Catching gamma ray bursts on the fly . . .

Swift UVOT Software Guide

M. Still, P. Boyd, S. Holland, R. O'Brien, M. Tripicco, R. Wiegand

NASA GSFC Swift Science Center
(swifthelp@athena.gsfc.nasa.gov)

Document history

Date	Version	Author	Comment
2004-Nov-19	1.0	Martin Still	Swift 1.0 pre-launch version. Software has not been tested on flight data. Recipes use data based on simulated telemetry,
2005-Mar-10	1.1	Martin Still	Swift 1.3 release version after feedback from the Science Team.

Table of Contents

Document history 2

Table of Contents 3

1. Introduction 5
    1.1 Structure of this document ... 5
    1.2 Software requirements ... 5
    1.3 Data and science modes ... 6
    1.4 Data archive and completeness ... 7
    1.5 Filenames and archive structure ... 9

2. UVOT Data Analysis Recipes 11
    2.1 Image mode data reduction ... 12
    2.2 Event mode data reduction ... 16
    2.3 Summing a series of images ... 19
    2.4 Creating region files ... 22
    2.5 Extracting source counts from an image ... 26
    2.6 Screening event data ... 29
    2.7 Extracting an image from event data ... 31
    2.8 Extracting a light curve from event data ... 32
    2.9 Extracting a spectrum from event data ... 33
    2.10 Light curve analysis ... 34
    2.11 Extract a grism spectrum ... 38
    2.12 Fitting grism data ... 41

3. Image Tools 45
    3.1 UVOTBADPIX ... 46
    3.2 UVOTMODMAP ... 50
    3.3 UVOTFLATFIELD ... 54
    3.4 SWIFTXFORM ... 56
    3.5 UVOTEXPMAP ... 61
    3.6 UVOTDETECT ... 65
    3.7 UVOT2PHA ... 70
    3.8 UVOTMAG ... 75
    3.9 UVOTIMGRISM ... 78
    3.10 UVOTRMFGEN ... 83
    3.11 UVOTMAGHIST ... 86
    3.12 UVOTIMSUM ... 90

4. Event Tools 93
4.1 UVOTSCREEN 94
4.2 UVOTEVGRISM..... 98

5. Glossary 102

6. References 103

1. Introduction

1.1 Structure of this document

This document describes the software, both developed and adopted, that we recommend for the reduction and analysis of UVOT data. In Chapter 1 we briefly describe the software requirements for your computer, the data formats and the structure of the Swift archive, although note that these items are all discussed in more detail on other documents.

The next two chapters describe the software from differing perspectives. Chapter 2 provides a set of "ABC" recipes that can be followed in sequence. The end result is to produce science quality products from raw spacecraft data. These recipes are far from exhaustive. Their aim is to provide a rough guide to UVOT data analysis, and should be used as the spring board for users who want to become familiar with software.

Chapters 3 and 4 are a more detailed look at the UVOT tools developed for this mission. It is envisioned that these sections will used for reference long after the user stops following the ABC recipes of Chapter 2 rigorously. They detail the full functionality and I/O of each tool.

An important note to make is that the philosophy of UVOT software development has been a conservative one. The goal has been convert the raw data into a form readily acceptable by standard data analysis tools as quickly as possible. We use these standard tools liberally in our examples from Chapter 2 but do not reproduce the documentation for these tools here. Instead URL links are provided to existing web material, wherever appropriate.

Please forward comments, suggestions and bug reports for this document and the UVOT software to the Swift Science Center at swifthelp@athena.gsfc.nasa.gov.

1.2 Software requirements

There are two major suites of software required to follow the steps in this guide. It is recommended that users download these packages and keep them up-to-date. In particular the HEADas package is expected to be updated on weekly timescales during the early phases of the mission:

1. HEASoft. This contains general and specific mission tools for past and current high-energy astrophysics space mission. Many of the generic tools for file viewing, imaging, photometry and spectroscopy will be used in our recipes. Download instructions can be found at http://heasarc.gsfc.nasa.gov/docs/software/lheasoft.

2. HEADas. This package contains Swift-specific tools, including the UVOT software described in Chapters 3 and 4.Download information will be appended when it is made available.

One further package, the image viewer ds9 (http://hea-www.harvard.edu/RD/ds9) will be required to follow the recipes precisely.

1.3 Data and science modes

All data files conform to the current OGIP (Office of Guest Investigator Programs) FITS (Flexible Image Transport System) standards and conventions. Details of these conventions can be found at:

http://heasarc.gsfc.nasa.gov/docs/heasarc/ofwg/ofwg_intro.html.

The file headers are often commented, both to give a brief description of each keyword, and, occasionally, slightly more detail is provided in '**COMMENT**' entries. The processing history of these files are recorded by the '**HISTORY**' keywords. Exposure data can be fully described using five criteria, which we summarize here.

1.3.1 Data mode

There are three basic data modes: Image, Event and Image&Event. Image mode provides a two-dimensional sky map with a start and stop time for the exposure, but no arrival times for individual events. Event mode provides position and time information for each recorded event. Image&Event mode provides both types of data from the same exposure. The two windows need not have the same size, location or binning. Data mode is the most basic criteria into which data is separated within the HEASARC archive. There are separate files for Image and Event data. The Image&Event mode data are split into their constituent parts and integrated into the separate Image and Event Level I products. Science users never see this process.

1.3.2 Filter

There are 11 slots in the filter wheel in front of the UVOT camera. The full list of filters and their character codes, used within the file name scheme is provided in Table 1.3.1. Filter type provides the secondary criteria with which to separate data in the archive. There will be one Level I file for every filter used for Image mode exposures during the observing sequence, and one file for every filter used during Event mode observations in the sequence.

Table 1.3.1: UVOT filter character codes.

Filter keyword[1]	Filter code[2]
U	uu
B	bb

V	vv
UVW1	w1
UVM2	m2
UVW2	w2
WHITE	wh
VGRISM	gv
UGRISM	gu
MAGNIFIER	mg
BLOCKED	bl
UNKNOWN[3]	un

[1]From the header keywords within the file.
[2]From the filename.
[3]The UNKNOWN filter type occurs when the filter wheel has got lost.

1.3.3 On-chip binning

UVOT images can be binned in four different configurations, 1x1, 2x2, 4x4 or 8x8, although only the GeNi image, which is transmitted through the TDRSS/GCN system, will ever be binned 8x8. Event mode data will always be binned 1x1.

1.3.4 Window size

The size of the science window, in raw pixels, can change at any time during the observing sequence.

1.3.5 Window location

The science window location can change at any point during the observing sequence.

1.4 Data archive and completeness

Publicly available Swift data can be downloaded from two separate locations. The main facility will be the HEASARC:

http://heasarc.gsfc.nasa.gov/docs/corp/data.html

An observing sequence will populate the archive once it has been fully telemetered to the ground and reprocessed at the SDC (Swift Data Center). For a brief period after a burst (~1 week), preliminary data will be available at the SDC repository:

http://swift.gsfc.nasa.gov/sdc

This is, of course, to meet the requirement that Swift data be publicly available as soon as possible. However, when using the repository, be aware that the entire observing sequence may not yet have been telemetered. Early versions of these files may differ from those that eventually populate the HEASARC archive at a later date.

The archive is populated by three separate levels of data product. Table 1.4.1 describes the distinction. Note that there is no external way to tell which product belongs to which level, but we provide some pointers in Sec 1.5.

Table 1.4.1: Data levels contained in the HEASARC archive

Level	Description
I	Raw data. Generally the scientist need start their analysis from Level I products only if there has been improvements in the telescope calibration since the most recent data reprocessing, or if non-standard data screening is intended. Images are stored in raw detector coordinates. Event data are unscreened and recorded in raw detector coordinates.
II	Reduced data. It is envisaged that most scientists will start their analysis from the Level II products. The UVOT reduction pipeline has been performed on images and event tables. Images are stored in sky coordinates (RA_{2000}, Dec_{2000}), although grism data is also stored in corrected detector coordinates. Exposure maps for each individual image are available. Columns containing corrected detector and sky coordinates have been populated. Event data has been screened for standard bad times, e.g., SAA passage, Earth limb avoidance.
III	High-level data products. Level III products are intended to be quick-and-dirty representations of many of the products that the scientist would routinely create during their analysis, e.g., time-averaged images, light curves, spectra etc. Since they are created non-interactively, they are not intended as publishable products but as useful guides to the scientist. The number and type of Level III products in the archive will be determined by the confidence in any non-interactive burst detection and/or the time since the burst. More Level III products will populate the archive at an early stage if the burst has been identified during standard pipeline processing. Products will take longer to arrive if interactive analysis is required to identify the burst.

1.5 Filenames and archive structure

The UVOT instrument provides a unique set of logistical problems for file formatting. The standard observing sequences for bursts are complex, yielding a large number of exposures, with a wide variety of science modes. Formatting schemes also require the flexibility to cope with alterations in the observing sequences uploaded after the start of the sequence and alterations in the science windows during an exposure. A further complication is provided by Swift telemetry constraints. Data is telemetered down in non-sequential order and sequences are not being completely telemetered down until days after the burst. The archiving scheme minimizes the number of data files and organizes the exposures sequentially. The caveat is that many files will contain image or event data with a variety of science windows (varying in position, size, binning and exposure time). In terms of obtaining accurate images and exposure times from the data, it is crucial that the scientist is aware of this issue and analyzes their data accordingly. Table 1.5.1 lists some of the common data files and their location. These can be opened and inspected by, e.g., the FITS viewer fv, which comes with the obligatory HEASoft software distribution.

Table 1.5.1: Selection of UVOT archive files. The characters in square brackets flags to which data level each product belongs.

Filename	Description
Directory: uvot/image	
sw00000001001uuu_rw.img	U images in raw units (pixels) [I]
sw00000001001ubb_rw.img	B images in raw units (pixels) [I]
sw00000001001uvv_rw.img	V images in raw units (pixels) [I]
sw00000001001uvv_dt.img	V images in detector units (mm) [II]
sw00000001001uvv_sk.img	V images in sky units (RA, Dec) [II]
sw00000001001uvv_ex.img	V images exposure maps [II]

Filename	Description
Directory: uvot/event	
sw00000001001uvvpo_uf.evt	Unscreened V filter event tables [I]
sw00000001001uvvpo_cl.evt	Screened V filter event tables [II]

Filename	Description
Directory: uvot/product	
sw00000001001u_sk.img	Co-added sky images (all filters) [III]
sw00000001001u_ex.img	Co-added exposure maps [III]
sw00000001001uvvskim.gif	V filter co-added gif sky image [III]
	GRB light curve derived from V Event data [III]
sw00000001001u.his	Magnitude history derived from Image data [III]
sw00000001001u.his	Catalogue of detected sources, arranged by filter [III]

Filename	Description
Directory: uvot/auxil	
sw00000001001sat.fits	Attitude history file [I[
sw00000001001sao.fits	Orbit and attitude filter data [I]

2. UVOT Data Analysis Recipes

This section provides a set of standard software recipes for reducing and analyzing UVOT science data. While not comprehensive or exhaustive, the spirit of these recipes is to:

1) Provide the software user with the ability to repeat these tasks on any of the data populating the HEASARC archive or SDC quicklook area.

2) Gain an understanding of the principles behind the software and data organization.

3) Obtain the confidence to move on to more advanced data analysis using software of the user's own choice.

The Recipes can be split into two parts. The first, covered in Sections 2.1 and 2.2, contain mostly software written specifically for the reduction of UVOT software from raw (level I) data to calibrated science-grade (Level II and III) products. The majority of these tools are only ever run in the Swift pipeline before data populates the quicklook area and archive. It is anticipated that the general user need only use these tools in cases where updates to the UVOT caldb make a significant impact on the level II and III products. In these cases the user should follow these recipes in order to create new science products.

The second set of recipes, Sections 2.3-2.12, comprise imaging, temporal and spectral analysis of UVOT level II and III science products. The philosophy of the UVOT pipeline software development has been to supply products that are, for the most part, readily readable by generic analysis software. These recipes reflect that, although they are biased toward the software developed at the HEASARC and contained in the HEASoft package (http://heasarc.gsfc.nasa.gov/docs/software/lheasoft). Since the installation of this package is a prerequisite for the Swifft HEAdas tools, the majority of the tools employed in these scripts are already installed on your machine. The exception is the image viewer ds9 (http://hea-www.harvard.edu/RD/ds9).

Note that while the contents of these scripts are topical for the work of Swift Burst Advocates, there exists a Burst Advocate Cookbook containing recipes specific to the unique tasks of Burst Advocacy.

2.1 Image mode data reduction

2.1.1 Synopsis

The purpose of this script is to recreate fully calibrated (level II) images from raw (level I) images. This is a necessary process if the UVOT calibration has changed significantly since the pipeline processing of the archived data. This will generally affect data collected in the first few months of operation, during which the caldb will evolve rapidly, or when data is extracted from the archive after some significant time since it was first stored.

2.1.2 Processing flowchart

Figure 2.1.1 represents the flow of data through the UVOT pipeline image chain. The recipe below follows this chain from generating the bad pixel maps to exposure map calculations. The level III portion of this flow diagram is covered in various individual recipes later in this Chapter.

- The orange boxes in the flow charts represent the execution of individual FTOOLs. Some are specific to UVOT data processing, e.g. UVOTBADPIX, while others are applicable to general Swift processing, e.g., SWIFTXFORM.

- The green boxes represent the input of calibration products into the FTOOL. These products populate the Swift CALDB. The caldb filenames are provided in the recipes and tool descriptions in Chapters 3-5.

- The blue boxes represent the science products. Level I products, light-blue, are input to the tools. Level II and Level III products, medium- and dark-blue respectively, can be either input or output. Filenames are provided in Chapters 3-5.

- White boxes refer to crucial intermediate files that are not archived.

2.1.3 Recipe

1) Create quality maps for each image exposure. Quality maps contain a raw array of flags associated with the goodness or badness of each detector pixel and are used to help minimize modulo 8 fixed-pattern noise from the images and build exposure maps.

