



omdetect

January 27, 2025

Abstract

A program for detecting sources on OM product images, parameterising and performing aperture photometry on those detected. It is also used for grism images to detect zero and first order spectra.

1 Instruments/Modes

Instrument	Mode
OM	FAST
OM	IMAGING

2 Use

pipeline processing	yes
interactive analysis	yes

3 Description

The primary purpose of **omdetect** is to perform source-detection on either **OM IMAGING** (images produced by **ommodmap**), **FAST** or **GRISM** mode product images. From version **5.17** it also performs upon images produced by **ommosaic**. For **GRISM** images the sources are the zero and first order spectra. For **OM IMAGING** mode data it also performs photometry on the detected sources and sets source flags. Its primary output file is a **FITS** file containing a **SRCLIST** table that contains various parameters of the detected sources.

The following provides an overall description of how **omdetect** works in the **IMAGING** mode.

The image is first loaded into memory from the **PRIMARY** FITS block in the file and also the **QUALITY** image. For image files produced by **ommodmap** the latter image will have been constructed by **omcosflag** and provides a map of where bad-pixels are. For image files produced by **ommosaic**, the **QUALITY** image contains additional information about pixel-quality issues. **Omdetect**'s detection algorithms will ignore any image pixel in which the corresponding quality-pixel is 1 (ie a bad pixel) and during the photometry stage it will use it to set bit 0 of the source-quality flag for sources containing one or more bad pixels.



Omdetect computes a background image in order to identify pixels that may be part of a source. It uses two different algorithms for detecting **point-like** and **extended** sources and a source is classified as either one from a comparison of its computed width with that of the **PSF** for the **OM** filter width. **Omdetect** uses two internal 2-d integer images to record the pixels that are assigned to **point-like** and **extended** sources (the **plevel** and **elevel** images, respectively).

The point-source detection algorithm is run first, and for each pixel higher than a specified value above the local background value various checks are made to see if the pixel could be part of a source. If a likely source is found its position and shape parameters are computed by moments and its classification is determined. Usually the source will be classified as **point-like** (since this algorithm is tuned for such sources) and the pixels on the **plevel** image that are assigned to the source are set equal to the source number. Occasionally the source will be classified as **extended** and then the pixels on the **elevel** image that are assigned to the source are set equal to the source number. The algorithm makes many passes over the image, using a number of different source-identification algorithms tuned to pick up different types of **point-like** sources (eg sources surrounded by mod-8 pattern, sources that have close neighbours), to pick up as many sources as possible. This algorithm largely overcomes the problem of locating sources contaminated by scattered light features. The pixels are examined in the order of decreasing pixel brightness, and this overcomes other problems due to strong mod-8 patterns, etc.

The extended-source detection algorithm works by examining groups of pixels above a background threshold. Various checks are made to ensure that the group of pixels is not just part of a **mod-8 pattern** or part of a **scattered-light feature**. If a likely source is found, its position and shape parameters are computed by moments and it is classified as either **point-like** or **extended**. Usually the source will be classified as the latter, and the **elevel** image pixels assigned to the source are set equal to the source number. Occasionally a faint source may be classified as **point-like**, and in this case the **plevel** image pixels assigned to the source are set equal to the source number. When the detection process is finished a check is made on **extended sources** to see if they can be split into two or more sources.

Having obtained a list of sources, comprising of various parameters such as positions, extension flags and shapes, aperture-photometry is then performed on each source to produce a count-rate for each source that is consistent with the **OM calibration** procedure. **Omdetect** does not compute instrumental magnitudes from these count-rates, and **ommag** needs to be run using the output source-list to accomplish this. Flags are also added to make the user aware of any potential problems with the photometry of a particular source.

An output source-list file is produced containing source positions, parameters measuring their size and shape, source count-rates, various other photometry parameters and source flags. A count-rate limit is written to the source list file header (keyword CRLIMIT) reflecting the mean 5 sigma above background detection threshold for the image. When the input image is a **mosaiced sky-image**, the **SRCLIST** table in the output file will contain columns for the **RA** and **DEC** coordinates. These coordinates will have been computed using the **CRPIX1****CRPIX2** and the **CRVAL1****CRVAL2** values in the header of the input image file.

4 Parameters

This section documents the parameters recognized by this task (if any).

Parameter	Mand	Type	Default	Constraints
-----------	------	------	---------	-------------

set	yes	string	none	
-----	-----	--------	------	--

name of OM product FITS image file (either one produced by **ommodmap** or one produced by **ommosaic**).



outset	yes	string	none	
---------------	-----	--------	------	--

name of output OM source list file

wdxset	no	string	""	
---------------	----	--------	----	--

name of OM priority-window data auxiliary file - needed for images produced by pre 5.4 sas

nsigma	no	real	2.0	1.0–
---------------	----	------	-----	------

number of σ above background required for a detection

minsignificance	no	real	1.0	0.0–
------------------------	----	------	-----	------

minimum significance of sources to be included in the **SRCLIST** table of the output source-list file. if 0 then no checking done.

