistics spectrum has identical dimensions and wavelength sampling as the data and data quality spectra. It stores for each vector element of the spectrum the statistical properties (standard deviation) carried over from previous reduction process(es). It is stored as a vector of short or long integers, floats or doubles, but of identical type as the data spectrum.

4.7.1 Mandatory header keywords for the data quality spectrum

If the extension is present, the following keywords and values are mandatory information to provide:

`TFORM12 = rI/J/E/D` : field 12 contains a number r of shorts/longs/floats/doubles
`TTYPE12 = 'STAT_SPE'`: Statistical error spectrum

Note that TUNIT12 is substituted by CTYPES.

4.7.2 Example of a statistical error information in the header

Since Obi-Wan meticulously reduced the data, he carefully kept track of the statistical error associated with each detector pixel (unlike e.g. Jabba the Hutt who systematically ignores all errors and thus faces uninterpretable results when using the Euro3D reduction software [unregistered copy on top of that]). The statistical errors are stored in column 12, the header of the data table is updated as given below for a single 'Ultimate 3D Spectrograph' exposure of Luke.

```
...
CTYPES  =             'ANGSTROM'/ wavelength scale (along NAXIS1)
CRVALS  =               4197.80 / reference wavelength
CDELTS  =               2.63985 / wavelength increment per pixel
...
TFORM12 =                 2048D / each spectrum covers 2048 detector pixels
TTYPE12 =           'STAT_SPE' / Statistical error spectrum
...
```

Note that the CTYPES, CRVALS, CDELTS, CUNITS cannot be repeated explicitly for the statistical error spectrum. Note further that the length of the statistical error spectrum (here 2048 detector pixels) *must* be identical to the length of the data (and DQ) spectrum.

5 Group table extension

The group table is **mandatory**. It is associated with the data table and defines the spaxel shape, as well as hosting position information on each group. The number of rows in the group table is equal to the number of groups defined in column 8 of the data table.

For example, if the file hosts data from one single exposure with one instrument, the number of group will be 1 and the group table will have one row only. If the file contains data of a mosaic of three exposures taken with the same instrument but at different orientations on the sky and at different airmasses (different atmospheric dispersion corrections), the number of groups will be 3, and the group table will have 3 rows. For example, if the instrument has several deployable IFUs, each IFU will get an associated group number, leading to several rows in the group table. Alternatively, it is also possible to declare a central IFU bundle together with a ring of different sky fibers at the periphery, or other mixed configurations.

The group table extension is identified by `EXTNAME = E3D_GRP`.

5.1 Structure of the table

The group table is a binary table which, for each group in the position table, contains 11 (and only 11) columns. Namely:

- column 1: the group number corresponding to column 8 of the data table
- column 2: the instrument spaxel shape (e.g. SQUARE, RECTANGULAR, HEXAGON, CIRCLE, ...)
- column 3: size parameter 1
- column 4: angle parameter
- column 5: size parameter 2
- column 6: wavelength for which the WCS matrix is valid (in the case that atmospheric dispersion has not been corrected for)
- column 7: airmass at which the group's observation was performed
- column 8: parallactic angle at which the group's observation was performed
- column 9: pressure at which the group's observation was performed
- column 10: temperature at which the group's observation was performed
- column 11: humidity at which the group's observation was performed

• Column 1 hosts the group number as given in column 6 of the data table.

• Column 2 gives the shape of a genuine instrument spaxel. At present, instruments with circular (e.g. fibre), square or rectangular (e.g. slicer based IFUs) and hexagonal (e.g. lenslet based

IFUs) spaxels have been developed. Thus, valid entries to this column are: `SQUARE`, `RECTANG`, `HEXAGON`, `CIRCLE`.

• Columns 3 to 5 define each of these shapes by one or more parameters given in the columns 3 to 5. **All three columns are mandatory, although only one or two might be used by certain shapes**.

For the `CIRCLE` only the radius of the spaxel is needed, i.e. only column 3 will be considered. If the entry in column 2 is `SQUARE` or `HEXAGON` the columns 3 and 4 are required, providing the size and orientation as defined in Figs. 2 and 3. If the entry in column 2 is `RECTANG`, columns 3, 4 and 5 are needed in order to define size 1, orientation and size 2 (column 3, 4 and 5, respectively) (see figures). Columns that have no valid entry for a given shape must be set to `NaN`.

• Columns 6 to 11 are needed if atmospheric dispersion corrections have to be made. The information is only processed if in the primary header the keyword `E3D_ADC` is set to *false*. Column 6 gives the wavelength for which the WCS matrix given in the header of the position table is valid. Column 7 to 11 give the information necessary to compute the atmospheric correction.