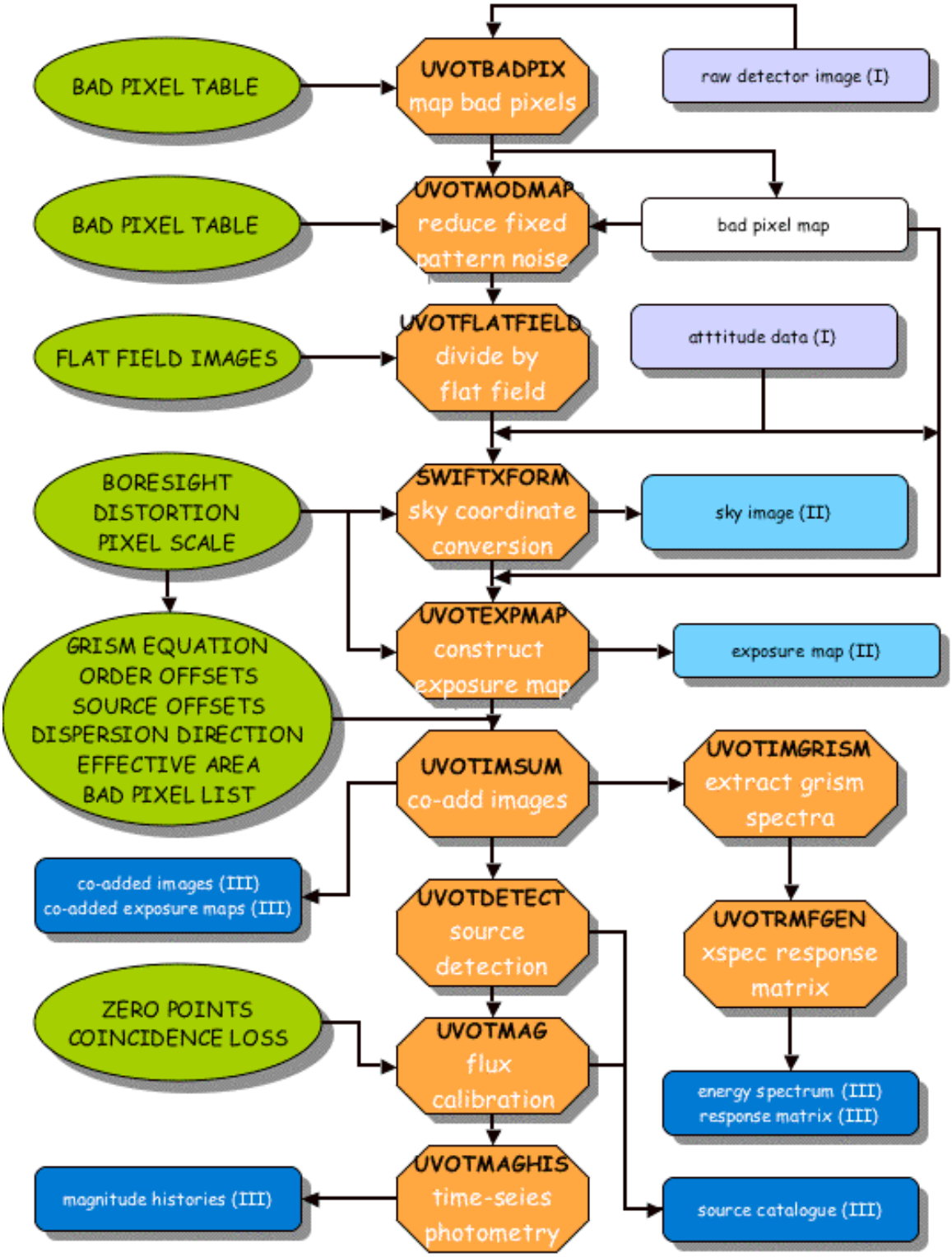

Figure 2.1.1: The UVOT imaging chain. Orange denotes a software tool, green a calibration product, light-blue a Level I science product, medium-blue a Level II product and dark-blue a Level III product. The white box is a file that is not archived.

```
Namibia> uvotbadpix infile=00072901259/uvot/image/sw00072901259ubb_rw.img.gz
outfile=quality_bb.img badpixlist=caldb compress=y
```

The "bb" in the output file name indicates the internal images were taken with the B filter. It is crucial that you keep track of the filter associated with image file. Repeat this step for all of the raw image files in the uvot/image directory, i.e., those files containing the string rw.img in their names. Ensure that each output file has a unique filename. Each output file will contain a series of images, mostly populated by zeros, except in the cases of cosmetically bad pixels, or pixels damaged by telemetry compression.

2) Reduce modulo 8 fixed-pattern noise in the images. Generally, raw image pixels are smaller than the original detector pixels by up to an integer factor of 8. This is because an on-board algorithm is able to centroid the splash of 107 events that a single photon produces. While this process improves the resolution of the detector, artificial fixed-pattern structure is introduced into the image. The nature of this structure is a bias in the photon position rather than an introduction of bogus counts. To some extent the bias can be removed statistically without any loss in photometric accuracy using the tool uvotmodmap:

```
Namibia> uvotmodmap
infile=00072901259/uvot/image/sw00072901259ubb_rw.img.gz
outfile=modmap_bb.img badpixfile=quality_bb.img mod8prod=no mod8file=none nsig=5
ncell=32 subimage=no
```

Repeat this step for all of the raw image files in the uvot/image directory,

3) Remove pixel-to-pixel variation in the image due to detector sensitivity using uvotflatfield:

```
Namibia> uvotflatfield infile=modmap_bb.img outfile=flatfield_bb.img flatfile=caldb
```

Repeat this step for all of the modulo 8 corrected image files in your working directory. Note that procedure can also be performed on the original raw data if you do not care to perform mod 8 correction on the images. However it would incorrect to perform the mod 8 correction after the flat field. Users are urged to follow these two steps in the correct order.

4) Convert the images from the raw coordinate system to a tangential projection of the sky:

```
Namibia> swiftxform infile=flatfield_bb.img outfile=sw00072901259ubb_sk.img to=SKY
attfile=misc/sw00072901259sat.fits.gz teldeffile=caldb ra=23.3555 dec=-41.8234
method=AREA bitpix=-32
```

Repeat this step for every filter taken in image mode. Either the raw, mod 8 corrected, or flat-fielded images can be used as input to this tool, but uvotmodmap and uvotflatfield should not be used on data output from swiftxform.

5) Construct exposure maps for each image.

```
Namibia> uvotexpmap infile=sw00072901259ubb_sk.img
outfile=sw00072901259ubb_ex.img badpixfile=quality_bb.img teldeffile=caldb
attfile=misc/sw00072901259sat.fits.gz method=MEANFOV attdelta=5
```

Repeat this step for every file produced by swiftxform.

2.2 Event mode data reduction

2.2.1 Synopsis

The purpose of this script is recreate fully calibrated (level II) event files from raw (level I) files. This is a necessary process if the UVOT calibration has changed significantly since the pipeline processing of the archived data. This will generally affect data collected in the first few months of operation, during which the caldb will evolve rapidly, or when data is extracted from the archive after some significant time since it was first stored. Furthermore a reprocessing would be necessary if the pipeline event screening is deemed too stringent for a specific user's needs. For example it is conceivable that raw event mode data come in a compressed format when telemetry volumes are high. In certain cases it is possible for events to lose their time tags and are consequently flagged as bad and rejected by screening. Users interested only in imaging do not care about photon arrival times and have lost a potentially valuable data. These can be reclaimed by performing the event chain on the raw data, as demonstrated in the following receipe.

2.2.1 Processing flowchart

Figure 2.2.1 represents the flow of data through the UVOT pipeline event chain. The recipe below follows this chain from COORDINATORTO UVOTSCREEN. The level III portion of the chain is covered in individual recipes later in the Chapter. While the filter file constructed by the PREFILTER tool is vital component of the chain, it is generated in the Swift pipeline and archived with the data. It is unlikely that a user will need to re-run PREFILTER.

- The orange boxes in the flow charts represent the execution of individual FTOOLs. Some are specific to UVOT data processing, e.g. UVOTBADPIX, whie others are pre-existing generic FTOOLS, e.g., COORDINATOR.

- The green boxes represent the input of calibration products into the FTOOL. These products populate the Swift CALDB. The caldb filenames are provided in the recipes and tool descriptions in Chapters 3-5.

- The blue boxes represent the science products. Level I products, light-blue, are input to the tools. Level II and Level III products, medium- and dark-blue respectively, can be either input or output. Filenames are provided in.

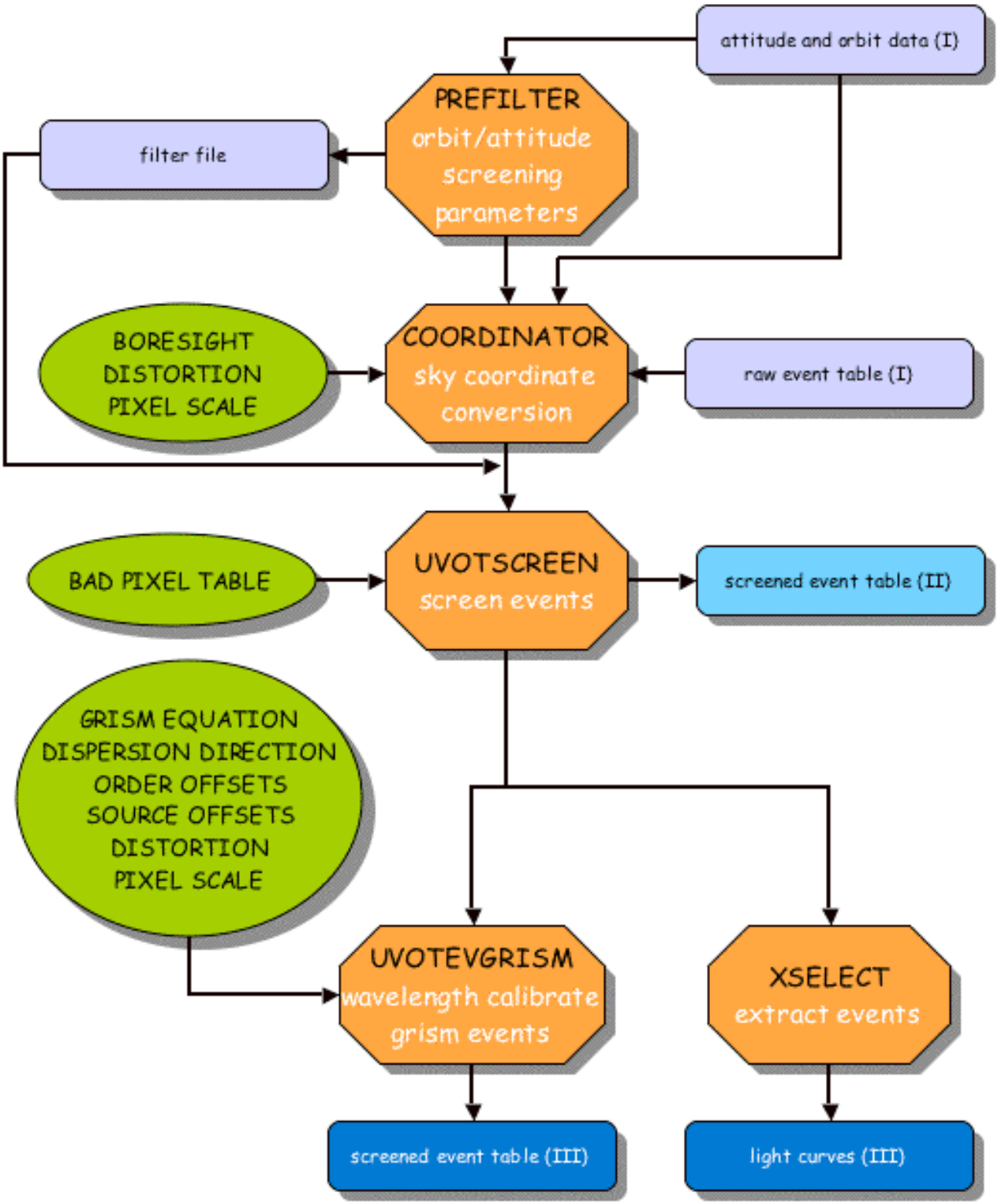

Figure 1.1.2: Flow of the UVOT event chain. Red boxes represent FTOOLs, green calibration products, light-blue Level I science products, medium-blue Level II science products and dark-blue Level III science products.

2.2.3 Recipe

1) Starting from the unscreened event tables in uvot/event, repopulate the DET and SKY table columns. These columns contain the coordinates of each event in physical (mm) units and over a tangential projection of the sky (RA_{2000} and Dec_{2000}).

```
Namibia> coordinator
eventfile=00072901259/uvot/event/sw00072901259ubbpo_uf.evt.gz.
eventext=EVENTS teldef=caldb attfile=00072901259/auxil/sw00072901259sat.fits.gz
aberration=n randomize=y seed=836 ra=23.3576 dec=-41.8234
```

This is a generic ftool and not a dedicated UVOT or Swift tool. Therefore we do not document it here. A detailed description of this tool and parameter use can be obtained by typing "fhelp coordinator" on your command line. Repeat this task for every raw event table in the uvot/event directory, e.g., those files including the string "pouf.evt". Users are recommended to change the value of 'seed' before each execution but maintain consistent values of ra and dec. While ra and dec are not critical parameters, they are used by imaging tools such as ds9 or ximage when reading these event tables to define the center of the displayed image. Note that this tool overwrites it's input table. If this is undesirable, make a copy of the raw files to work upon.

2) Screen the events using time-tagged quality information both internal and external to the file.

```
uvotscreen infile=uvot/event/sw00072901259ubbpo_uf.evt
attorbfile=00072901259/auxil/sw00072901259sao.fits.gz
outfile=sw00072901259ubbpo_cl.evt badpixfile=caldb aoexpr="ANG_DIST < 100. &&
ELV.> 10. && SAA == 0" evexpr="QUALITY == 0"
```

In order to keep events with corrupted time tags, as in our example at the top of the recipe, we should replace the evexpr argument with an OR statement exexpr="QUALITY == 0 || QUALITY == 32". Repeat this step for every event table produced by coordinator. A full list of parameters contained in the external attitude and orbit file, sw00072901259sao.fits.gz, and their definitions can be found by opening the header keywords of the file using FITS viewer fv, e.g. on the command line: "fv sw00072901259sao.fits.gz" and opening the header keyword extension.

2.3 Summing a series of images

2.3.1 Synopsis

UVOT images generally range in exposure time from 10s to 10^3s. To obtain the deepest observation possible from a sequential or non-sequential series of images, either of the same filter or different filters, it is necessary to co-add. This is non-trivial because both the pointing and rotation of each exposure will be slightly different, requiring a procedure of shifting and rebinning. We use the HEASoft image analysis package ximage to demonstrate how this can be achieved.

2.3.2 Recipe I

1) Call the ximage package from the command line, co-add,e.g., three images, write out the new image to a file, total_sk.img, and quit the package:

```
Namibia> ximage
[XIMAGE> read 00072901259/uvot/image/sw00072901259ubb_sk.img+1
[XIMAGE> save
[XIMAGE> read 00072901259/uvot/image/sw00072901259ubb_sk.img+5
[XIMAGE> sum
[XIMAGE> save
[XIMAGE> read/size=2632 00072901259/uvot/image/sw00072901259uvv_sk.img+7
[XIMAGE> sum
[XIMAGE> save
[XIMAGE> write total_sk.img
[XIMAGE> exit
```

Images contained within the same file, and images contained in separate files can be co-added. In this example we combine the first image extension in the B filter file, the fifth image extension in the same file and the first image extension in the V filter file.

ximage makes a default assumption that all images are smaller than 1024x1024 pixels in size. Any images larger than this will be automatically truncated unless the user supplies the size of the image in the read command. UVOT sky images can be significantly larger than this limit. For example, an unbinned, full-frame, raw image

rotated on the sky by 45º will result in a sky image of 2896x2896 pixels. In the example above we supply the size of the third image, which exceeds the ximage default size.

2) Repeat the above step for the exposure maps corresponding to each of the images:

```
Namibia> ximage
[XIMAGE> read 00072901259/uvot/image/sw00072901259ubb_ex.img+1
[XIMAGE> save
[XIMAGE> read 00072901259/uvot/image/sw00072901259ubb_ex.img+5
[XIMAGE> sum
[XIMAGE> save
[XIMAGE> read/size=2632 00072901259/uvot/image/sw00072901259uvv_ex.img+5
[XIMAGE> sum
[XIMAGE> save
[XIMAGE> write total_ex.img
[XIMAGE> exit
```

3) It is a trivial exercise to normalize the image by the exposure map, if desired, using the ftool farith:

```
Namibia> farith infil1=total_sk.img infil2=total_ex.img outfil=total_nm.img ops=div
null=y
```

2.3.3 Recipe II

Recipe I above can be a relatively typing-intensive and time consuming process, but provides the user with the freedom to co-add any number of images from a variety of different files. A faster method has been codedd as part of the Swift pipeline in order to provide the deepest possible images for each filter as level III product. The increase in speed is obtained because the tool is aware that all UVOT sky images are oriented East-North, whereas ximage makes no such assumption. Scientists can employ this faster tool with the caveat that all images in a specified file will be co-added by default.