levelimage	no	string	none	
-------------------	----	--------	------	--

name of image of island detections

backgroundimage	no	string	none	
------------------------	----	--------	------	--

name of background image file

regionfile	no	string		
-------------------	----	--------	--	--

name of ds9 region file

detectextended	no	bool	true	
-----------------------	----	------	------	--

Determines whether algorithm for detecting extended sources should be used

psfphotometryenabled	no	bool	false	
-----------------------------	----	------	-------	--

If set to true the photometry for FAST mode images will be performed using psf fitting, and for IMAGING mode for those sources with close neighbours. In the current version of the task the use of this parameter is disabled.

neighboursforpsfphotometry		integer	1	0-7
-----------------------------------	--	---------	---	-----



Determines the minimal number of neighbours of a source to implement its PSF-photometry (in the case of the parameter *psfphotometryenabled* set to *true*). To save computing time, PSF-photometry is not applied to the sources having the number of close neighbours less than *neighboursforpsfphotometry*.

psfphotometryset	no	string		
-------------------------	----	--------	--	--

name of the output file for the source list with PSF-photometry

maxrawcounttrate	no	real	1.0	0.0–
-------------------------	----	------	-----	------

Maximum raw count-rate (as determined by aperture photometry) above which psf photometry should not be used.

backgroundmethod	no	integer	1	1-5
-------------------------	----	---------	---	-----

Determines which background method should be used- 1=current SAS method, 2=mean, 3=median, 4-mode, 5=using background image.

5 Running the program

The parameter **nsigma** (default value=2) controls the minimum detection threshold; ie all pixels with a value greater than or equal to **background value** + **nsigma** × **background noise** will be considered to be part of a possible source. For values greater than 2, the part of the algorithm that searches for very "faint" sources is bypassed, and when you are only interested in the relatively bright sources then it is a good idea to set this parameter to **2.01**, which will significantly decrease the execution time (this will cause the algorithms that specifically look for very faint sources to be bypassed).

The parameter **minsignificance** (default value=2) controls what sources are kept in the source-list- all detected sources with a computed significance less than or equal to **minsignificance** are discarded. If this parameter is set to 0 then all the detected sources will be kept in the source-list. Of course, the lower the value of minsignificance the more spurious sources may be in the source-list. If it is thought that there are too many spurious sources, then it is a good idea to re-run **omdetect** with a **minsignificance** value of 3, or maybe even 4.

The parameter **detectextended** determines if the algorithm is to search for "large" extended sources (**true**), or not (**false**). If you are not interested in such sources, a considerable amount of computing time can be saved if this parameter is set to **false**.

Setting the parameter **psfphotometryenabled** to *yes* would make this task to PSF-fit the 2D-image profiles of those sources that have close neighbours (which mean that the point-spread functions of these neighbours overlap). The photometry of these sources would then be based on the fitting parameters. With this feature enabled, an additional source-list table will be generated, containing sources with their photometry based on PSF-fitting. Such a fitting, however, consumes a large amount of computing time, so the use of the parameter **psfphotometryenabled** must be taken with certain care.



6 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.

image file is not 2D (*fatal*)

different binning in x and y axes (*fatal*)

Could not allocate memory for: *name* (*fatal*)

Memory allocation for the named allocatable array failed

WrongFilter (*fatal*)

Filter in not one of White, WHITE, V, B, U, UVW1, UVW2, UVM2, GRISM1, GRISM2 or MAGNIF

Failed to release memory for: *name* (*warning*)

corrective action: Memory deallocation for the named allocatable array failed

7 Input Files

One **OM** product image file is required, which has to be one of the following:

1. Product FITS image file (output from OMMODMAP)
2. Mosaiced sky-image produced by **ommosaic** from the mosaicing of two or more **omatt** product sky images.
3. Mosaiced unrotated-image produced by **ommosaic** from the mosaicing of two or more **ommodmap** images.

The OM priority-window data auxiliary file may also be needed for images produced by pre 5.4 sas.

8 Output Files

1. Intermediate OM OSW source list
2. Background-image file (optional)
3. Source number image (optional)
4. ds9 region file of source list (optional)



9 Output source-list file

The source-list file produced by **omdetect** contains a list of the positions of all the detected sources, the parameterizations of their shapes, their extracted count-rates, and "corrected count-rates" which contain some of the corrections that are needed to convert the raw rates to the rates that can be used to compute instrumental magnitudes. Fully corrected count-rates and instrumental magnitudes are obtained by running **ommag**, with its parameter "**set**" set to the **omdetect** source-list.

The source-list file is in **FITS** file format and contains a **SRCLIST** table with the columns shown in the following table.