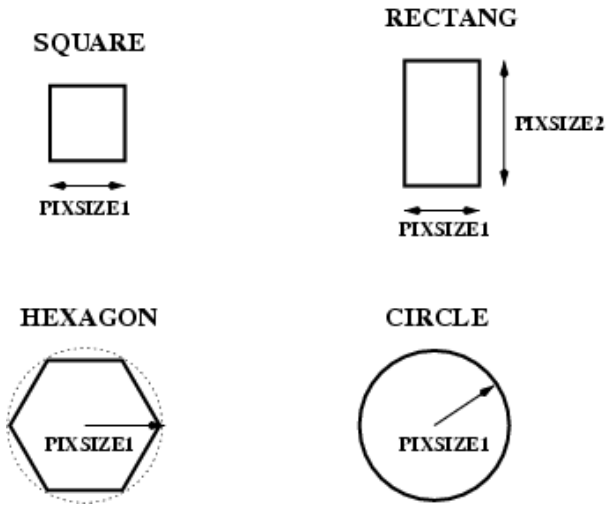

Figure 2: The figure is illustrating the size parameters for 'SQUARE', 'RECTANG', 'HEXAGON', 'CIRCLE' shaped spaxels, together with the definition of `G_SIZE1` and `G_SIZE2`.

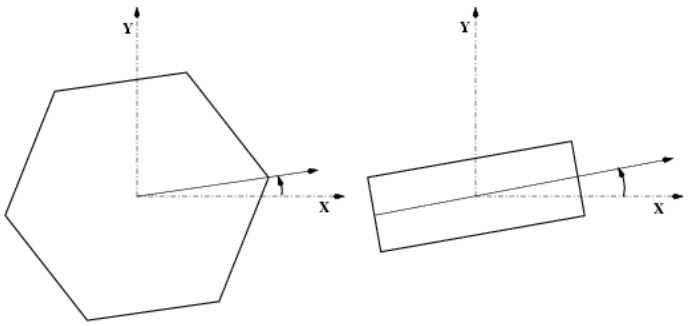

Figure 3: Definition of the orientation `G_ANGLE` for the shapes 'HEXAGON' and 'RECTANG': the angle is given in degrees, counter-clockwise. 'SQUARE' follows the same definition as 'RECTANG'.

5.2 Mandatory information

The following keywords (and their values when indicated) are mandatory:

`XTENSION = 'BINTABLE'` : type of extension (binary table)
`BITPIX = 8` : binary table
`NAXIS = 2` : table with rows and columns
`NAXIS1 = [INTEGER]`: number of bytes per row
`NAXIS2 = [INTEGER]`: number of rows (equal to the number of groups defined in the position table)
`PCOUNT = 0`: always 0
`GCOUNT = 1`: always 1
`TFIELDS = 11` : number of columns fixed to 11 in the group extension
`EXTNAME = E3D_GRP` : extension name for Euro3D group table
`TFORM1 = 1B` : field 1 contains one signed 8-bit integer
`TTYPE1 = 'GROUP_N'`: group number (in position table)

`TFORM2 = 1A` : field 2 contains one 8-bit ASCII character string
`TTYPE2 = 'G_SHAPE'`: type of spaxels in group *n*, allowed are `SQUARE, RECTANG, HEXAGON, CIRCLE`
`TFORM3 = 1D` : field 3 contains one 64-bit float
`TTYPE3 = 'G_SIZE1'`: first size parameter (use for all shapes)
`TUNIT3 = [STRING]` : units of the parameter
`TFORM4 = 1D` : field 4 contains one 64-bit float
`TTYPE4 = 'G_ANGLE'`: orientation of the spaxel (used for all shapes but CIRCLE)
`TUNIT4 = [STRING]` : units of the parameter
`TFORM5 = 1D` : field 5 contains one 64-bit float
`TTYPE5 = 'G_SIZE2'`: second size parameter (only used for RECTANG)
`TUNIT5 = [STRING]` : units of the parameter