1) Co-add all image extensions for a specified file:

```
uvotimsum infile=00072901259/uvot/image/sw00072901259ubb_sk.img
outfile=total_sk_bb.img method=GRID
```

2) Perform the same operation for the appropriate set of exposure maps:

```
uvotimsum infile=00072901259/uvot/image/sw00072901259ubb_ex.img
outfile=total_ex_bb.img method=GRID
```

Figure 2.3.1 illustrates typical co-added images and exposure maps. The variation of window size during the observing sequence is generally apparent in these images.

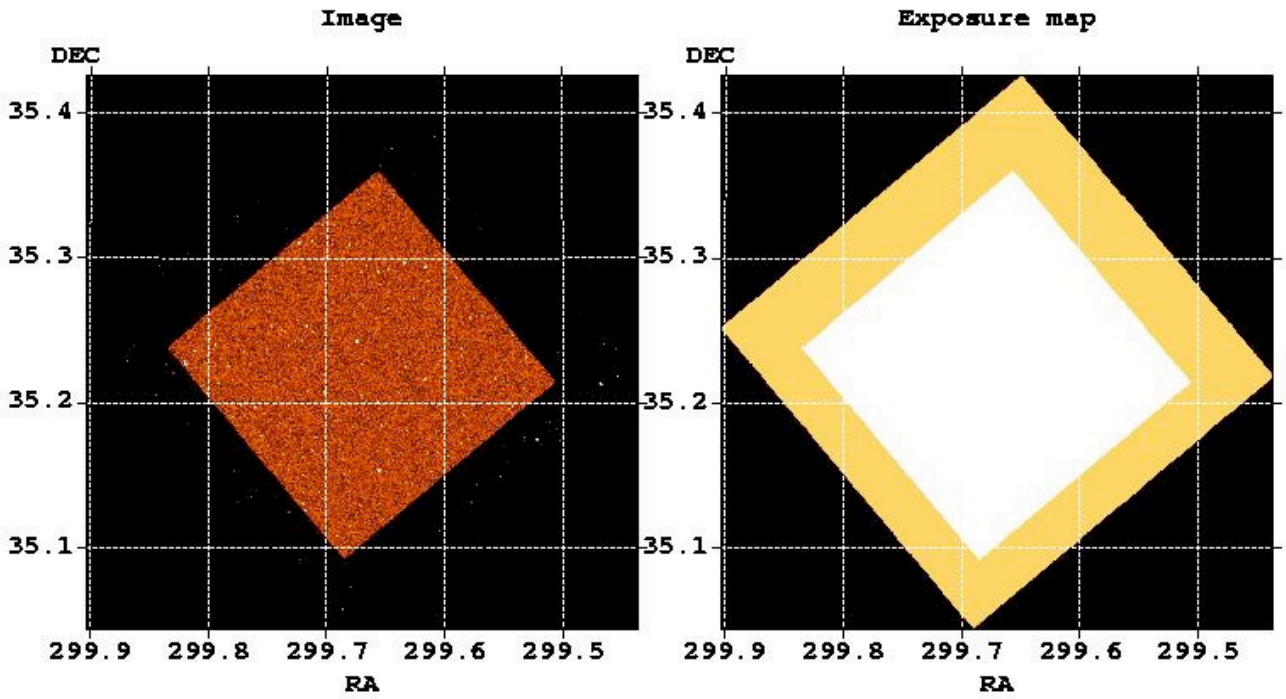

Figure 2.3.1: Examples of co-added images and the corresponding exposure map.

2.4 Creating region files

2.4.1 Synopsis

The general procedure for extracting source counts from an image, or producing light curves and spectra from, or screening, event tables involves the creation of source and background region files as a first step. Region files are formatted in ascii and can be constructed with variety methods, including composing them within an editor. For example here are some simple samples of source and background files:

Source-
fk5;circle(299.6563,35.15280,8.4")

fk5 is a keyword indicating that the region is defined in the J2000 coordinate system. circle indicates that the region is a circular shape with a center described by the first two numbers in units of decimal degrees. The third number is the radius of the circle in arcsec units.

Background-
fk5;-circle(299.6563,35.15280,8.4")
fk5;circle(299.6564,35.15280,20.3")

The background region file has a similar format. The minus sign before the first region indicates that the region is to be excluded. In effect this file represents an annulus with inner and outer radii of 8.4 and 20.3 arcsec. Region files can, of course, be more complicated than these examples. A full description of the region file format is provided at http://hea-www.harvard.edu/RD/ds9/ref/region.html#RegionFileFormat. A word of caution here, the xselect region logic differs from the ds9 logic. It is necessary to manually edit you region file if you intend to use it in the xselect environment. ds9 writes the exclusion zones first in the expression, whereas xselect expects them elsewhere. In our simple example above, the two zones must be swapped in order, e.g.:

fk5;circle(299.6564,35.15280,20.3")
fk5;-circle(299.6563,35.15280,8.4")

A description of the xselect logic can be found at
http://heasarc.gsfc.nasa.gov/docs/software/ftools/xselect/node56.html.

The following recipe describes the steps required to manually create region files using the image viewer ds9 (http://hea-www.harvard.edu/RD/ds9).

2.4.2 Recipe

1) Create a region file for a specific source, or set of sources. Note that given a file with multiple image extensions, ds9 will read only the first extension. Steps b-d below circumnavigate this problem for most practical purposes, but users have the option of extracting an individual image from a UVOT file before calling ds9 if they prefer using the ftool fcopy.

    i. Namibia> ds9 uvot/product/sw0002901259usk.img &
    ii. left-click **Region** (note there are two Region buttons on the ds9 GUI. In this case click on the one in the upmost tool bar).
    iii. Left-click **File Coordinate System**
    iv. Left-click **WCS**
    v. Left-click **Scale**
    vi. Left-click **log**
    vii. Using the mouse, place your cursor over a source
    viii. Click and leave your finger on the left mouse button
    ix. Drag the mouse away from the source, creating a circular region
    x. Release the mouse button
    xi. Left-click **Region**
    xii. Left-click **save**
    xiii. Supply the filename '`src.reg`' in the **Selection:** box and left-click **OK**

    Steps b-d are performed in order to write the region files in RA and Dec (decimal degree) rather than image pixel units. This enables the region file to be used across multiple images and files where the pointing, binning and size of images is not consistent.

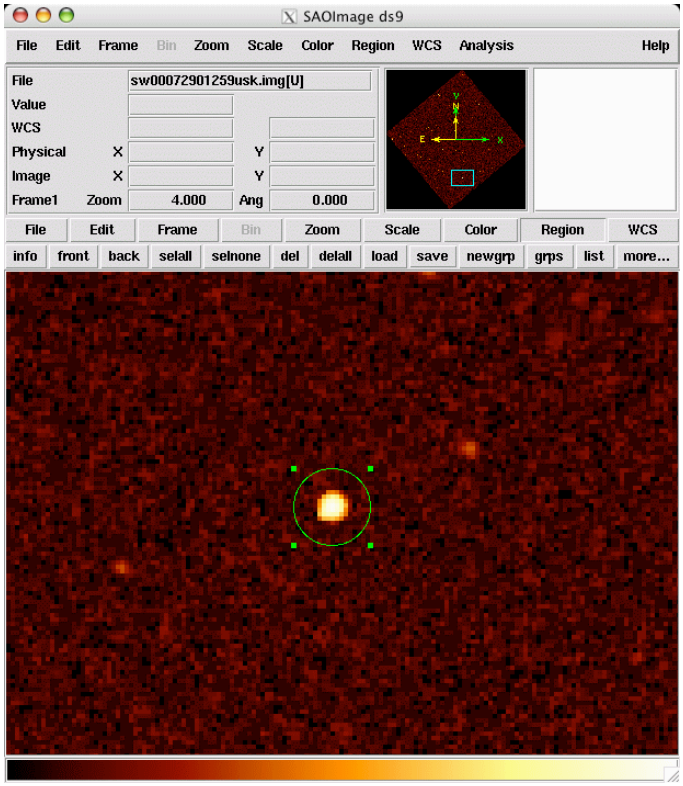

Figure 2.4.1: A source region from ds9.

2) Create a region file for the background round the source:

i. Left-click over the circular region created in 1.
ii. Left-click **Region**
iii. Left-click **Properties**
iv. Left-click **Exclude**
v. Left-click with the mouse pointing away from the source region.
vi. Left-click **Region**
vii. Left-click **Properties**
viii. Left-click **Include**
ix. Move the mouse over an arbitrary point of the image
x. Click and leave your finger on the left mouse button
xi. Drag the mouse away from the point, creating a circular region. This region should be larger than the source region.
xii. Release the mouse button.
xiii. Left-click on the background region.
xiv. Left-click and drag the background region over the source.
xv. Left-click **Region**
xvi. Left-click **save**
xvii. Supply the filename 'bkg.reg' in the **Selection:** box and left-click **OK**

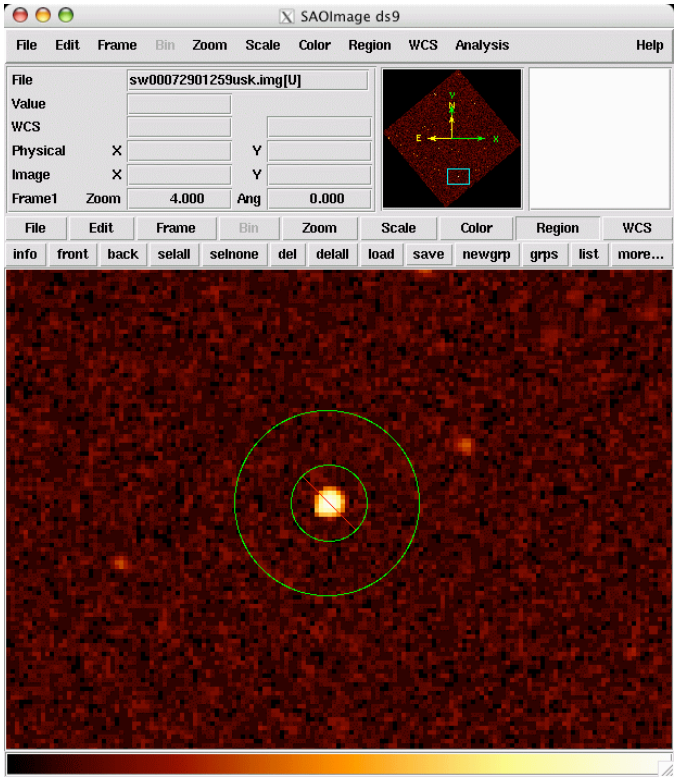

Figure 2.4.2: A background region from ds9.

2.5 Extracting source counts from an image

2.5.1 Synopsis

One of the fundamental properties provided by the UVOT detector is source brightness. This recipe illustrates a simple method of extracting source counts from an image, employing region files similar to those constructed in Section 2.4. The first recipe, Section 2.5.2, performs count extraction from a single generic image, whereas the second recipe, Section 2.5.3, extracts a series of source counts from the sequence of images in a UVOT FITS image file.

2.5.2 Recipe I

1) Read an image using ximage and sum the counts within both source and background region files:

```
Namibia> ximage
[XIMAGE> read/size=1400 00072901259/uvot/image/sw00072901259ubb_sk.img+3
[XIMAGE> cpd /xtk
[XIMAGE> disp/log
[XIMAGE> counts/regionfile=tot.reg
```

The counts command will draw the region on the image and produce the following screen output:

```
        Using MAP1
      Total      Area        Area        Average        Average
      (count)    (img-pix)   (det-pix)   (per img-pix)  (per det-pix)
#     1346       1892.0000   7568.0000   255.99471      63.998679

[XIMAGE> counts/regionfile=bkg.reg

        Using MAP1
      Total      Area        Area        Average        Average
      (count)    (img-pix)   (det-pix)   (per img-pix)  (per det-pix)
#     312        5856.0000   7568.0000   255.99471      63.998679
```

2) The 'Total' and 'Area' columns are used to calculate the source counts. In order to find the net source counts, C_{net}, we need to subtract the background counts, C_{bkg}, from the

total counts, C_{tot}, in the source region, but we must also compensate for the different areas inside each region, A_{tot} and A_{bkg}:

$$C_{net} = C_{tot} - \frac{A_{tot}}{A_{bkg}} C_{bkg}$$

In this example,

$$C_{net} = 1346 - \frac{1892}{5856} \times 312 = 1245.2$$

2.5.3 Recipe II

The procedure in Section 2.5.2 is wrapped into a script called uvotmaghist that calls all image extensions in a UVOT file, in sequence.

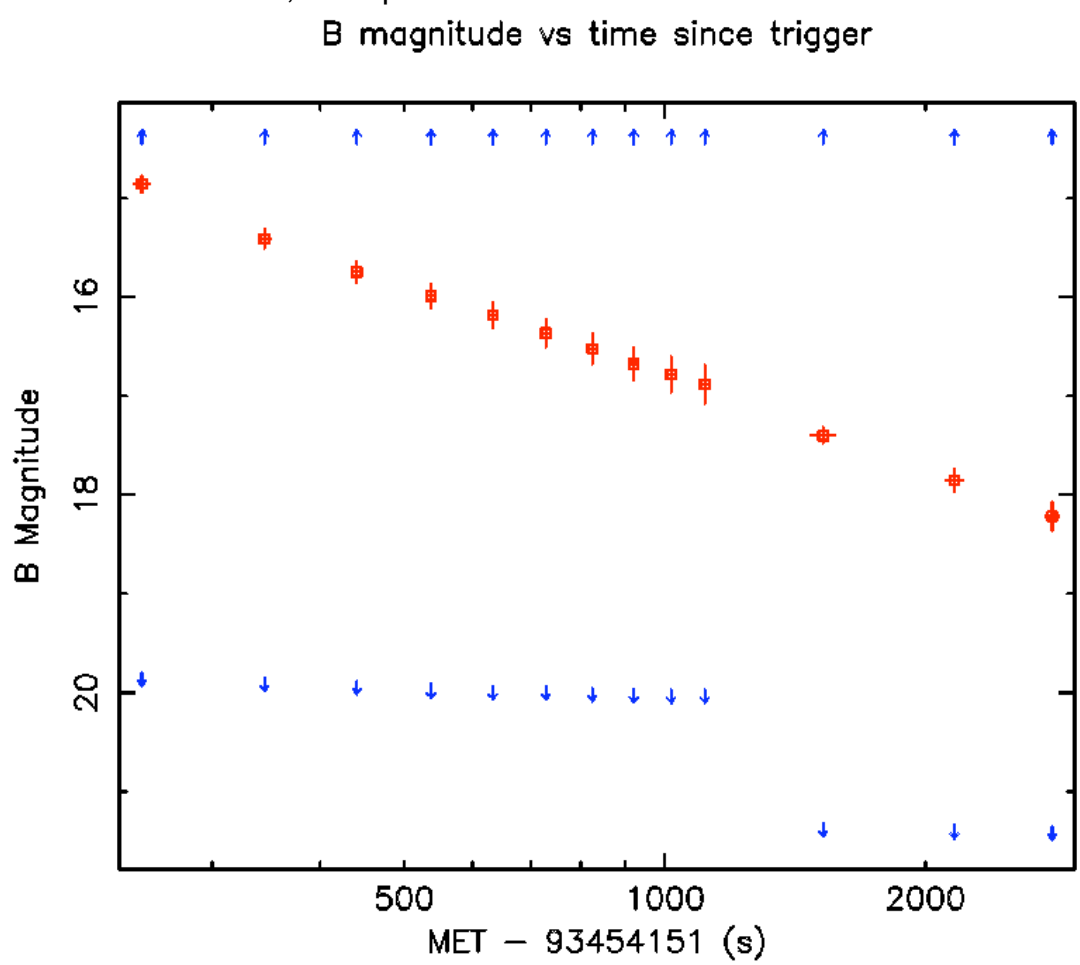

Figure 2.5.1: Example GIF output from uvotmaghist.