COLUMN NAME	KIND	UNITS	DESCRIPTION
SRCNUM	INT 32	none	Source number - order of detection
XPOS	FLOAT	pixels	x-coordinate of centroided source centre
YPOS	FLOAT	pixels	y-coordinate of centroided source centre
POSERR	FLOAT	pixels	error in x and y centroided position
RA	DOUBLE	degrees	RA (J2000) mosaiced sky-images only
DEC	DOUBLE	degrees	DEC (J2000) mosaiced sky-images only
RA_CORR	DOUBLE	degrees	RA (J2000) corrected for the boresight offset
DEC_CORR	DOUBLE	degrees	DEC (J2000) corrected for the boresight offset
LII	DOUBLE	degrees	Galactic longitude
BII	DOUBLE	degrees	Galactic latitude
RATE	FLOAT	counts/sec	Total measured source count-rate (after background subtraction) within source extraction aperture (default aperture=12 un-binned pixels, but a smaller one may have been used), divided by exposure time.
RATE_ERR	FLOAT	counts/sec	Error in the raw count-rate
BACKGROUND_RATE	FLOAT	counts/sec	Total estimated background count-rate (scaled by area ratio, from the background annulus) within default source extraction aperture, divided by exposure time
SIGNIFICANCE	FLOAT	none	source counts / sqrt(source + background counts) i.e. signal-to-noise ratio, where the counts refer to the raw counts, before any corrections have been applied.
CORR_RATE	FLOAT	counts/sec	Background subtracted source count rate after applying PSF (if necessary) and coincidence-loss corrections.
CORR_RATE_ERR	FLOAT	counts/sec	Error in the corrected rate
CORR_BACKGROUND_RATE	FLOAT	counts/sec	BACKGROUND_RATE after correcting for coincidence losses.
PSF1_CORR	FLOAT	%	Point-spread function correction applied to extrapolate the RATE value from the measurement aperture area to the default extraction area of 12 un-binned pixels
PSF2_CORR	FLOAT	%	Point-spread function correction applied to extrapolate the RATE value from the extraction area of 12 un-binned pixels to the CAL (photometric) radius
SPB_COILOSS	FLOAT	%	Coincidence-loss correction applied to the source plus background rate (ratio of the coincidence-loss corrected source + coincidence-loss corrected background count-rate to the raw source + raw background count-rate)
BK_COILOSS	FLOAT	%	Coincidence-loss correction applied to the raw-background rate (ratio of the coincidence-loss corrected background count-rate to the raw background count-rate)
MAG	FLOAT	mag	Instrumental magnitude
MAGERR	FLOAT	mag	Instrumental magnitude error
NOPIXELS	INT 32	none	Number of pixels contained by an extended source (EFLAG=1). Point-sources (EFLAG=0) will have this value set to zero.
APERTURE	FLOAT	unbinned pixels	For point sources (EFLAG=0), the aperture radius (unbinned pixels) used in the photometry. For extended-sources (EFLAG=1) this value will be set to NULL.



COLUMN NAME	KIND	UNITS	DESCRIPTION
FWHM_MAJ	FLOAT	arcsecs	The full-width-at-half-maximum of the semi-major axis of the ellipse describing the shape of the source (computed by moments - Set to NULL if a reliable value couldn't be computed.)
FWHM_MAJ_ERR	FLOAT	pixels	The computed error of the major-axis
FWHM_MIN	FLOAT	pixels	The full-width-at-half-maximum of the semi-minor axis of the ellipse describing the shape of the source (computed by moments - Set to NULL if a reliable value couldn't be computed.)
FWHM_MIN_ERR	FLOAT	arcsecs	The computed error of the minor-axis
PA	FLOAT	degrees	The position-angle (measured anti-clockwise from the positive x-axis) of the semi-major axis of the ellipse describing the shape of the source (computed from moments - Set to NULL if a reliable value couldn't be computed.)
PA_ERR	FLOAT	degrees	The computed error of the position angle
QFLAG	INT 16	none	Source quality flag - Please see below
CFLAG	INT 8	none	Source confusion flag - Please see below
EFLAG	INT 8	none	(source extension flag) Either 0 point-source (non-extended) or 1 extended.
SRC_ID	INT 32	none	Source ID number in the source-list file

9.1 Mosaiced images

Please note that for mosaiced sky-images no coincidence-loss corrections will be applied to the count-rates. Hence the accuracy of the photometry of sources with raw-count rates > 5 counts per second will start to decrease, as this neglected correction becomes larger. The units of the semi-major/minor axes will also be in arcsecs.

9.2 QUALITY FLAGS

Each source has an associated “quality flag”. The flag settings are shown in the following table.

BIT NUMBER	REASON	INTEGER VALUE	Region Colour
0	BAD PIXEL	1	red
1	READ-OUT STREAK	2	magenta
2	SMOKE-RING	4	yellow
3	SOURCE ON STAR-SPIKE	8	white
4	Mod-8 pattern	16	black
5	Source within central enhancement	32	red
6	Source lies near to a bright source	64	black
7	Near an edge	128	blue
8	point-source within extended source	256	blue
9	Weird source	512	red
10	Multiple exposure values within photometry aperture	1024	red



9.2.1 Region Colours

Omdetect creates a **ds9** region file if the parameter **regionfile** is set equal to some file name. When the input image is viewed using **ds9** the sources can be overlaid on the image by selecting the region file.