`TFORM6 = 1D` : field 6 contains one 64-bit float
`TTYPE6 = 'G_POSWAV'`: wavelength for which the WCS is valid
`TUNIT6 = [STRING]` : units of the wavelength
`TFORM7 = 1D` : field 7 contains one 64-bit float
`TTYPE7 = 'G_AIRMAS'`: airmass for ADC
`TFORM8 = 1D` : field 8 contains one 64-bit float
`TTYPE8 = 'G_PARANG'`: parallactic angle for ADC
`TUNIT8 = [STRING]` : units of the parallactic angle
`TFORM9 = 1D` : field 9 contains one 64-bit float
`TTYPE9 = 'G_PRESSU'`: pressure for ADC
`TUNIT9 = [STRING]` : units of the pressure
`TFORM10 = 1D` : field 10 contains one 64-bit float
`TTYPE10 = 'G_TEMPER'`: temperature for ADC
`TUNIT10 = [STRING]` : units of the temperature
`TFORM11 = 1D` : field 11 contains one 64-bit float
`TTYPE11 = 'G_HUMID'`: humidity for ADC
`TUNIT11 = [STRING]` : units of the pressure

`END`

5.3 Examples of a group table (header and content)

Example 1: *Luke took a mosaic of two exposures of the Death Star. For the second exposure he rotated the 'Ultimate 3D Spectrograph' by 45 degrees with respect to the first position. His instrument has hexagonal spaxels. OWL is located on the moon Yavin 4 which is surrounded by a thick Nitrogen/Oxygen atmosphere, therefore both exposures need to be corrected for atmospheric dispersion. The group table of the file containing the mosaic of both exposures looks as follows:*

Header

```
XTENSION=            'BINTABLE'/ IMAGE extension type
BITPIX  =                    8 / 32 bit integer
NAXIS   =                    2 / data array is 2 dimensional
NAXIS1  =                   34 / number of bytes per row
NAXIS2  =                    2 / number of groups
PCOUNT  =                    0 /
GCOUNT  =                    1 /
TFIELDS =                   10 / number of columns (8-10, here 9)
EXTNAME =            'E3D_GRP '/ Euro3D spectral extension
TFORM1  =                   1B / the group number is given as a 8-bit integer
TTYPE1  =            'GROUP_N '/ column 1 contains the group number
TFORM2  =                   1A / the shape is given as ASCII string
TTYPE2  =            'G_SHAPE '/ column 2 contains the shape keyword
TFORM3  =                   1D / field 3 contains one 64-bit float
TTYPE3  =             'G_SIZE1'/ first spaxel size parameter for the group
TUNIT3  =            'ARCSEC  '/ units of the parameter
TFORM4  =                   1D / field 4 contains one 64-bit float
TTYPE4  =             'G_ANGLE'/ orientation of the spaxel
TUNIT4  =            'DEGREE  '/ units of the parameter
TFORM5  =                   1D / field 5 contains one 64-bit float
TTYPE5  =             'G_SIZE2'/ second spaxel size parameter for the group
TUNIT5  =            'ARCSEC  '/ units of the parameter
TFORM6  =                   1D / the wavelength is given as a 64-bit float
TTYPE6  =            'G_POSWAV'/ column 6 contains the wavelength for ADC
TUNIT6  =            'ANGSTROM'/ units of the wavelength
TFORM7  =                   1D / the airmass is given as a 64-bit float
TTYPE7  =            'G_AIRMAS'/ column 7 contains the airmass for ADC
TFORM8  =                   1D / the parallactic angle is given as a 32-bit float
TTYPE8  =            'G_PARANG'/ column 8 contains the parallactic angle for ADC
TUNIT8  =            'DEGREE  '/ units of the parallactic angle
TFORM9  =                   1D / the pressure is given as a 64-bit float
TTYPE9  =            'G_PRESSU'/ column 9 contains the pressure for ADC
TUNIT9  =            'MILLIBAR'/ units of the pressure
TFORM10 =                   1D / the temperature is given as a 64-bit float
TTYPE10 =            'G_TEMPER'/ column 10 contains the temperature for ADC
TUNIT10 =            'KELVIN  '/ units of the temperature
TFORM11 =                   1D / the humidity is given as 64-bit float
TTYPE11 =             'G_HUMID'/ humidity for ADC
TUNIT11 =             'PERCENT'/ the humidity is given in percent
```

Content

GROUP_N	G_SHAPE	G_SIZE1	G_ANGLE	G_SIZE2	G_POSWAV	G_AIRMAS	G_PARANG	G_PRESSU	G_TEMPER	G_HUMID
1	HEXAGON	0.453	0	NaN	5432.78	1.21	17.4	926	12.3	15.5
2	HEXAGON	0.453	45	NaN	5432.78	1.17	23.5	923	12.6	15.6

Example 2: *Han Solo also managed to take a few pictures of the Death Star with the on-board cameras of his Millennium Falcon. The Millennium Falcon is carrying a survey 3D Spectrograph (two arrays of circular spaxels), as well as a short range 3D camera (rectangular pixels). Obi-Wan combined the three images of the two different IFUs (circular and rectangular spatial spaxels) into a mosaic. Images obtained from the spacecraft do not need to be corrected for atmospheric dispersion. The group table of the mosaic looks as follows:*