1) Call uvotmaghist, creating a FITS and GIF record of count rate over time.

```
Namibia> uvotmaghist infile=00072901259/uvot/image/sw00072901259ubb_sk.img
outfile=maghist.fit plotfile=maghist.gif zerofile=caldb coinfile=caldb ra=23.35
dec=41.823 srcas=3 bkgas=10
```

The output GIF file resembles Fig 2.5.1.

The advantage of recipe II is that it also converts source count rates to instrumental magnitudes and fluxes, storing these quantities in the output FITS table.

2.6 Screening event data

2.6.1 Synopsis

The general tool for event screening, developed as part of the HEASoft package, is xselect. This is a workhorse applied to data from multiple missions and has a wide variety of applications including screening and the extraction of light curves, image and spectra. A thorough understanding of this tool is recommended and can be found at http://heasarc.gsfc.nasa.gov/docs/software/ftools/xselect/xselect.html. Recipes 2.6-2.8 provide only the simplest examples of xselect's features. In the recipe below we will create a file of Good Time Intervals (GTIs) based on the angular distance between the telescope pointing and the bright Earth limb, e.g. to minimize Earth-glow effects in the data. Events occurring within the GTIs will be included in an output table, while those occurring outside the intervals will be thrown away.

2.6.2 Recipe

1) Create a GTI file based on data from the orbit filter file contained in the auxiliary directory of your data:

```
Namibia> maketime infile=00072901259/auxil/sw00072901259sao.fits.gz
outfile=br_earth.gti expr="BR_EARTH > 20." name=NAME value=VALUE time=TIME
compact=n prefr=1 postfr=1
```

2) Read an event table into xselect and screen using the GTI file:

```
Namibia> xselect
> Enter session name >[xsel]  bertie
xsel:ASCA > read events sw00072901259ubbpo_cl.evt
> Enter the Event file dir >[./] uvot/event
Got new mission: SWIFT
> Reset the mission ? >[yes] y
xsel:SWIFT-UVOT-EVENT >  filter time file br_earth.gti
xsel:SWIFT-UVOT-EVENT >  extract events
xsel:SWIFT-UVOT-EVENT > save events output.evt
Wrote events list to file output.evt
> Use filtered events as input data file ? >[no] n
xsel:SWIFT-UVOT-EVENT > exit
```

```
> Save this session? >[no] n
```

2.7 Extracting an image from event data

2.6.1 Synopsis

The general tool for event extraction, developed as part of the HEASoft package, is xselect. A thorough understanding of this tool is recommended and can be found at http://heasarc.gsfc.nasa.gov/docs/software/ftools/xselect/xselect.html. Recipes 2.6-2.8 provide only the simplest examples of xselect's features. In the recipe below we will assume that the event file has been screened by the user, and extract a sky image from the event list.

2.7.2 Recipe

1) Read an event table into xselect, extract, plot and save an image:

```
Namibia> xselect
> Enter session name >[xsel]  bertie
xsel:ASCA > read events sw00072901259ubbpo_cl.evt
> Enter the Event file dir >[./] uvot/event
Got new mission: SWIFT
> Reset the mission ? >[yes] y
xsel:SWIFT-UVOT-EVENT > set xycenter 2000.5 2000.5
xsel:SWIFT-UVOT-EVENT > set xysize 300 450
xsel:SWIFT-UVOT-EVENT > set xybinsize 2
xsel:SWIFT-UVOT-EVENT > extract image
xsel:SWIFT-UVOT-EVENT > plot image
xsel:SWIFT-UVOT-EVENT > save image output.img
Wrote image to file output.img
> Use filtered events as input data file ? >[no] n
xsel:SWIFT-UVOT-EVENT > exit
> Save this session? >[no] n
```

by setting xycenter, xysize and xybinsize, the user has the option to specify the center, dimensions and binning of the output image.

2.8 Extracting a light curve from event data

2.8.1 Synopsis

The general tool for light curve extraction, developed as part of the HEASoft package, is xselect. A thorough understanding of this tool is recommended and can be found at http://heasarc.gsfc.nasa.gov/docs/software/ftools/xselect/xselect.html. Recipes 2.6-2.8 provide only the simplest examples of xselect's features. In the recipe below we will assume that the event file has been screened by the user, and extract a source light curve from the event list. This recipe requires a region file, resembling the source file from Sec. 2.4.

2.8.2 Recipe

1) Read an event table into xselect, extract using a region file, plot and save a light curve:

```
Namibia> xselect
> Enter session name >[xsel]  bertie
xsel:ASCA > read events sw00072901259ubbpo_cl.evt
> Enter the Event file dir >[./] uvot/event
Got new mission: SWIFT
> Reset the mission ? >[yes] y
xsel:SWIFT-UVOT-EVENT > filter region "ds9.reg"
xsel:SWIFT-UVOT-EVENT > extract curve
xsel:SWIFT-UVOT-EVENT > plot curve
xsel:SWIFT-UVOT-EVENT > save curve output.lc
Wrote light curve to file output.lc
> Use filtered events as input data file ? >[no] n
xsel:SWIFT-UVOT-EVENT > exit
> Save this session? >[no] n
```

2.9 Extracting a spectrum from event data

2.9.1 Synopsis

Grism data obtained in Event mode is particularly useful for measuring spectral evolution over time. in the recipe below we will assume that the event file has been screened for time constraints by the user, extract an energy spectrum from the resulting event list. A pre-requisite for this recipe is to have passed the event file through the UVOT tool uvotevgrism.

2.9.2 Recipe

1) Create a binned spectrum over wavelength, with bin size of 20Å:

```
Namibia> fhisto
infile=00072901259/uvot/event/sw00072901259uvupo_cl_time1.evt+1
outfile=spectrum_time1.fits column=WAVELENGTH binsz=20
```

Note that this recipe provides a spectrum of counts, calibrated over wavelength. This is, of course suitable for comparing count rates at different epochs. If you have a requirement to flux the data, or fit spectral model in e.g., xspec, the correct procedure would be to extract an image from the event table in DETX, DETY units, and then pass the image through the uvotimgrism and uvotrmfgen tools (Sec 2.11).

2.10 Light curve analysis

2.10.1 Synopsis

The xronos package (http://heasarc.gsfc.nasa.gov/docs/xanadu/xronos/xronos.html) has been installed on your machine as part of the LHEASoft distribution. It contains applications dedicated to the analysis of timing data, such as the light curve produced by xselect in Sec. 2.7. Table 2.10.1 lists a subset of these applications. More details and applications can be found at the URL above.

Table 2.10.1: xronos applications for time series analysis.

Application	Description
autocor	Auto-correlation
crosscor	Cross-correlation of two time series
earth2sun	Barycentric correction
efold	Epoch folding
efsearch	χ^2 period search
flc2ascii	FITS-to-ASCII conversion
lcmath	Arithmetic between two time series
lcstats	Time series statistics
lcurve	Creates binned light curves and colour diagrams
powspec	Power density analysis

The recipe below provides a simple example of rebinning and plotting a UVOT light curve, which was originally extracted from Event mode data using xselect.

2.10.2 Recipe I

1) Load the unbinned light curve into the lcurve application:

```
Namibia> lcurve

lcurve 1.0 (xronos5.21)

Number of time series for this task[1] 1
Ser. 1 filename +options (or @file of filenames +options)[] sw00000001001ubb_sr.lc
Series 1 file    1:sw00000001001ubbsr.lc

 Selected FITS extensions: 1 - RATE TABLE;

 Source ........... Safe Pointing 1     Start Time (d) .... 12991 15:34:19.567
 FITS Extension ....  1 - `RATE      `   Stop Time (d) ..... 12991 16:17:38.559
 No. of Rows .......       35665       Bin Time (s) ......   0.1100E-01
 Right Ascension ... 23.35             Internal time sys.. Converted to TJD
 Declination ....... -41.82305556       Experiment ........ SWIFT    UVOTA
 Filter ........... B
 Corrections applied: Vignetting - No ; Deadtime - No ; Bkgd - No ; Clock - No

 Selected Columns:  1- Time;  2- Y-axis;  3- Y-error;  4- Fractional exposure;

 File contains binned data.

Name of the window file ('-' for default window)[-] -

 Expected Start ... 12991.64883758009  (days)     15:34:19:567  (h:m:s:ms)
 Expected Stop .... 12991.67891850347  (days)     16:17:38:559  (h:m:s:ms)

 Minimum Newbin Time   0.11000000E-01  (s)
 for Maximum Newbin No..         236272

 Default Newbin Time is:   5.0821114    (s) (to have 1 Intv. of     512 Newbins)
 Type INDEF to accept the default value
```

The tool has read the light curve and provided some statistics to your shell such as the source name and position, the instrument name and filter, type of data, start and stop times, the time system, the amount of data and it's binning.

2) Choose an new bin size:

```
Newbin Time or negative rebinning[100] 1.0

 Newbin Time ......     1.0000000      (s)
```

```
 Maximum Newbin No.          2599
Default Newbins per Interval are:      512
 (giving      6 Intervals of        512 Newbins each)
 Type INDEF to accept the default value
```

3) Segregate the data into a number of time intervals. This is particularly useful for plotting the evolution of colour-colour diagrams over the time interval, but has limited value in this example. The full set of data has 2599 1-s bins, so we plot them all together:

```
Number of Newbins/Interval[98] 2599
 Maximum of      1 Intvs. with       2599 Newbins of      1.00000    (s)
```

4) Define the output filename and the output plotting device:

```
Name of output file[default] output.flc
Do you want to plot your results?[yes] yes
Enter PGPLOT device[/XW] /XW

   2599 analysis results per interval

100% completed

 Intv   1   Start 12991 15:34:20
   Ser.1    Avg  9.940      Chisq  337.0     Var  14.50    Newbs.   405
        Min  0.000       Max  40.11   expVar  12.41     Bins  35665
PLT> q
Writing output file: output.flc
```

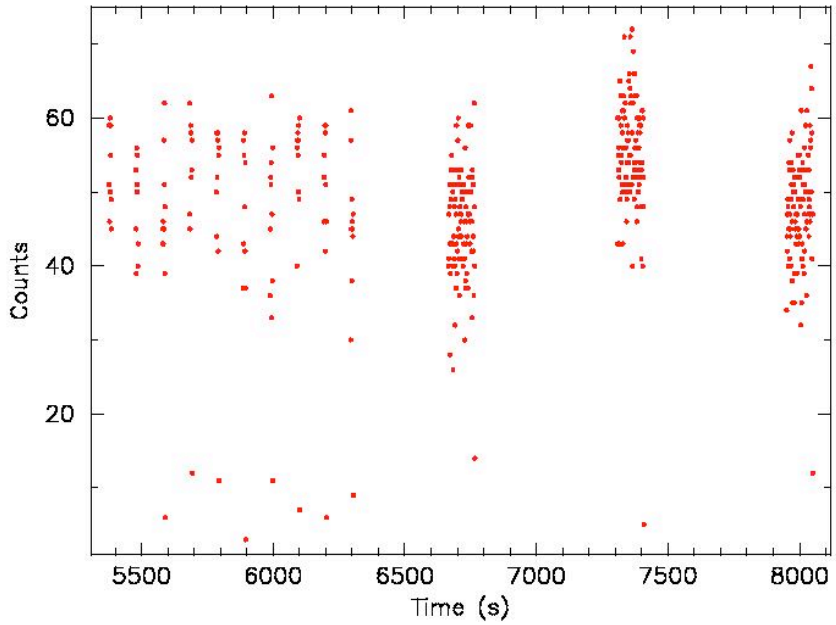

Figure 2.10.1: Time series output from the lcurve tool.

Before plotting the new light curve to the xserver, the tool provides a few statistics, such as the average count rate, minimum and maximum count rate values and variability measures. Fig 2.10.1 provide typical output.

2.10.3 Recipe II

UVOT data from different filter can never be simultaneous, therefore plotting colour information is not practical without choosing large time bins. However much more detailed colour analysis is possible if further time series data is loaded into the tool from e.g. the XRT or ground-based observatories.

1) In these instance the first input parameter asks or the number of time series used in the analysis:

```
Number of time series for this task[1]
```

In the previous example we used only one time series. But we can use up to three. A minimum of two time series are required to plot a colour ratio versus time, while three are required to plot a colour-colour diagram.

2) After the plotting device has been set, the tool will ask what type of plot is required.These are designated by a numeric character whose meaning depends on the number of time series used:

```
Enter PLOT style number (default=1)[]
```

For the two-series case:
1 = hardness ratio versus time
2 = intensity of both series versus time
3 = both intensity and harness ration versus time

For the three-series case:
1 = colour-colour diagram
3 = plot all three intensities versus time

2.11 Extract a grism spectrum

2.11.1 Synopsis

This recipe illustrates the method of extracting a calibrated grism spectrum from an image. Strictly speaking, this is a one-step process. The tool uvotimgrism extracts and calibrates the spectrum internally, producing a FITS table containing a fluxed spectrum in units of erg s^{-1} cm^{-2} $Å^{-1}$, over wavelength, in units of Å. However a powerful application of Swift data will be to perform panchromatic fits across UV, optical and X-ray energies. Therefore we have striven to male UVOT spectral data consistent with that expected from the xspec spectral fitting package (http://heasarc.gsfc.nasa.gov/docs/xanadu/xspec). This requires the raw spectrum and calibration to be separated into distinct files. Therefore the output from uvotimgrism will have two extension of spectral data, one containing a fully-calibrated spectrum, the other a raw spectrum. Users wishing to use xspec will have to perform a further task, creating a response matrix for the spectrum. It is important to note that the uvotimgrism tool requires an input image in detector (DET) coordinates rather than RA and Dec (SKY).

2.11.2 Recipe

1) Extract a spectrum from a DET grism image:

```
Namibia> uvotimgrism infile=sw00000001001ugv_sr.img+1
outfile=sw00000001001ugv_sr.pha backfile=sw00000001001ugv_bk.pha
badpixfile=badpix.img+1 wavefile=caldb areafile=caldb teldeffile=caldb ra=23.35
dec=-41.823 vang=206.7 vsrcwid=20 vbkgwid=50 vmin=2900 vmax=5500 nsigma=5
cleanup=y clobber=y history=y chatter=1
```

The output spectrum file will contain three extension, an image containing just the 0th and 1st order light of the source, plus the background regions chosen by the input parameters above. The image data has been rotated so that the X-axis is parallel with the 1st order dispersion direction, as in Fig 2.10.1.

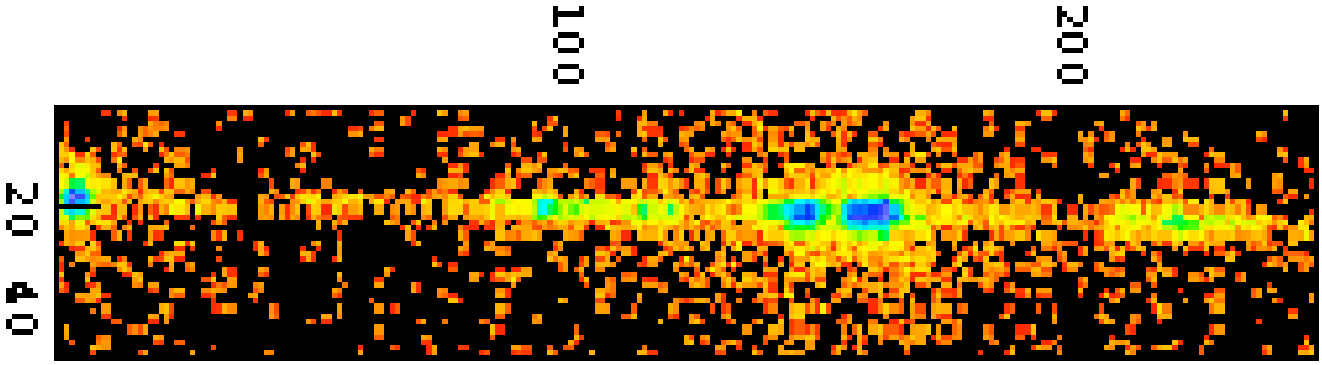

Figure 2.10.1: A typical image contained in the output from uvotimgrism.