For each source, an ellipse is drawn in either green or cyan, if it is “point-like” or “extended”, respectively. The axes lengths and position angle of the ellipse are proportional to those computed for the source. The number of the source is also plotted. For each quality flag set for the source, another ellipse is also drawn, with slightly larger axes, with the corresponding colour given above for the flag.

Thus, when an image is displayed using **ds9** and the regions overlaid, a point-source may be surrounded by one or more rings, each one indicating that a particular quality flag has been set. Unfortunately, due to the small number of colours that **ds9** uses, some flag settings have the same colour.

In the case of source-detection on mosaiced sky-images, many of the quality-flags will **not** be set by **omdetect** (specifically bits 1, 2, 3 and 5 and bit 7 will not be set for the majority of sources). However, the colour-scheme above still applies, but the region file is less useful for seeing potential source-quality problems.

Please note that these flag values are summarized in the SRCLIST header of the source-list file.

9.2.2 Examples

1. Quality flag 1 - Source contains one or more bad pixels
2. Quality flag 3 - Source contains one or more bad pixels and lies on a read-out streak
3. Quality flag 7 - Source contains one or more bad pixels and lies on a read-out streak and lies within a “smoke-ring” region.

9.2.3 Notes on individual flags

1. **Bad Pixels** A point-source will have this flag set if any pixel within the photometry aperture, or any within the background annulus, has a corresponding pixel in the quality image with a non-zero value. Similarly, an extended source will have this flag set if any corresponding pixel in the quality image has a non-zero value. The quality image was added to the image file by **omcosflag**, and a “bad” pixel may either be a damaged or dead one.
2. **Weird source** Indicates that the source has an isolated extremely bright pixel.

9.2.4 Short-comings

Comprehensive source-flagging was introduced into **SAS 6.5** in 2005. Many of the flags are set by empirical nature and since they are based on a statistical sample of images they are not infallible to error in individual cases.

Here is a list of known shortcomings\problems-

1. **Mod-8 pattern**- occasionally a source that does not have any obvious mod-8 pattern around it gets flagged and vice versa.



2. **Source within central enhancement region** Currently **omdetect** uses a circular region with a fixed radius to flag sources. This means that occasionally some sources will not get flagged when they are clearly affected by the central light, and some may get flagged unnecessarily.
3. **SOURCE ON STAR SPIKE** Occasionally some sources may not be flagged when they clearly should be.

9.3 Example OM image and ds9 regions

Figure 1 shows an example of a **OM** image with the detected ds9 source regions- to enable the regions to be seen more clearly, only part of the image is shown.