Header

```
XTENSION=            'BINTABLE'/ IMAGE extension type
BITPIX  =                    8 / 32 bit integer
NAXIS   =                    2 / data array is 2 dimensional
NAXIS1  =                   34 / number of bytes per row
NAXIS2  =                    3 / number of groups
PCOUNT  =                    0 /
GCOUNT  =                    1 /
TFIELDS =                   10 / number of columns (8-10, here 9)
EXTNAME =            'E3D_GRP '/ Euro3D spectral extension
TFORM1  =                   1B / the group number is given as a 8-bit integer
TTYPE1  =            'GROUP_N '/ column 1 contains the group number
TFORM2  =                   1A / the shape is given as ASCII string
TTYPE2  =            'G_SHAPE '/ column 2 contains the shape keyword
TFORM3  =                   1D / field 3 contains one 64-bit float
TTYPE3  =             'G_SIZE1'/ first spaxel size parameter for the group
TUNIT3  =            'ARCSEC  '/ units of the parameter
TFORM4  =                   1D / field 4 contains one 64-bit float
TTYPE4  =             'G_ANGLE'/ orientation of the spaxel
TUNIT4  =            'DEGREE  '/ units of the parameter
TFORM5  =                   1D / field 5 contains one 64-bit float
TTYPE5  =             'G_SIZE2'/ second spaxel size parameter for the group
TUNIT5  =            'ARCSEC  '/ units of the parameter
TFORM6  =                   1D / the wavelength is given as a 64-bit float
TTYPE6  =            'G_POSWAV'/ column 6 contains the wavelength for ADC
TUNIT6  =            'ANGSTROM'/ units of the wavelength
TFORM7  =                   1D / the airmass is given as a 64-bit float
TTYPE7  =            'G_AIRMAS'/ column 7 contains the airmass for ADC
TFORM8  =                   1D / the parallactic angle is given as a 64-bit float
TTYPE8  =            'G_PARANG'/ column 8 contains the parallactic angle for ADC
TUNIT8  =            'DEGREE  '/ units of the parallactic angle
TFORM9  =                   1D / the pressure is given as a 64-bit float
TTYPE9  =            'G_PRESSU'/ column 9 contains the pressure for ADC
TUNIT9  =            'MILLIBAR'/ units of the pressure
TFORM10 =                   1D / the temperature is given as a 64-bit float
```

```
TTYPE10 =            'G_TEMPER'/ column 10 contains the temperature for ADC
TUNIT10 =            'KELVIN  '/ units of the temperature
TFORM11 =                   1D / the humidity is given as 64-bit float
TTYPE11 =             'G_HUMID'/ humidity for ADC
TUNIT11 =             'PERCENT'/ the humidity is given in percent
END
```

Content

GROUP_N	G_SHAPE	G_SIZE1	G_ANGLE	G_SIZE2	G_POSWAV	G_AIRMAS	G_PARANG	G_PRESSU	G_TEMPER	G_HUMID
1	CIRCLE	0.35	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	CIRCLE	0.35	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	RECTANG	0.053	53.2	0.072	NaN	NaN	NaN	NaN	NaN	NaN

6 Science table(s) extension

This extension is **optional**.

The science table extension is in binary table format and is used to store any scientific quantities derived from each spectrum. For example, these could be for galaxy spectra values such as the line of sight velocity and velocity dispersion, element abundances, etc...

The science table is optional. However, if present, **the first column must contain the spectrum identifier, as in the data table, of the spectrum to which the data in this row corresponds**.

Note that the spatial information is not repeated in the science table, but could be recovered from the data table if needed.

The science table extensions are identified by `EXTNAME = E3D_SCI`*n*, where *n* is used to number each extension.

6.1 Mandatory information

If the extension is present, the following keywords and values are mandatory information to provide:

`XTENSION = 'BINTABLE'` : type of extension (binary table)
`BITPIX = 8` : binary table
`NAXIS = 2` : table with rows and columns
`NAXIS1 = [INTEGER]`: number of bytes per row
`NAXIS2 = [INTEGER]`: number of rows ((equal to the number of rows in the spectrum extension)
`PCOUNT = [INTEGER]`: number of bytes in extension beyond the binary table
`GCOUNT = 1`: always 1
`TFIELDS = [INTEGER]` : number of columns
`TFORM1 = 1J` : format of column 1 = 1 signed 32-bit integer
`TTYPE1 = 'SPEC_ID'`: spatial element row number (in 2D spectra)
...
`EXTNAME = E3D_SCI`*n* : extension name for Euro3D science table
...
`END`

A Example of a data extension header

Luke Skywalker observed the Death Star using the 'Ultimate 3D Spectrograph' on the OWL telescope owned by the Jedi Council. The data comply (of course) with the Euro3D format. Obi-Wan Kenobi did the data reduction and analysis (using the Power-auto-reduction software Euro3D), and published the results in the Jedi Journal of Science (Kenobi et al. 2011, JJoS 3475, 237).