Plotting the wavelength column in the third extension against counts, using, e.g., fv, yields a spectrum similar to that in Fig. 2.10.2:

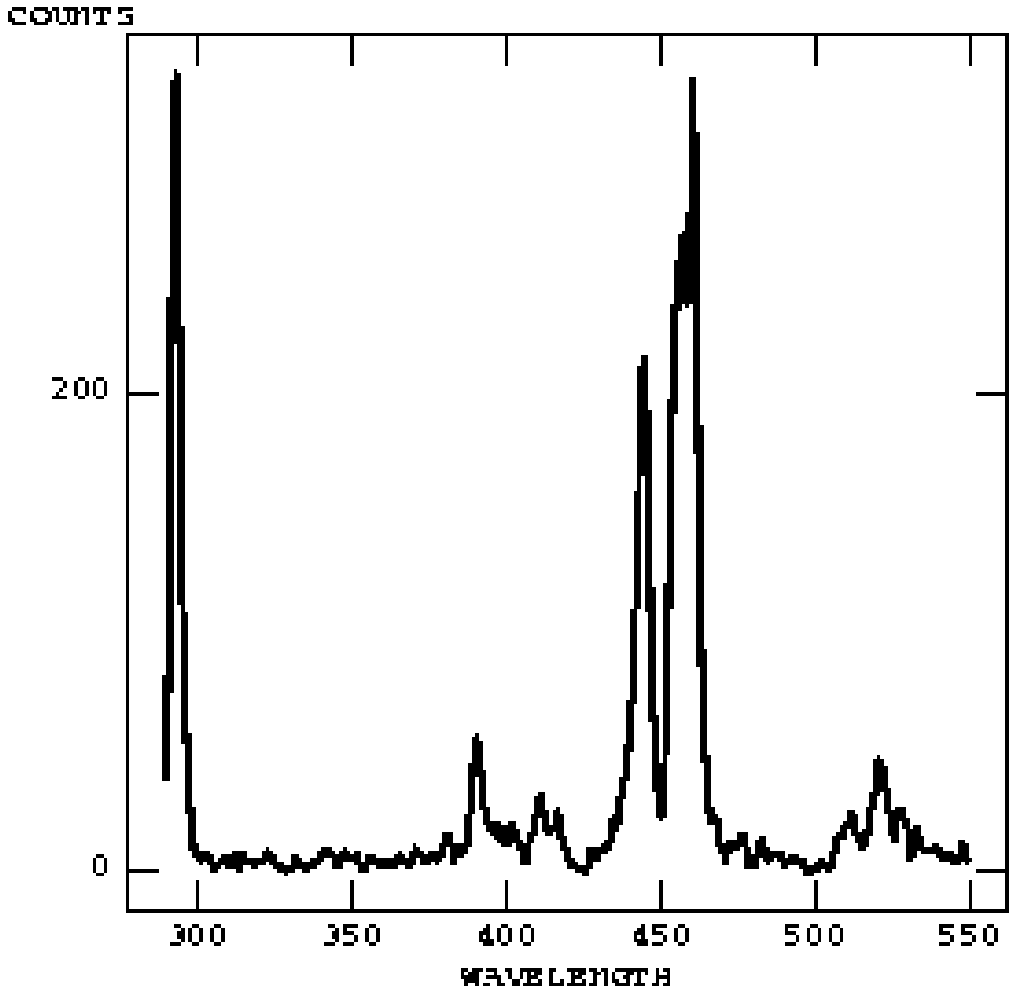

Figure 2.10.2: Spectrum output from uvotimgrism.

2) Construct a response matrix, containing wavelength and flux calibration data suitable for use in xspec using uvorfmrgen:

```
Namibia> uvotrmfgen spectrum=sw00000000100lugv_sr.pha
outfile=sw00000000100lugv_sr.rsp areafile=caldb lsfile=caldb
```

2.12 Fitting grism data

2.12.1 Synopsis

This recipe illustrates a simple fitting procedure using xspec. The xspec package has a large number of useful features that we will not discuss here, but needless to say, the user will find reading the xspec manual a worthwhile activity before ploughing too far into this topic:

http://heasarc.gsfc.nasa.gov/docs/xanadu/xspec

2.12.1 Recipe

1) Start xspec and load grism source and background spectra and the response matrix:

```
Namibia> xspec

        Xspec 11.3.1   13:27:05 19-Nov-2004

 For documentation, notes, and fixes see  http://xspec.gsfc.nasa.gov/

 Plot device not set, use "cpd" to set it

 Type "help" or "?" for further information
XSPEC>data spectrum.pha
 Net count rate (cts/s) for file   1   2.834   +/-  6.3628E-02( 90.7% total)
  using background file...        background.pha
  1 data set is in use
XSPEC>resp spectrum.rmf
```

2) Ignore bad pixels:

```
XSPEC>ignore bad
```

3) Define a spectral model and provide some starting parameters. In this example we will combine three multiplicative models – a powerlaw (power), reddening (redden), and a neutral H edge, redshifted by *z* = 3 (zphabs):

```
XSPEC>mo zphabs*redden*power
 Model:  zphabs<1>*redden<2>( powerlaw<3> )
Input parameter value, delta, min, bot, top, and max values for ...
            1    0.001       0       0    1E+05    1E+06
1:zphabs:nH> 1
            0   -0.01       0       0      10       10
2:zphabs:redshift>1
          0.05    0.001       0       0      10       10
3:redden:E(B-V)> 0.1
            1     0.01      -3      -2       9       10
4:powerlaw:PhoIndex> 0.9
            1     0.01       0       0    1E+24    1E+24
5:powerlaw:norm> 0.1
 ---------------------------------------------------------------------
 ---------------------------------------------------------------------
 Model:  zphabs<1>*redden<2>( powerlaw<3> )
 Model Fit Model Component  Parameter  Unit     Value
 par   par comp
 1   1   1  zphabs    nH      10^22     1.00000    +/-  0.00000
  2   2   1  zphabs    redshift         1.00000    frozen
  3   3   2  redden    E(B-V)          5.000000E-02 +/-  0.00000
  4   4   3  powerlaw   PhoIndex         1.00000    +/-  0.00000
  5   5   3  powerlaw   norm           0.100000    +/-  0.00000
 ---------------------------------------------------------------------
 ---------------------------------------------------------------------
Chi-Squared =     33685.34    using   522 PHA bins.
Reduced chi-squared =     65.02961    for    518 degrees of freedom
Null hypothesis probability =  0.00
```

The information reported back to you after entering the initial fit parameters are the parameter values and χ^2 goodness-of-fit statistics. These will be updated every time you fit to the data.

4) Fit the data:

```
XSPEC>fit 100
```

Xspec now reports:

```
---------------------------------------------------------------------
 ---------------------------------------------------------------------
 Model:  zphabs<1>*redden<2>( powerlaw<3> )
 Model Fit Model Component  Parameter  Unit     Value
 par   par comp
  1   1   1  zphabs    nH      10^22     1.00000    frozen
```

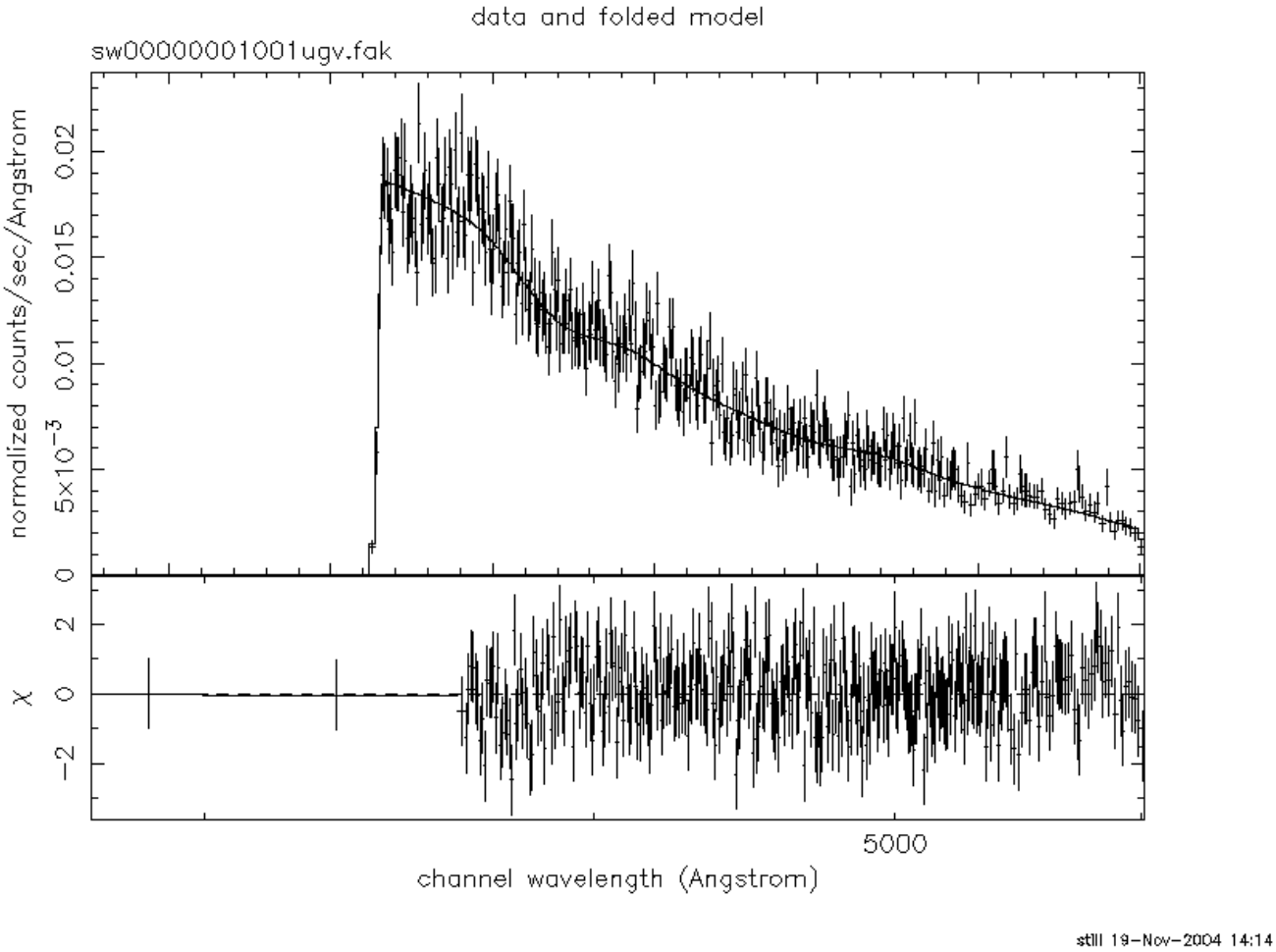

Figure 2.12.1: An idealized VGRISM spectrum with fit residuals.

```
2   2   1   zphabs    redshift        3.00346    +/-  0.106201E-02
3   3   2   redden    E(B-V)          2.280192E-02 +/-  0.224064
4   4   3   powerlaw  PhoIndex        1.06898    +/-  0.841269
5   5   3   powerlaw  norm            0.602656   +/-  3.73260
 ------------------------------------------------------------------
 ------------------------------------------------------------------
Chi-Squared =     564.6284    using   522 PHA bins.
Reduced chi-squared =    1.0914510    for    518 degrees of freedom
Null hypothesis probability =  1.00
```

Which according to the χ^2 statisitc is a good fit.

5) Uncertanties for each active parameter can be calculated thus:

```
XSPEC>uncer 2-5

Parameter   Confidence Range (    2.706)
2    3.00097      3.00439      ( -3.326416E-03,    9.632111E-05)
3    0.00000      2.122159E-02  (    0.00000   ,    2.122159E-02)
4   0.471290      1.25561      ( -0.691941   ,    9.237909E-02)
5   0.313522      8.59642      ( -4.709452E-03,    8.27819   )
```

The 2nd and 3rd columns contain the lower and upper 90% confidence limits.

6) Finally, set some plotting options and plot the data with fit, plus fit residials:

```
XSPEC>cpd /xs
XSPEC>setplot wavelength
XSPEC>setplot command log off
XSPEC>setplot rebin 5 100
XSPEC>plot data delchi
```

Some typical output can be found in Figure 2.12.1.

3. Image Tools

3.1 UVOTBADPIX

3.1.1 Updates

Table 3.1.1: UVOTBADPIX update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.1.2 Description

UVOTBADPIX creates *bad pixel maps* for the images. These are deposited in a level I FITS image file. A bad pixel map is an image of the same dimensions, binning and hardware window as the original data, but populated by flags that describe the quality of each pixel. The meaning of each flag is described in Table 3.1.1.

Most of the bad pixel cases result from chip defects on the instrument detector. Pixels in the Level I images are generally sub-sampled, for instance a pixel binning of 1x1 results in sub-sampling by a factor 8, so it is common to find bad pixels clustered in 8x8 arrays. Dead hardware pixels contain no charge whatsoever. Cold pixels contain charge at a reduced level, resulting in fewer counts than expected. Hot pixels contain more counts than expected, while flickering pixels can be bad during some intervals, while good at others. By the nature of the detector, hot pixels are not expected to occur. Cosmetic pixels are stored in the CALDB within a bad pixel table.

Compression damage may occur if the images were compressed on-board before being transmitted to Earth. The compression algorithm stores the difference between counts in consecutive pixels. If the difference is greater than a certain threshold, defined by flight software, then the pixel value is clipped to reduce the telemetry rate. The telemetry-to-fits software that runs in the Swift pipeline, UVOT2FITS, converts telemetry back to raw counts using reverse-compression, however the software has no means of reconstructing the correct

number of counts in clipped pixels. To combat this, UVOTBADPIX compares adjacent pixels. If the difference between them equates to the on-board compression threshold then the second pixel is flagged as damaged in the bad pixel map. The user has the choice of whether to search for compression-damaged pixels or not.

NULL-valued image pixels result from either corrupted or missing telemetry. These are also flagged in the bad pixel map.

It is possible for pixels to suffer from more than one type of badness. In these cases the bad pixel map contains the sum of two or more flags. For example, a bad pixel flag of 136 indicates that a flickering pixel has suffered from compression damage (8+128).

Table 3.1.2: Values for the potential flags within a bad pixel map.

Quality value	Description
0	Good
1	Dead pixel
2	Cold pixel
4	Hot pixel
8	Flickering pixel
128	Compression damaged value
256	NULL value

3.1.3 Input files

UVOTBADPIX requires two input files:

1) A level I FITS image file. This may contain multiple extensions, containing one image each. Each image has raw chip coordinates, RAWX and RAWY, e.g., sw00072901259.012/data/uvot/image/sw00072901259uvvrw.img.gz

2) The caldb product containing a cosmetic bad pixel list, e.g., $CALDB/data/swift/uvot/bcf/badpix/swubadpix20041007v001.fits

3.1.4 Output files

UVOTBADPIX has a single output file:

1) A FITS image file, with an identical extension structure as the input image. Each extension contains an image in RAWX, RAWY coordinates which matches the window size, location and binning of the input image. Each output image is populated with image flags, according to Table 3.1.2. This output file is temporary in nature and not archived by HEASARC.

3.1.5 Parameters

Table 3.1.3 lists the input parameters for UVOTBADPIX. Parantheses indicate that parameters are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can do this by typing "*plist uvotbadpix*".

Table 3.1.3: Parameter descriptions for UVOTBADPIX.