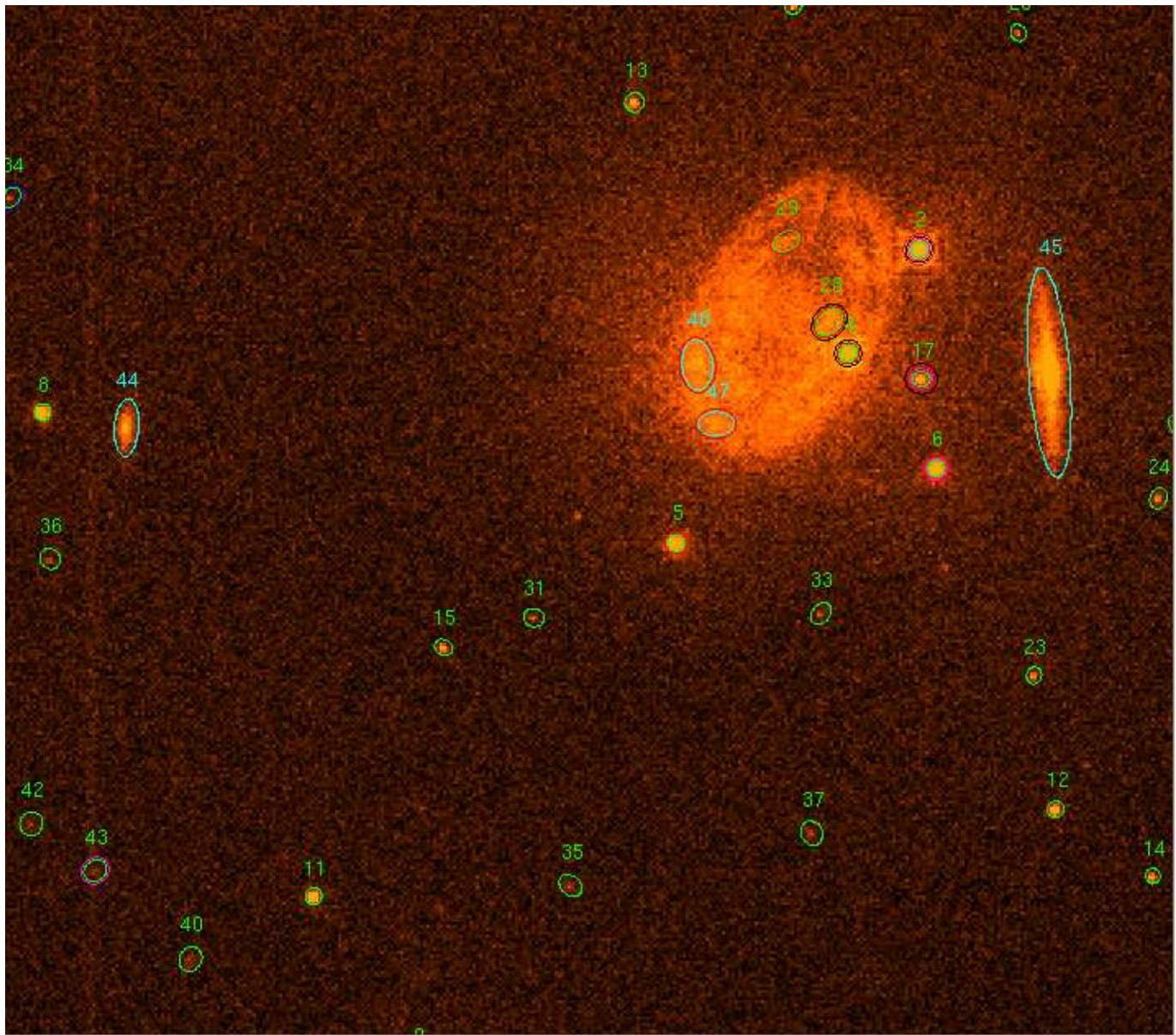


Figure 1: Example **OM** image showing the ds9 regions of the detected sources



The ds9 source regions in this image have their number in the source-list table written above them (in **green** if a **point-like** source or **cyan** if an **extended** source). Each source has one or more circles/ellipses centred on its computed coordinates. The innermost circle/ellipse identifies the source as either **point-like** (green) or **extended** (cyan).

Some features of these regions are now described:

- Source 34 has green and blue circles, identifying it as a point-source and lying close to an edge of the image.
- Sources 43, 6 and 17 have green and magenta circles, identifying them as a point-sources and lying close to a read-out streak (barely discernible- for source 43 the star responsible for the streak is off the image, for the other two sources source 2 is responsible).
- Many sources towards the upper-right of the image have red circles/ellipses indicating that they lie within the central-enhancement region.
- Sources 2 and 4 have green and black circles, indicating that they are point-sources and are very bright with likely modula-8 patterns.
- Sources 28 and 17 have also green and black circles, but in their case the black circles indicate that they lie close to a very bright source (source 4).

9.4 CONFUSION FLAGS

These flags are set for "point" sources by omdetect when the photometry is performed. The flags are stored in an 8-bit integer variable and the settings are:

BIT NUMBER	REASON	INTEGER VALUE
0	Point source has one or more sources within a radius of 6-12 unbinned pixels	1
1	Point source has one or more sources within a radius of 3-6 unbinned pixels	2

A source could have a confusion flag value of 3, indicating that both bits 0 and 1 are set.

10 Background algorithms

A parameter **backgroundmethod** determines which background method to be used with aperture photometry. There are 5 methods that can be used, and the default method of one which is the current SAS method is recommended since it is the one that has been used in the calibration. The other methods should be used with care.

10.1 Method 1- Current SAS algorithm (SAS 10)

The algorithm is based on the **sextractor** algorithm with a few modifications for low backgrounds (as in the UV).



10.2 Method 2- Mean

Using the mean of the pixel values to obtain an estimate of the background is risky because one or more source pixels can seriously effect its value.

10.3 Method 3- Median

The median can give a good estimate of the background value in moderately crowded fields (ie no more than 50 percent of the pixels used are contaminated by sources).

10.4 Method 4- Mode algorithm

Various photometry packages use the modal value for the background value, but most compute it from the empirical relationship between the mean, median, and mode, which appears to hold for unimodal curves of moderate asymmetry and is given by $\text{mean-mode} \approx 3(\text{mean-median})$. Bickel, D.R. 2002 ("Robust estimators of the mode and skewness of continuous data", Bickel, D.R., 2002, Computational Statistics and Data analysis Volume 39, Issue 2, pages 153-163) has presented a simple and robust algorithm to compute the mode. Our background-mode algorithm is based on his algorithm with a minor modification for low pixel values.

It can give a good estimate of the background value in crowded fiels (ie even if up to 75 percent of the pixels used are contaminated by sources).

10.5 Method 5- Using the computed background map

For this method the background map pixels within the pixels are used to compute a mean value for the background.

11 Algorithm

11.1 General Comments

Omdetect will find sources on an image using both a **peak-finding** and **connected-pixel** algorithm. The former will mainly detect "point" sources and the latter "extended" ones. The peak-finding algorithm is used first to try to locate all point sources, in order of decreasing brightness. This algorithm mainly overcomes problems with **modular-8** patterns around bright sources and scattered light features. When the detection algorithms have finished, photometry is performed on the sources and then the flagging algorithms are run.

11.2 Summary of the algorithm for IMAGING mode data

The following summarizes the way the program works for **IMAGING** mode data. More details of several of the processes mentioned are given later on. **FAST** mode data is processed in a simpler fashion- the



extended-source algorithms are not used.