Luke's 'Ultimate 3D Spectrograph' has 64×64 spaxels, each spectrum occupying 2048 detector pixels. He observed the Death Star in the optical light (4197.80 Å$< \lambda <$9604.21 Å).

```
XTENSION=             BINTABLE / type of extension (binary table)
BITPIX  =                    8 / must be 8 for a binary table
NAXIS   =                    2 / table with rows and columns
NAXIS1  =                32784 / number of bytes per row 16+(8+4+4)x2048
NAXIS2  =                 4096 / number of rows (spectra) here 64x64
PCOUNT  =                    0 / no data in the extension after the binary table
GCOUNT  =                    1 / always 1 for binary tables
TFIELDS =                   12 / 12 and only 12 columns in the data table
EXTNAME =             E3D_DATA / extension name for Euro3D position table
...
CTYPES  =            'ANGSTROM'/ wavelength scale (along NAXIS1)
CRVALS  =              4197.80 / reference wavelength
CDELTS  =              2.63985 / wavelength increment per pixel
...
TFORM1  =                   1J / spectrum ID expressed a one signed 32-bit integer
TTYPE1  =              SPEC_ID / spectrum ID
TFORM2  =                   1L / selection flag is a logical value (1bit)
TTYPE2  =             SELECTED / selection flag\\
...
TFORM3  =                   1J / field 3 contains one signed 32-bit integer
TTYPE3  =              'NSPAX' / number of spaxels associated with the spectrum
...
TFORM4  =                   1J / field 4 contains one signed 32-bit integer
TTYPE4  =           'SPEC_LEN' / length of useful spectrum
TFORM5  =                   1J / field 5 contains one signed 32-bit integer
TTYPE5  =           'SPEC_STA' / starting vector element of the useful spectrum
...
TFORM6  =                   1D / field 6 contains a 32-bit float
TTYPE6  =                'XPOS' / spatial element X position
TUNIT6  =              'ARCSEC' / units of the X coordinates
TFORM7  =                   1D / field 7 contains a 32-bit float
TTYPE7  =                'YPOS' / spatial element Y position
TUNIT7  =              'ARCSEC' / units of the X coordinates
TFORM8  =                   1J / field 8 contains one unsigned 8-bit integer
TTYPE8  =            'GROUP_N' / IFU group number
...
TFORM9  =                   1A / NSPAX=1 thus the entry is only one 8-bit ASCII char strin
TTYPE9  =              SPAX_ID / user ID for each spaxel
```

```
...
TFORM10 =                2048D / each spectrum covers 2048 detector pixels
TTYPE10 =           'DATA_SPE' / Data spectrum
TFORM11 =                2048J / each spectrum covers 2048 detector pixels
TTYPE11 =           'QUAL_SPE' / Data quality spectrum
TFORM12 =                2048D / each spectrum covers 2048 detector pixels
TTYPE12 =           'STAT_SPE' / Statistical error spectrum
...
TCTYP6  =                 'RA' / axis type
TCTYP7  =                'DEC' / axis type
TCUNI6  =             'ARCSEC' / axis unit
TCUNI7  =             'ARCSEC' / axis unit
TCRVL6  =                   1D / reference value
TCRVL7  =                   1D / reference value
TCDLT6  =                   1D / coordinate increment
TCDLT7  =                   1D / coordinate increment
TCRPX6  =                   1D / reference point
TCRPX7  =                   1D / reference point
TP66    =                   1D / transformation matrix
TP67    =                   1D / transformation matrix
TP76    =                   1D / transformation matrix
TP77    =                   1D / transformation matrix
TC66    =                   1D / transformation matrix
TC67    =                   1D / transformation matrix
TC76    =                   1D / transformation matrix
TC77    =                   1D / transformation matrix
TWCS6   =         'Rest frame' / coordinate name
TWCS7   =         'Rest frame' / coordinate name
TCRD6   =                   1D / random error
TCRD7   =                   1D / random error
TCSY6   =                   1D / systematic error
TCSY7   =                   1D / systematic error
...
```

___**oOo**___