Parameter	Description
infile	Name of Level I FITS image file. These will have "raw" within the file name and reside in the "images" subdirectory.
badpixlist	Name of the badpixel list that resides in the caldb. If the $CALDB environment variable is set, the path and name of the pixel list can be replaced simply with "*caldb*".
outfile	The name of the output bad pixel map file.
(compress)	Should UVOTBADPIX search for compression-damaged pixels? The default is "*yes*". Turning this switch off will increase the speed of the tool, but should only be done if the user is certain that the data was telemetered in an uncompressed format.
(clobber)	Should UVOTBADPIX overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTBADPIX write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (1 = quiet. 5 = noisy). The default is "*1*"

3.1.6 Example

Below we provide a typical invocation of the UVOTBADPIX tool. The example files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotbadpix.

```
Namibia> uvotbadpix infile=sw00072901259ubb_rw.img badpixlist=caldb
outfile=sw00072901259ubb_bd.img compress=y clobber=y chatter=1
```

3.1.7 Warnings and Errors

3.2 UVOTMODMAP

3.2.1 Updates

Table 3.2.1: UVOTMODMAP update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.2.2 Description

All images contain systematic modulo-8 fixed-pattern noise of amplitude ~50%. This effect is the result of pixel sub-sampling on the detector. Photons passing through the UVOT aperture are amplified by a series of phospher layers and micro-channel plates, so that an individual photon results in a “splash” of 10^7 photons (an “event”), on the CCD detector. Since the event is spread over approximately 3x3 CCD pixels, an on-board algorithm can fit this distribution and centroid the event to sub-pixel resolution. CCD pixels are therefore sub-sampled on-board such that a relatively coarse detector of 256x256 4x4” active pixels yields a 2048x2048 image of 0.5x0.5” pixels.

The on-board centroiding algorithm is relatively simple, whereas the actual event distribution can be variable in both CCD position (due to incident-angle effects) and time (due to variations in detector gain). It is not efficient use of time to consistently calibrate the noise pattern and up-load this regularly to flight software. Therefore Level I FITS images contain modulo-8 fixed pattern noise.

Unfortunately there is no clean method of removing the fixed pattern noise without destroying photometric accuracy. Note that within each 8x8 pixel block, photometry is conserved. Fourier filtering would degrade this accuracy. A further complication arises when sources are bright (> 20 count s^{-1}) and suffer from coincidence losses. Over these cases the fixed-pattern noise is modified and cannot be recovered without a well-calibrated Monte Carlo analysis. The fixed-pattern will be obvious in images around very bright sources as a grid-like effect.

The existing algorithm to help treat the mod-8 problem is identical to the algorithm in the *XMM-Newton* OM tool kit. It takes each image individually, calculating the mean structure within a sliding cell or group of *n* 8x8 pixels, using a σ-clipping technique to reject outliers. In order to retain photometric accuracy, it then resizes individual pixels according to the counts within each and then rebins the entire image to yield a new linear array. An example of this algorithm working on a combined flat field image is provided in Fig. 3.2.1.

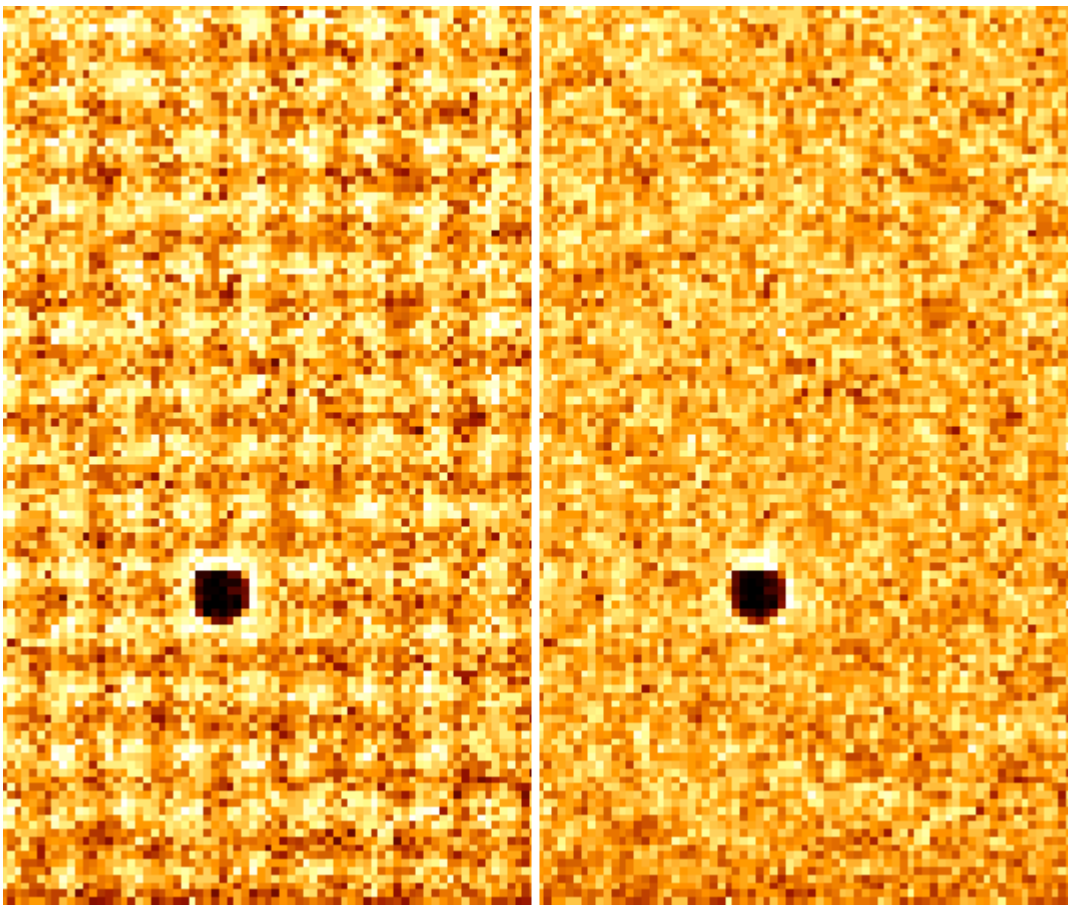

Figure 3.2.1: A sub-image of many flat fields combined, containing one cosmetically bad pixel, before and after mod-8 fixed pattern noise correction.

The disadvantage of this method is, of course, that it only works correctly on fields that are uniform. In individual images, this means that only the background is useful and point sources are thrown out by σ-clipping. Since the UVOT background is small, a large amount of pixels have to be averaged in order to provide useful event statistics and this hampers the effort to remove mod-8 noise on local scales.

Since the primary UVOT science goals are the astrometry and photometry of point sources, fixed-pattern noise will not affect this science in a major way. The one exception is grism analysis, where the mod-8 noise will be obvious in bright spectra.

Since the Swift pipeline is dedicated to processing all spacecraft, BAT, XRT and UVOT data in < 2 hr, mod-8 correction is not performed in the pipeline. UVOTMODMAP is very CPU-intensive, the overhead of running UVOTMODMAP in the pipeline is unacceptable with questionable scientific gain. It is left to the users taste, whether or not to reprocess archived data with mod-8 correction included.

In the long-term, the correct way to approach this problem is iterative fitting to a Monte Carlo generated model, that ray-traces photons through the detector. This will treat mod-8 noise and coincidence losses in a consistent way, retaining photometric accuracy. However, it is currently not clear whether the fixed-noise pattern will remain stable enough to make this method viable.

3.2.3 Input files

UVOTMODMAP requires two input files:

A level I FITS image file. This may contain multiple extensions, containing one image each. Each image has raw chip coordinates, RAWX and RAWY, e.g., sw00072901259.012/data/uvot/image/sw00072901259uvvrw.img.gz

1) The bad pixel maps. A FITS image file, with an identical extension structure as the input image. Each extension contains an image in RAWX, RAWY coordinates which matches the window size, location and binning of the input file. Each output image is populated with image flags, according to Table 3.2.1. This file is a temporary structure generated by UVOTBADPIX.

3.2.4 Output files

UVOTBADPIX has one compulsory output file and one optional output file:

1) An image file, with the same extension structure as the input image file. The new coordinate system in most UVOT cases will be either SKY or DET. This is a temporary file and not archived by the HEASARC.

2) An image file, with the same extension structure as the input image file, but containing the modulo-8 fixed-pattern noise that was calculated from each input image. This is an optional output file, not created during pipeline processing and not archived by the HEASARC.

3.2.5 Parameters

Table 3.2.3 lists the input parameters for UVOTMODMAP. Parantheses indicate that parameters are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotmodmap*".

Table 3.2.2: Parameter descriptions for UVOTMODMAP.

Parameter	Description
infile	Name of the input image file. These will probably have multiple image extensions.
badpixfile	Name of the file containing the bad pixel maps.
outfile	Name of the output image file.
(mod8prod)	Should the tool output maps of the modulo-8 fixed-pattern noise? The default is "*no*".
(mod8file)	Name of the optional output file containing modulo-8 fixed-pattern noise structure.
nsig	Significance level for σ-clipping. Those pixels with counts above or below this threshold from the mean value in a cell are discarded, and the mean recalculated.
ncell	The size of the cell that slides over the image, in units of 8 pixels.
(clobber)	Should UVOTMODMAP overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTMODMAP write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (1 = quiet. 5 = noisy).

3.2.6 Example

Below we provide a typical invocation of the UVOTBADPIX tool. The example files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotmodmap.

```
Namibia> uvotmodmap infile=sw00072901259ubb_rw.img
outfile=sw00072901259ubb_md.img badpixfile=sw00072901259ubb_bd.img mod8prod=n
mod8file=foo.fits nsig=3 ncell=16 clobber=y chatter=1 history=y
```

3.2.7 Warnings and Errors

3.3 UVOTFLATFIELD

3.3.1 Updates

Table 3.3.1: UVOTFLATFIELD update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.3.2 Description

Sensitivity variations occur across the detector. Due to the nature of the photon-counting detector, large-scale variations are expected to be negligible. However pixel-to-pixel variations may be significant. These are corrected for by dividing each image through by a calibration image of a flat field. Typically this field will be accumulated from many deep pointing with the point sources removed and the image normalized. The operation of dividing this filed is a trivial one, but this tool provides a labour-saving device to correct the many images from a given FITS file semi-autonomously.

3.3.3 Input files

UVOTFLATFIELD requires two input files:

1) Raw image file. It is optional whether these files have been treated for modulo-8 noise using uvotmodmap or not. Either case is valid. An example file location in the archive is 00072901259/uvot/image/sw00072901259uvv_rw.img.gz.

2) The caldb product containing the flat field image, e.g., $CALDB/data/swift/uvot/bcf/flats/swuppsens20041007v001.fit

3.3.4 Output files

UVOTFLATFIELD has a single output file:

1) An FITS file containing corrected images of the same dimensions, windowing and binning as the input images. These files are temporary and not archived by the HEASARC.

3.3.5 Parameters

Table 3.3.2 lists the input parameters for UVOTFLATFIELD. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotflatfield*".

Table 3.3.2: Parameter descriptions for UVOTFLATFIELD.

Parameter	Description
infile	Name of the input image file.
outfile	Name of the output image file.
flatfile	Name of the calibration file containing the flat field data. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
(cleanup)	Whether to delete temporary files created during the execution of this tool. The default is "*yes*".
(clobber)	Should UVOTFLATFIELD overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTFLATFIELD write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.3.6 Example

Below we provide a typical invocation of the UVOTFLATFIELD tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotflatfield.

```
Namibia> uvotflatfield infile=sw00072901259ubb_rw.img
outfile=sw00072901259ubb_ff.img flatfile=caldb cleanup=y clobber=y chatter=1
```

3.3.7 Warnings and Errors

3.4 SWIFTXFORM

3.4.1 Updates

Table 3.4.2: SWIFTXFORM update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.4.2 Description

SWIFTXFORM is a general *Swift* tool that converts images from one coordinate system to another. The input coordinate system can be one of the following:

Table 3.4.2: The coordinate systems in UVOT data files.

System	Description
RAW	A linear array, identical to the pixel array within the UVOT camera. The image can be any rectangular sub-array of the detector, with binning of 1, 2, 4 or 8. The array unit is "*pixels*".
DET	An array, in physical units of the image window. The array unit is "*mm*". Compared to a RAW image, the array has been flipped across the RAWX axis, so that we look up at the sky, rather than down upon the detector, distortion due to the telescope optics has been corrected for, and rebinned so that the image remains linear.
SKY	A tangential projection of the image in ecliptic coordinates, (RA_{2000}, Dec_{2000}). Compared to a DET image, the SKY image has been rotated according to the roll angle of the telescope. The array has been rebinned so that pixels are linear and orthogonal to RA and Dec.

Similarly, the output coordinate system can be one of the three alternatives in Table 3.4.2. The UVOT pipeline will only perform the conversion RAW to SKY; the one exception is for grism images where analysis should be performed after a RAW to DET conversion.

By default, SWIFTXFORM calculates the smallest output array possible that will contain the input image. However, the user does have control over the center of the image when converting to SKY, which also determines the size of the output array.

Output image are always linear arrays and this requires pixel rebinning after coordinate transformations. The algorithm calculates the output coordinates of the four corners of each input pixel. Those output pixels that overlap the resulting quadrilateral are flagged. The counts within each input pixel are distributed among the output pixels, weighted according to one of the following schemes listed in Table 3.4.3., which can be chosen by the user.

Table 3.4.3: Count redistribution methods available in SWIFTXFORM

System	Description
AREA	This method treats the counts in each pixel as being spread evenly over the area of the pixel. It distributes the value of each original pixel among the transformed pixels it covers proportionally by the fraction of overlapping area. This method preserves the sum over pixels, so the transformed image can be used to calculate fluxes. The transformed pixels values will not be integers. The argument 'bitpix=-32' must always be used with this option.
CENTER	This method assumes the counts in each pixel are concentrated at the center. For each pixel in the original image, it transforms the position of the pixel center and then adds the full pixel value to the corresponding pixel in the transformed image. This method is fast to calculate, and it preserves the sum over pixels and the integer nature of the original image. However, it produces artifacts when the original and transformed pixel grids do not coincide. So this method should only be used for e.g. translations by an integer number of pixels.
EVENT	This method mimics the effect of the COORDINATOR FTOOL on an event list. It assumes that the value of each pixel gives the number of events in that pixel. It then assigns random positions for each event within the pixel and transforms those positions to the new coordinate system. Then the events are binned into the pixels of the transformed image. If the pixel values are not integers, imagexform gives a warning and converts the values to integers in an arbitrary way. This method preserves the total (integer) number of counts in the image, and guarantees that the transformed pixel values will all be integers. This method is best for "counts" images.
INTERPOLATE	This method linearly interpolates the input image. For each pixel in the transformed image it calculates the position of the center of that pixel in the

	original image. It then linearly interpolates the original image to get the transformed pixel value. The output pixel values will not generally be integers. This is a common technique for transforming terrestrial images, but the sum of the pixels is not preserved, so fluxes derived from the transformed image will be inaccurate.
DEFAULT	In the UVOT case, this method is identical to “AREA”.

Data containing the pixel sizes, reference telescope coordinates, optical distortion parameters and rotation and flip conventions are stored in the Telescope Definition (TELDEF) file that resides in the caldb. Optical distortion is stored as an array of RAW X and Y offset parameters, sampled across the detector. Corrections for each pixel location are interpolated in two dimensions using these offsets.

3.4.3 Input files

SWIFTXFORM requires three input files:

1) An input image file.

    - Example of archive location: 00072901259/uvot/image/sw00072901259uvv_rw.img.gz

2) An attitude history file, which is a tabular record of the spacecraft’s pointing and orientation during the observation, generally with a time resolution of 10 s.