1. Load the image
2. Load the pixel quality map image
3. Create a background image
4. Run initial point-source detection algorithm
5. Look for “smoke” rings
6. Look for scattered-light features, and mask-out the pixels of any that are found.
7. Look for largish extended sources
8. Look for smaller extended sources
9. Look for faint sources
10. Compute the photometry of the point-like sources
11. Check for faint point-sources near much brighter sources, and delete if necessary.
12. If any largish extended sources have been found re-compute the background image
13. Compute the photometry of the extended sources
14. Check for faint extended-sources near much brighter sources, and delete if necessary.
15. Set the source-flags
16. Create the source-list file

11.2.1 Mosaiced images

Mosaiced image-files are produced by **ommosaic** and contain three image extensions:

1. A count-rate image
2. An exposure map containing pixel-by-pixel effective exposure times.
3. A quality array containing position-dependent image quality-flags.

All three images are of equal size, pointing and orientation. In preparation for source-detection, mosaiced images undergo an additional and initial process where the count-rates are multiplied uniformly by the median exposure time across the whole image. This artificial step is performed only to enable accurate source detection and is accounted for during aperture-photometry.

12 Background image

The background image produced by **omdetect** is first constructed by using a median-smoothing algorithm that filters out bright pixels. When the sources have been found, the image is refined by masking-out source regions and interpolating across them.



13 Deficiencies

- If two point-sources are close together so that their images overlap, the detection algorithm for detecting point-sources will either fail to detect one or both. However, the detection algorithm for detecting extended sources will either pick up the missing object or detect both as a single extended source.
- An extended source may consist entirely of pixels below the detection threshold, and therefore will not be detected, even though the source is clearly visible on the image. The parameter **nsigma** controls the detection threshold and, if necessary, this can be decreased from its default value of 2 to the minimum allowed value of 1. **Please note that using this value may significantly increase the execution time..**

14 Source X and Y coordinates

The position of a source on the image (**XPOS** and **YPOS**) are referred to a coordinate origin in which the pixel at the far-left and bottom of the image has coordinates 1.0, 1.0. The source barycentre coordinates are computed using the first-order moments of the intensity profile.

14.1 Source RA and DEC coordinates (mosaiced sky-images only)

RA and **Dec** coordinates for sources detected from mosaiced sky-images will be computed from the **World Coordinate System** keywords stored within the image **FITS** header system.

15 Source shape parameters (semi-major/minor axes and position angle)

These parameters are computed using the second-order moments of the intensity profile.

16 Source classification

A source is classified as either “point-like” or “extended” by comparing its computed semi-major axis with that of the point-spread function, both having been computed using moments. A source is “extended” if the difference between its semi-major axis and that of the **PSF** is greater than 3 times the error in the semi-major axis of the source, otherwise it is classified as “point-like”.

17 Photometry

Please note that for grism data `omdetect` is only used to locate the positions of possible zero and first-order spectra- the SAS task `omgrism` is used to refine the positions and compute



the spectra. For Fast-mode data the photometric parameters computed by **omdetect** are not used- the photometry is recomputed using the SAS task **omlcbuild**.

If you think that the photometry of one or more sources could be improved (eg in crowded fields), then it is worth considering using either **omsource** or **omphotom** to recompute the photometry of one or more sources in the source-list produced by **omdetect**.

17.1 Point-like Sources (non-mosaiced images)

After the source-detection algorithms have finished, **omdetect** will compute raw count-rates for each detected source. For point-sources (ie those sources having widths consistent with those of the point-spread function) the count-rates are obtained by aperture photometry. For binned and unbinned images the aperture is 6 and 12 pixels, respectively. For a source with one or more neighbours that may be partly within the aperture, a smaller aperture will be used and its size is determined from the distance to its nearest neighbour. The actual aperture radii (unbinned pixels) that were used in the photometry are given in the column **APERTURE** of the **SRCLIST** table of the output source-list table.

The background is subtracted using the pixels within an annulus region. For the optical filters the inner and outer diameters of the annulus are 8 and 13 pixels for binned images, respectively, and 14 and 25 pixels for unbinned images, respectively. For the UV filters the inner and outer diameters of the annulus are 19.5 and 22 pixels for binned images, respectively, and 37 and 42 pixels for unbinned images, respectively. These values are stored as the fits keywords **INNBCKAN** and **OUTBCKAN** in the primary header of the output source-list file

When each point-source has had its photometry parameters computed, those with a significance of less than or equal to 10 sigma have their photometry re-computed using an aperture of 6 unbinned pixels, in order to reduce the errors due to the background noise.

These “raw” count-rates have to be corrected for coincidence-loss, dead-fraction and the time-dependent sensitivity degradation before instrumental magnitudes can be computed (using the task **ommag**). **Please note that the time-dependent sensitivity and dead-time corrections are applied by **ommag**.** The reference aperture through which XMM-OM coincidence-loss factors have been calibrated have a 12 pixel diameter. All raw count-rates obtained through reduced-size apertures are corrected to the reference aperture sizes using a calibrated PSF curves-of-growth, before coincidence-loss correction. The photometric zero-point calibration was performed on a 12 pixel diameter aperture in **U**, **B** and **V** filters, and a 35 pixel diameter one in **UVW2**, **UVM2** and **UVW1** filters. Therefore, **UV** point-source count-rates are extrapolated by curve-of-growth to 35 pixels before photometric calibration.