    - Example of archive location: 00072901259/auxil/sw00072901259sat.fits.gz

3) A telescope definition file. In order to convert from one coordinate system to another, the tool requires telescope data such as boresight direction, focal length, pixel size and distortion map. These quantities are contained within a single CALDB file.

    - Example of caldb location: $CALDB/data/swift/uvot/bcf/teldef/swugen20041007.teldef

3.4.4 Output files

SWIFTXFORM yields a single output file:

1) A FITS file containing one or more image extensions.

    - Archive location example: 00072901259/uvot/image/sw00072901259uvv_sk.img.gz

3.4.5 Parameters

Table 3.4.4 lists the input parameters for SWIFTXFORM. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist swiftxform*".

Table 3.4.4: Parameter descriptions for SWIFTXFORM.

Parameter	Description
infile	Name of the input FITS image file. The input file will generally have more than one image extension. SWIFTXFORM will perform a transformation on a single extension if the extension name is specified in this parameter, e.g., sw00073422001.img[00148265E]. Alternatively, all extensions will be processed if only the filename is specified.
outfile	Name of the output FITS image file. This can have multiple image extensions.
attfile	Name of the input attitude history file. This will reside in the "*att*" subdirectory of archived data.
method	Interpolation method during the transformation. The options are: AREA, CENTER, EVENTS, INTENSITY and DEFAULT. The recommended method is AREA. Definitions for these methods are provided in Table 3.4.3.
teldeffile	Name of the input telescope definition file. Entering "*caldb*" for this parameter will allow the tool to find the most appropriate version of the alignment file in the *Swift* CALDB.
to	The coordinate system of the output file, RAW, DET or SKY.
ra	For a SKY output image, the RA(2000) coordinate of the center of the array. This parameter is redundant for other types of transformation. Units are decimal degrees.
dec	For a SKY output image, the Dec(2000) coordinate of the center of the array. This parameter is redundant for other types of transformation. Units are decimal degrees.
(bitpix)	The number of bits stored in the output image. The default is to write the same type of array to output as the input array. This argument should be used when this is not a desired feature. The recommended value is 32 for most applications, corresponding to an integer array of single precision. The one exception to this rule occurs when method=AREA and the output array must be real, i.e. bitpix=-32.

(seed)	Random number seed for the EVENT interpolation method.
(aberration)	Whether velocity aberration is included in a translation to or from SKY coordinates. The default is "*no*" since velocity aberration is compensated for by on-board software before the attitude data is transmitted.
(zeronulls)	Whether to treat NULL pixel values as zero.The default is "*no*".
(copyall)	Copy all image extensions over to the output file. This is only applicable when a specific image extension is requested as **input**. The default is "*no*".
(clobber)	Should SWIFTXFORM overwrite a file with the same name as the output? The default is "*no*".
(history)	Should SWIFTXFORM write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(cleanup)	SWIFTXFORM is a perl script that stitches several base tools together. The default, "*yes*", indicates that the tool will delete temporary files created during execution.
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.4.6 Example

Below we provide a typical invocation of the SWIFTXFORM tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/swiftxform.

```
Namibia> swiftxform infile=sw00072901259uvv_rw.img
outfile=sw00072901259uvv_sk.img attfile=sw00072901259sat.fits method=AREA
to=SKY ra=299.67 dec=35.235 teldeffile=caldb bitpix=-32 clobber=y chatter=1
cleanup=y history=y
```

3.4.7 Warnings and Errors

3.5 UVOTEXPMAP

3.5.1 Updates

Table 3.5.1: UVOTEXPMAP update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.5.2 Description

UVOTEXPMAP creates exposure maps for each UVOT image in SKY coordinates. Telescope pointing will vary by some degree during, and between, exposures. Exposure maps are arrays of the same size as SKY images populated with effective exposure times ≤ the exposure integration time. They are therefore useful when analyzing sources near bad pixels or window edges where telescope drift will reduce the effective exposure times of individual pixels. Note that UVOT does not suffer from vignetting, so count rates will, in general, be linear across the detector.

Given the exposure times of individual images, at it's most basic, UVOTEXMAP takes a bad pixel map, sets good pixels to 1 and bad pixels to 0, multiplies the image by the exposure time and finally performs a transformation from RAW to SKY coordinates. Each pixel contains the interval, in units of seconds, that it was effective during the exposure.

If an image has been constructed from event-like data, it is likely that photon positions have been corrected to their true sky locations using attitude history data or similar diagnostics of the drift in spacecraft pointing. In these cases an identical correction must be made to the exposure map. A large number of intermediate exposure maps are created for each individual exposure, one map for each time quanta used to correct photon positions. The position (and

roll if available) of each map is defined by the mean telescope pointing during that time interval. The final exposure map is constructed by co-adding the series of maps and rebinning so that the final product contains the same pointing and binning as the parent image. If consecutive pointing entries in the drift data differ by a user-defined threshold, the tool will linearly interpolate new points in the table so that the threshold is never exceeded.

The UVOT approach to exposure map creation will depend upon the data format. For standard imaging mode the process is relatively simple and fast. In this case, each image pixel has been exposed for equal amounts of time so no delicate calculations of aspect drift corrections around window edges and bad pixels are required. However, if spacecraft pointing stability is found to be poor, a shift-and-add algorithm will be implemented by on-board software, where photon arrival locations will be shifted in the RAWX and RAWY dimensions according to the attitude data at 10s intervals before the image is summed. In this case UVOTEXPMAP must recreate these on-board shifts when creating the exposure map. In this mode, the spacecraft will send down an aspect report packet containing the shifts employed during this procedure. UVOTEXPMAP requires this data in order to recreate the correction in the exposure maps. Finally exposure maps for images constructed from event data can also be constructed using position corrections from attitude history data provided the same data was used to calculate the sky position of each photon. These last two cases are time consuming and CPU intensive tasks. Since the pipeline processing must complete within 2 hours, it remains to be seen whether these steps can be used by the pipeline.

3.5.3 Input files

UVOTEXPMAP requires four input files:

1) A Level II FITS input file, containing SKY coordinate images, e.g., 00072901259/uvot/image/sw00072901259uvv_sk.img.gz

2) A bad pixel file containing image extensions that correspond to the input images. This is a FITS image file, with an identical extension structure as the Level II sky image. Each extension contains an image in RAWX, RAWY coordinates. This file is a temporary structure generated by UVOTBADPIX.

3) Am aspect report packet or attitude history file which contains a tabular record of the spacecraft's pointing and orientation during the observation. , e.g., 00072901259/auxil/image/sw00072901259sat.fits.gz

4) A telescope definition file. In order to convert from one coordinate system to another, the tool requires telescope data such as boresight direction, focal length, pixel size and distortion map. These quantities are contained within a single CALDB file.

- Example of caldb location:
  $CALDB/data/swift/uvot/bcf/teldef/swugen20041007.teldef

3.5.4 Output files

UVOTEXPMAP has a single output file:

An output image array, containing exposure map image extensions in SKY coordinates. Image sizes, positions and file extension structure are identical to the input image.

3.5.5 Parameters

Table 3.5.2 lists the input parameters for UVOTEXPMAP. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotexpmap*".

Table 3.5.2: Parameter descriptions for UVOTEXPMAP.

Parameter	Description
Infile	Name of the Level II FITS image file, containing images in SKY coordinates. Files may contain more than one image extension. This file is produced by SWIFTXFORM.
badpixfile	Name of the file containing the bad pixel maps that correspond to the input images. This file is produced by UVOTBADPIX.
attfile	Name of the attitude history file, contained in the "*att*" directory in archived data.
teldeffile	Name of the telescope definition file contained in the CALDB. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the teldef file. This file contains telescope data such as an optical distortion map, focal length, boresight and pixel size.
outfile	The name of the FITS image file containing exposure maps corresponding to the input images. This file will have an identical structure to the input image file.
method	This parameter defines the method of map construction used by the tool. MEANFOV should be used for standard images, SHIFTADD for images resulting from the onboard shift-and-add process, and ATTHIST for images constructed from event mode data.

attdelta	If consecutive position records in the attitude history file differ by more than this parameter, virtual records are interpolated linearly between them so that the difference between adjacent records is no more than attdelta. The unit is arcsec.
(aberration)	Account for relativistic velocity aberration of source positions. This correction is made to attitude data on-board, so should never be required. The default is "*no*".
(cleanup)	UVOTEXPMAP is a perl script that wraps several base tools. This parameter tells the tool to delete all intermediate files in the working directory. The default is "*yes*".
(clobber)	Should UVOTEXPMAP overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTEXPMAP write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.5.6 Example

Below we provide a typical invocation of the uvotexpmap tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotexpmap.

```
Namibia> uvotexpmap infile=sw00072901259uvv_sk.img
outfile=sw00072901259uvv_ex.img  badpixfile=sw00072901259uvv_bd.img
teldeffile=caldb attfile=sw00072901259sat.fits method=MEANFOV clobber=y chatter=1
cleanup=y history=y clobber=y
```

3.5.7 Warnings and Errors

3.6 UVOTDETECT

3.6.1 Updates

Table 3.6.1: UVOTDETECT update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.6.2 Description

UVOTDETECT detects sources within a single UVOT SKY image and constructs a source table. This tool is a script wrapped around the public package sextractor. A homepage for this software, containing a description of the algorithm, may be found at:

http://terapix.iap.fr/rubrique.php?id_rubrique=91

The choice of this tool was made because of sextractor's capability to detect and measure extended sources. This script records a subset of the sextractor output in the resulting FITS table, including source positions in both epoch 2000 sky coordinates and pixel coordinates, the faint detection threshold, the size, shape and orientation of extended sources (assuming elliptical distributions), and warning flags from the tool.

3.6.3 Input files

UVOTDETECT requires one mandatory input file nd one optional one:

1) A level II input FITS file, containing one or more SKY coordinate images. Only the specified image extension will be operated upon.

Archive location, e.g.:
00072901259/uvot/image/sw00072901259uvv_sk.img.gz

2) An optional image file containing an array of weights for each pixel in the input image. This is used to determine more accurate detection thresholds for each source. The exposure map appropriate for the input image is a viable weight map.

Archive location, e,g,:
00072901259/uvot/image/sw00072901259uvv_ex.img.gz

3.6.4 Output files

UVOTDETECT has a single output file:

1) A FITS table containing one row for every source detected in the input image. Table 3.6.2 describes the content of each row.

Table 3.6.2: Description of the output table from UVOTDETECT.

Column	Description
REFID	Internal reference number of each source, 1-n.
RA	Epoch 2000 right ascension of the source in decimal degrees.
DEC	Epoch 2000 declination of the source in decimal degrees.
RA_ERR	1σ uncertainty in RA.
DEC_ERR	1σ uncertainty in Dec
THRESHOLD	Detection threshold in magnitude. This parameter is related to the **threshold** input parameter.
X_IMAGE	x-position of the source in the input image in pixels.
Y_IMAGE	y-position of the source in the input image in pixels.
X_ERR	1σ uncertainty in X_IMAGE

Y_ERR	1σ uncertainty in Y_IMAGE
PROF_MAJOR	Size of semi-major axis of source in arcsec.
PROF_MINOR	Size of semi-minor axis of source in arcsec.
PROF_THETA	Angle subtending the major axis and equator in degrees counter-clockwize.
FLAGS	Warning flags generated by sextractor.
ORIGIN	Origin of source list, e.g., image, TDRSS packet.

Quoting directly from Sec 8.1 of the sextractor user's guide, the origin of warning flags will be one of the following:

Table 3.6.3: Key to warning flags in the UVOTDETECT source table.

Flag	Description
1	The object has neighbours, bright and close enough to significantly bias photometry, or bad pixels where more than 10% of the integrated area is affected.
2	The object was originally blended with another one.
4	At least one pixel of the object is saturated, or very close it.
8	The object is truncated, i.e. too close to the image boundary.
16	Data in the extraction aperture is incomplete or corrupted.
32	Object's isophotal data is corrupted or incomplete.
64	A memory overflow occurred during deblending.
128	A memory overflow error occurred during extraction.

Note that more than one warning could be given for a source, hence FLAGS = 8+16+32 = 56 is a possible value.

3.6.5 Parameters

Table 3.6.4 lists the input parameters for UVOTDETECT. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotdetect*".

Table 3.6.4: Parameter descriptions for UVOTDETECT.

Parameter	Description
infile	Input level II image FITS file. Only one image is accepted, so if the file has multiple image extensions, the correct extension should be provided.
outfile	Output FITS source table.
weightfile	An image extension containing exposure weights for each pixel. "*none*" is the default, but an accurate exposure map should be supplied here, if the input image is a mosaic or sum of several individual exposures,
threshold	Detection threshold above the background in terms of signal-to-noise. Potential sources detected below this threshold will not be written to the output table.
(cleanup)	UVOTDETECT is a wrapper script. This parameter determines whether intermediate files are deleted from the working directory. The default is "*yes*".
(clobber)	Should UVOTDETECT overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTDETECT write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(cleanup)	UVOTDETECT is a perl script that wraps an external program. In doing so, several parameter control files need to be created before executing sextractor. This parameter tells the tool to delete the control files in the working directory. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.6.6 Example

Below we provide a typical invocation of the UVOTDETECT tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotdetect.

```
Namibia> uvotdetect infile=sw00000001001u_sk.img+1 outfile=sources.fit
weightfile=sw00000001001u_ex.img+1 threshold=2 history=y cleanup=y clobber=y
chatter=1
```

3.6.7 Warnings and Errors

3.7 UVOT2PHA

3.7.1 Updates

Table 3.7.1: UVOT2PHA update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.7.2 Description

Swift is a panchromatic instrument consisting of three detectors, BAT, XRT and UVOT. It will often be critical to analyze data from two or three of these instruments simultaneously and consistently. We therefore provide this tool that constructs data files from the UVOT images that are compatible with XSPEC, the X-ray spectral analysis package which both XRT and BAT data conform to. It then becomes possible to perform panchromatic spectral fitting across the optical, UV, soft X-ray and hard X-ray bands simultaneously. XSPEC is incorporated in the XANADU subpackage of the HEASoft distribution which is required before attempting any Swift data analysis, therefore it should already exist on your machine. A description of the XSPEC package can be found at:

http://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/index.html

UVOT2PHA is only applicable to images derived from the UVOT's lenticular (broad-band) filters, i.e., U, B, V, UVW1, UVM2, UVW2, WHITE and MAGNIFIER. The tasks for converting grism image data to XSPEC compatible spectra are contained in UVOTIMGRISM and UVOTRMFGEN. Of course, other than a central wavelength, data from an individual lenticular filter contains no spectral information. However, by combining two or more lenticular filter data files, or one or more lenticular filter file with XRT or BAT files, spectral analysis may be

performed in XSPEC such as spectral slope and redshift measurements. These can also be fit simultaneously with any XRT or BAT data available. Note this one word of caution, UVOT exposures are taken in series, not parallel, so the user must understand the potential of time-variable phenomena biasing the results of spectral fitting.

The core of this tool is inherently simple. Given two region files, one containing source counts from a specific object, the other containing background counts from around that source, UVOT2PHA will extract counts from both regions accompanied by Poisson uncertainties. These four quantities will be cast into two XSPEC-compatible files. The specific file format is documented at the HEASARC:

http://heasarc.gsfc.nasa.gov/docs/heasarc/ofwg/docs/summary/ogip_92_007_summary.html

The extraction regions should be stored in two external ascii files prior to executing this tool. These can be created by hand, or more conveniently, using standard image analysis tools such as DS9:

http://hea-www.harvard.edu/RD/ds9

or XIMAGE:

http://heasarc.gsfc.nasa.gov/docs/xanadu/ximage/ximage.html

XIMAGE is also part of the XANADU package and will be installed on your machine at the same time as the rest of the HEASoft distribution. Note that when using DS9 to create these extraction region files the user must ensure that they are written in image pixel coordinates rather than RA and Dec coordinates. This is the tool default, but it can be verified by clicking on the “region” button on the upmost tool bar above the image. The data extraction itself is executed by an internal call to XIMAGE, using the command “*counts/regionfile*”.