The following summarizes the procedure used to correct the source extracted count-rates for coincidence-losses-

- Extrapolate (if necessary) the source count-rate to the 12 unbinned pixel aperture, using the point-spread function for the filter.
- Extrapolate (if necessary) the background count-rate to the 12 unbinned pixel aperture, by using the scaling factor ($144 / \text{aperture radius}^2$).
- Apply a coincidence-loss correction to the total (source+background) count-rate.
- Apply a coincidence-loss correction to the background count-rate.
- Subtract the background coincidence-loss corrected count-rate from the coincidence-loss corrected total count-rate- this is the coincidence-corrected source-count rate.



Please note that these coincidence-loss corrections do not contain the factor $1/(1-\text{deadfraction})$.

The following further corrections will then be applied by **ommag-**

- If a UV filter, a further point-spread function correction is applied to extrapolate the source count-rate from 12 to 35 unbinned pixels.
- The “dead-time correction” - $1/(1-\text{deadtime})$
- The correction for the time-dependent sensitivity degradation.

17.2 Point-Spread Function Fitting

If the parameter **psfphotometryenabled** is set to true (default is false) then after the photometry of point-sources is performed (using an aperture) the photometry will be repeated using point-spread function fitting. **Please note, that the implementation of this algorithm in the task is still under development, so that the source photometry based on PSF-fitting might not be accurate enough; in the current task version the usage of this parameter is disabled.** Sources that have a computed raw-count rate greater than the value specified through the parameter **rawcounttrate** will be excluded from PSF-fitting, because a strong mod-8 pattern will make the fitting procedure unreliable.

In the case of **FAST** mode images, every source with a raw-count rate no greater than **maxrawcounttrate** will have its photometry re-computed, but in the case of **IMAGING** mode data sources with no close neighbours will also be excluded.

The algorithm is a development of the psf-fitting algorithm introduced into the task **omphotom** several years ago and works as follows:

The counts for a source are obtained by solving the following set of equations for each pixel included in the fit by least-squares

$$c(i, j) = \text{background} + \sum_{k=1}^n a_k \text{PSF}(\Delta x_k, \Delta y_k) ,$$

where, the background is either the constant term in the least-squares fit or a fixed value, n the number of sources are included in the fit, a_k is the central height of the fitted PSF source k and $\Delta x_k, \Delta y_k$ are the distances from the centre of source k to the image point (i, j) in the x and y directions, respectively. All pixels within a radius of 7 unbinned pixels from the centre of a source are included in the fit.

A discrete PSF function is used based on the PSF image obtained through the **CAL** routine

CAL_omGetPSFmap.

Interpolation is used to compute the value of the PSF at any $\Delta x, \Delta y$, distance from the centre.

The total raw counts of a source k is then equal to the value of a_k multiplied by the total counts in the PSF image within a radius of 12 unbinned pixel from it centre, which has been normalized to a central height of 1. The corrected count-rate is computed from this raw count-rate in an identical way to that for aperture photometry (no aperture correction is needed).

Since the width of the PSF is narrow, source centroiding errors can have a significant effect on the computed raw-count rates, and to reduce this error the algorithm “fits” the source coordinates by allowing



the positions of all the sources to vary within strict bounds. For each set of trial positions, the least-squares fit is performed and the set of positions that gives the best least-squares fit are used- the source being fitted has its coordinates replaced with the best fit values. All sources (up to a maximum of 9) within a radius of 26 unbinned pixels, are included in the fit- each one is used in order of increasing distance from the central one.

The function that is minimized carries out the least-squares fit just discussed, using the current coordinates of each source, and the function value is the weighted sum of squares of the residuals. Obviously, the CPU time increases rapidly with the number of sources included in the fit. The minimization algorithm is based on that described in J, Canto, S. Curiel and E. Martinez-Gomez, 2009, Astronomy and Astrophysics, 501, 1259. "A simple algorithm for optimization and model fitting: AGA (asexual genetic algorithm)". Other minimization routines have been considered but this one is particularly simple and robust, not requiring derivatives to be calculated, allowing upper and lower bounds on the variables and not being fooled by local minima.

The algorithm is designed for robustness and since the minimization algorithm is relatively slow compared to more sophisticated but less reliable methods, performing psf photometry on a large source-list could take a considerable amount of time.

When a source has its photometry re-computed with psf photometry, some of the computed values such as the raw-count rate are compared to the aperture-photometry values.

17.2.1 Weighting of the equations

The weighting factor for each pixel used in the least-squares fit is given by $1/\sigma$, where *sigma* is the pixel variance ($\sqrt{\text{pixelvalue}}$).

17.3 Extended Sources (non-mosaiced images)

Each **extended source** has a group of connected pixels associated with it, each one having a value greater than a minimum height above the corresponding background image pixel. These pixels can be seen by looking at the level image map (**parameter levelimage**) and are used to compute the source count-rate through the following steps:

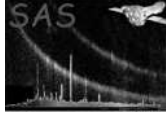
- 1) The quantities the **total raw pixel counts**, **total corrected raw pixel counts** and the **total raw background counts** are initialised to zero.

Loop through each source pixel and

- a) Each image pixel is treated as a point-source and a 12 pixel (unbinned) aperture is centred on it. The **total raw pixel counts** is incremented by the image pixel value.
- b) The counts within the aperture are obtained and a **coincidence-loss** factor is computed.
- c) The pixel value is multiplied by the **coincidence-loss** correction and this value is added to the **total corrected pixel counts**.
- d) The **total raw background counts** is incremented by the value of the corresponding background-image pixel.

End of looping through each source pixel.

The **raw count-rate** is computed by subtracting the **total raw background counts** from the **total raw pixel counts** and dividing by the exposure counts. The quantity **corrected background**



counts is computed by applying a coincidence-loss correction to the **total raw background counts**, using the mean **background-counts per pixel** and a 12 pixel (unbinned) aperture. This correction is generally negligible. The **corrected count-rate** is computed by subtracting the **corrected background counts** from the **total corrected pixel counts** and divided by the exposure.

17.4 Point-like\extended Sources using an exposure image

Each pixel from the input image will have been divided by the corresponding exposure image pixel. **Omdetect** will then have multiplied each pixel by the median exposure value from the exposure image. This is done because the source-detection\background algorithms do not work very well with the original image.

In the following formulae, the term $\frac{1}{expConstant}$ converts the image from counts back to counts/second (ie the original image).

The counts and count-rates are computed using the following formulae-

The total counts (TC) within the aperture is given by

$$TC = \frac{1}{expConstant} \times \sum_{i=il}^{iu} \sum_{j=jl}^{ju} image(i, j) \times area(i, j) \times exposure(i, j),$$

For “**point-like**” sources, the indices **il**, **iu**, **jl** and **ju** encompass all the pixels within the aperture centred on the source, whilst for “**extended**” ones **il**, **iu**, **jl** and **ju** encompass all the pixels assigned to the source- only a pixel within the box that is actually part of the source is used in the following summations. **area(i, j)**, is the fractional area of the pixel within the aperture (0- completely outside, 1- completely inside, for extended sources it is always 1), **exposure(i, j)** is the value of the exposure (from the exposure image) and **expConstant** is the exposure value that all the image pixels have been multiplied by.

The total background counts (TBC) within the aperture is given by

$$TBC = \frac{1}{expConstant} \times \sum_{i=il}^{iu} \sum_{j=jl}^{ju} BCP \times area(i, j) \times exposure(i, j),$$

where **BCP** is the background counts per pixel. For “**point-like**” sources this value is computed using the pixels within a circular annulus centred on the source of inner and outer radii of 8 and 13 unbinned pixels, respectively. Image pixels in which the corresponding **quality** pixel is 1 (ie a bad pixel) are ignored. In any case where the number of such pixels is less than 100 (ie less than one third that would normally be contained within the annulus, the background image is used to obtain a background value using the same annulus). For “**extended**” sources the background image is used to subtract the background.

The total background-subtracted counts (TBSC) within the aperture is given by

$$TBSC = \frac{1}{expConstant} \times \sum_{i=il}^{iu} \sum_{j=jl}^{ju} (image(i, j) - BCP) \times area(i, j) \times (exposure(i, j)),$$

The total background-subtracted count-rate (TBSCR) within the aperture is given by

$$TBSCR = \frac{1}{expConstant} \times \sum_{i=il}^{iu} \sum_{j=jl}^{ju} (image(i, j) - BCP) \times area(i, j),$$



The total background-subtracted count-rate error squared (TBSCRES) within the aperture is given by

$$TBSCRES = \frac{1}{expConstant} \times \sum_{i=il}^{iu} \sum_{j=jl}^{ju} ((S(i, j) + 2 \times B(i, j)) \times area(i, j) / exposure(i, j)),$$

where $S(i, j)$ and $B(i, j)$ are the background-subtracted counts and background counts, respectively, at pixel(i, j).

The source significance (signal-to-noise) is given by

$$TBSC/\sqrt{TC}$$

The background-subtracted count-rate error (BSCRE) is given by

$$BSCRE = \sqrt{TBSCRES}$$

The effective exposure (EE) for the source is given by

$$EE = TBSC/TBSCR$$

17.4.1 Important notes

For the photometry conducted on mosaiced images, it is important to note the following-

- The value for the exposure in the output source-list file (**FITS keyword EXPOSURE**) will be set to 1.
- No coincidence-loss corrections will have been applied.

18 Source Flagging Algorithms

18.1 Quality Flags

A description of the algorithms used to set these flags is given in the documentation for **omqualitymap**-please refer to this for further details.

19 Future developments

References