While the output from UVOT2PHA is XSPEC-ready, it is not scientifically useful without a suitable response matrix, which defines the spectral properties of the data. These can be downloaded from the Swift web pages, where there is one for every lenticular filter. It is critical that the correct response matrix be used with the data:

http://swift.gsfc.nasa.gov/docs/swift/proposals/swift_responses.html

UVOT2PHA propagates required keywords from the image header extension to the spectrum files. However if the user attempts to execute this tool using image files that do not contain these mandatory keywords then he or she is required to supply suitable keywords values

manually using the optional parameters ra, dec, date-obs, time-obs, date-end and time-end described in Table 3.7.2.

UVOT2PHA is a value-added tool and not employed in the automated Swift data reduction pipeline.

3.7.3 Input files

UVOT2PHA requires three input files:

1) A specific extension within a level II UVOT FITS image file, e.g., 00072901259/uvot/image/sw00072901259uvv_sk.img+1.

2) An ascii file containing the source extraction region using pixel coordinates. A simple example would be:

   image;circle(1018,1125,13.642161)

   the coordinates are relative to the lower-left pixel in the image.

3) An ascii file containing the background extraction region, e.g.:

   image;circle(1018,1125,38.700068)
   image;-circle(1018,1125,15.177994)

3.7.4 Output files

UVOT2PHA has two output files:

1) An XSPEC-compatible FITS file containing source counts.

2) An XSPEC-compatible FITS file containing background counts.

3.7.5 Parameters

Table 3.7.2 lists the input parameters for UVOT2PHA. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvot2pha*".

Table 3.7.2: Parameter descriptions for UVOT2PHA.

Parameter	Description
infile	Input image file or extension.
srcpha	Output pha file containing the source + background count rate.
bkgpha	Output pha file containing the background count rate.
srcreg	Region file defining the source extraction area.
bkgreg	Region file defining the background extraction area.
(phatype)	Type of output data. The options are "*counts*", providing an output table containing counts, or "*rate*", providing an output table containing data in units of count s^{-1}. The rate option is essential if the user intends to run UVOTMAG on the output from this tool.
(ra)	The Right Ascension (epoch 2000) of the source in decimal degrees.
(dec)	The Declination (epoch 2000) of the source in decimal degrees.
(date_obs)	Date at the start of the observation.
(time_obs)	The observation start time in MET.
(date_end)	Date at the end of the observation.
(time_end)	The observation end time in MET.
(tmpdir)	Name of the directory to store temporary files. The default is ".".
(clobber)	Should UVOT2PHA overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOT2PHA write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.7.6 Example

Below we provide a typical invocation of the UVOT2PHA tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvot2pha.

```
Namibia> uvot2pha infile=sw00072901259uvv_sk.img+1 srcpha=v.pha
bkgpha=b_bkg.pha srcreg=v.reg bkgreg=v_bkg.reg clobber=y chatter=1
```

3.7.7 Warnings and Errors

3.8 UVOTMAG

3.8.1 Updates

Table 3.8.1: UVOTMAG update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.8.2 Description

From a column of count rate data contained in a FITS table (and optional error column), UVOTMAG calculates instrumental magnitudes and fluxes, in units of erg s^{-1} cm^{-2} $Å^{-1}$, using filter-specific zero-points and flux conversion coefficients. Zero-points are defined as the magnitude at which the count rate is 1 s^{-1}. UVOTMAG will work on a wide variety off FITS tables, such as source lists, light curves and broad-band filter PHA files, provided the input column has units s^{-1}.

The tool also corrects for dead time and coincidence losses, which occur because the detector can only recognize one photon per detector pixel within each individual frame. The CCD frame rate is 11 ms, hence sources brighter than 20 count s^{-1} will suffer from coincidence loss. Obviously the magnitude of this correction is directly related to the source brightness.

3.8.3 Input files

UVOTMAG requires three input files:

1) A FITS table containing a column of count rates in units of s^{-1}. Count rate errors are optional, e.g., 00072901259/uvot/image/sw00072901259u.cat.

2) A calibration file containing the zero-points for each filter, e.g., $CALDB/data/swift/uvot/cpf/phot/swuphot20041007v001.fits.

3) A calibration file containing the coefficients of a polynomial fit to tabulated coincidence loss data versus count rate, e.g., $CALDB/data/swift/uvot/bcf/coinc/swucntcor20041007v001.fits.

3.8.4 Output files

UVOTMAG has a single output file:

1) A FITS table identical to the input file, except for additional columns for instrumental magnitude and flux. Error columns will also be generated if an input error column was provided. The original input file is over-written.

Table 3.8.2: Description of additional columns in the UVOTMAG output table.

Parameter	Description
MAG	The instrumental magnitude of the source.
MAG_ERR	Uncertainty on the instrumental magnitude.
FLUX	Source flux in units of erg s^{-1} cm^{-2} $Å^{-1}$.
FLUX_ERR	Uncertainty on the source flux in units of erg s^{-1} cm^{-2} $Å^{-1}$.

3.8.5 Parameters

Table 3.8.3 lists the input parameters for UVOTMAG. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotmag*".

Table 3.8.3: Parameter descriptions for UVOTMAG.

Parameter	Description

infile	FITS table containing a row of count rate data in units of s^{-1}.
zerofile	A CALDB file containing filter-dependent zero-points and linear flux conversion coefficients. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the zero-point file.
coinfile	A CALDB file containing coincidence loss correction data.
filter	The tool will automatically look for the correct FILTER in the keywords of the image file if this parameter is set to "*default*". However UVOTMAG is a generic tool and user has the option of passing a filter name. The filter options are u, b, v, uvw1, uvm2, uvw2, white, magnifier, ugrism, and vgrism.
(ratecol)	Name of the input table column containing the count rate data.
(errcol)	Name of the input table column containing the count rate error data.
(history)	Should xxxxxx write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.8.6 Example

Below we provide a typical invocation of the xxxxxx tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvvotmag.

```
Namibia> uvotmag infile=sw00000001001u.cat+1 zerofile=caldb coinfile=caldb filter=B
ratecol=RATE errcol=RATE_ERR chatter=1
```

3.8.7 Warnings and Errors

3.9 UVOTIMGRISM

3.9.1 Updates

Table 3.9.1: UVOTIMGRISM update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.9.2 Description

UVOTIMGRISM extracts 1-D FITS spectrum tables from 2-D grism images. The input image must be in DET coordinates to ensure that optical distortion has been removed and that the dispersion direction is at a fixed angle in the image array. The user supplies the position of a source in RA_{2000} and Dec_{2000} coordinates. UVOTIMGRISM converts these coordinates to DETX and DETY, adds appropriate offsets to the source position and rotates this image about the 0th order source so that the 1st order dispersion axis is parallel to the x-direction. The image is rebinned so that square pixels are linear in size, oriented in the dispersion and cross-dispersion directions.

The user also supplies extraction limits for the tool. The dispersion limits are in units of Angstroms (Å) and define the wavelength range of the output spectrum. The cross-dispersion limits are in pixel units and some thought should be taken to optimize the source signal over the background using these parameters. The source spectrum, with background, is generated by summing each column of pixels between the limits in the cross-dispersion direction. The uncertainty in each spectrum bin is calculated assuming a Poisson photon distribution. Note that the detector, while containing a CCD device, behaves an event detector. Consequently there is no effective readout noise.

The background is estimated by determining the mean spectrum in two user-defined regions on either side of the source in the cross-dispersion direction. The mean is calculated iteratively with pixel outliers rejected from the sum. Once a solution has been converged upon, the background spectrum under the source extraction region is determined by interpolation. Errors are, again, treated as Poisson.

Wavelength bins that contain bad pixels are flagged in the QUALITY column of the output spectrum tables. Spectrum bins are converted to wavelength units using a multi-termed polynomial defined in the CALDB. Count rates in each wavelength bin are converted to flux (erg s^{-1} cm^{-2} $Å^{-1}$) using effective area curves stored in the CALDB.

3.9.3 Input files

UVOTIMGRISM requires five input files:

1) A FITS grism image file. The file may contain multiple extensions but the tool will work on only one image at a time, so the extension of an individual image must be specified. The image must be in DET coordinates, e.g.,
   00072901259/uvot/image/sw00072901259uvu_dt.img+1

2) The corresponding bad pixel image file. This is created by UVOTBADPIX. Note that this file is not archived by HEASARC or quicklook.

3) A CALDB file containing polynomial coefficients that describe the grating equation. This equation defines the mapping between DET pixel and wavelength e.g.,
   $CALDB/data/swift/uvot/bcf/grism/swuvgrism20041007v001.fits

4) A CALDB file containing the wavelength-dependent 1st order effective area curve for the grism filters, e.g.,
   $CALDB/data/swift/uvot/cpf/arf/swu20041007v001.arf

5) A telescope definition file. In order to convert from SKY to DET coordinates. The data required to perform this operation contained within a single CALDB file, e.g.,
   $CALDB/data/swift/uvot/bcf/teldef/swu20041007.teldef

3.9.4 Output files

UVOTIMGRISM has two output file:

1) The source spectrum FITS table. This file contains two versions of the grism spectrum. The first, residing in the CALSPEC extension, is fully-calibrated and background subtracted, provided in both net counts and flux units (erg s^{-1} cm^{-2} $Å^{-1}$). 1σ uncertainties

are provided in both count and flux units. Wavelength I provided in Å. This spectrum is intended as a standalone product for inspection and analysis with generic software.

The second version, in the SPECTRUM extension, is constructed to be XSPEC (http://heasarc.gsfc.nasa.gov/docs/xanadu/xspec) compatible. This allows the grism spectra to be analyzed consistently and simultaneously with the XRT data. Conventions for high-energy spectral analysis differ from the usual optical approach. Raw, uncalibrated spectra are stored separately from calibration files. Spectral models are fit iteratively to the data by folding the model through the calibrations and then minimized by comparison with the raw data. Fluxes are determined generally from the best-fit model, not the data. XSPEC-compatible data are provided in terms of raw counts, pixels (CHANNEL) and bad pixel flags (QUALITY). The user must supply a suitable calibration file before XSPEC analysis, and this is constructed using UVOTRMFGEN. The counts in this extension have not been background-subtracted. This is done within XSPEC software and requires a separate background spectrum file.

A third extension, IMAGE, contains the 0th and 1st order source in the DET image, rotated to the dispersion axis.

An example of this FITS file can be found in the archive typically at a location resembling, e.g., sw00072901259.012/data/uvot/product/sw00072901259uvu.pha

2) The background spectrum FITS table for use during XSPEC analysis, e.g., sw00072901259.012/data/uvot/product/sw00072901259uvubk.pha.

3.9.5 Parameters

Table 3.9.2 lists the input parameters for UVOTIMGRISM. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotimgrism*".

Table 3.9.2: Parameter descriptions for UVOTIMGRISM.

Parameter	Description
infile	Name of the input grism image file or extension.
outfile	Name of the output source spectrum file.
backfile	Name of the output background spectrum file.

wavefile	Name of the calibration file containing the pixel-to-wavelength conversion factors. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
areafile	Name of the calibration file containing the filter effective areas. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
teldeffile	Name of the calibration file containing the telscope definition data, such as pixel size and optical distortion map. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
badpixfile	Name of the bad pixel image file that corresponds to the input grism image. The source of this bad pixel image is UVOTBADPIX.
ra	Right Ascension (epoch 2000) of the source.
dec	Declination (epoch 2000) of the source.
(sourcex)	0th order position in DETX units (mm). This position will be used instead of ra if ra = -1.
(sourcey)	0th order position in DETY units (mm). This position will be used instead of dec if dec = -1.
(nsigma)	For each image column, the threshold above or below the mean value over which individual pixels should be rejected from the background calculation.
(uvang)	UV grism filter only. The angle in degrees subtended by the DETX axis and the dispersion direction of the 1st order light.
(uvsrcwid)	UV grism filter only. The size of the spatial filter region of the source in cross-dispersion direction. The units are pixels.
(uvbkgwid)	UV grism filter only. The extent of the spatial filter region of the background in cross-dispersion direction. The units are pixels.
(uvmin)	UV grism filter only. The lower limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(uvmax)	UV grism filter only. The upper limit of the spatial filter region in the dispersion direction. The units are Angstroms.

(vang)	V grism filter only. The angle in degrees subtended by the DETX axis and the dispersion direction of the 1st order light.
(vsrcwid)	V grism filter only. The size of the spatial filter region of the source in cross-dispersion direction. The units are pixels.
(vsrcwid)	V grism filter only. The extent of the spatial filter region of the background in cross-dispersion direction. The units are pixels.
(vmin)	V grism filter only. The lower limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(vmax)	V grism filter only. The upper limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(cleanup)	UVOTIMGRISM is a wrapper script. This parameter determines whether intermediate files are deleted from the working directory. The default is "*yes*".
(clobber)	Should UVOTIMGRISM overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTIMGRISM write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.9.6 Example

Below we provide a typical invocation of the UVOTIMGRISM tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotimgrism.

```
Namibia> uvotimgrism infile=sw00072901259ugv_dt.img+1
outfile=sw00072901259ugv_sr.pha backfile=sw00072901259ugv_bk.pha
badpixfile=badpix.img+1 wavefile=caldb areafile=caldb teldeffile=caldb ra=23.35
dec=-41.823 vang=206.7 vsrcwid=20 vbkgwid=50 vmin=2900 vmax=5500 nsigma=5
cleanup=y clobber=y history=y chatter=1
```

3.9.7 Warnings and Errors

3.10 UVOTRMFGEN

3.10.1 Updates

Table 3.10.1: UVOTRMFGEN update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.10.2 Description

As described in section 3.09, the output from UVOTIMGRISM is not yet ready for spectral analysis using the XSPEC package. First we must construct a specific calibration file that maps raw pixels to physical units and converts raw counts to flux energy. The term for this calibration file is Redistribution Matrix FILE or RMF. XSPEC will fold analytic or tabular spectral models through the RMF before performing a statistical comparison between data and model and iterative minimization of the parameters to produce a best-fit model.

The redistribution matrix is constructed using two factors. The first is a simple translation from pixel number to photon energy using the grism dispersion equation stored in the caldb. This takes the form of a low-order polynomial. The second is the finite possibility that a photon will land on the pixel predicted by the dispersion equation. In other words each row of the redistribution matrix is convolved by a function that defines the 1st order spectral resolution of the grism. Currently this function is assumed to be Gaussian, where the Full-Width Half-Maximum (FWHM), which varies with wavelength, is stored in the caldb.

In order to convert raw counts to flux, the area under the convolution function above for each pixel varies as a function of wavelength and is equal to the 1st order effective area of the optics.

3.10.3 Input files

UVOTRMFGEN requires three input files:

1) A FITS table containing an XSPEC-compatible spectrum. UVOTIMGRISM will create this file, e.g., 00072901259/uvot/product/sw00072901259uvu_sr.pha

2) A CALDB file containing the wavelength-dependent 1st order effective area curve for the grism filters, e.g., $CALDB/data/swift/uvot/cpf/arf/swu20041007v001.arf

3) A CALDB file containing the line spread function of the 1st order grism spectrum, e.g., $CALDB/data/swift/uvot/bcf/grism/swvgrism20041007v001.fits

3.10.4 Output files

UVOTRMFGEN has a single output file:

A FITS file containing an ancillary response matrix, normalized to the effective area of the 1st order grism throughput, e.g., 00072901259/uvot/product/sw00072901259uvu_sr.rsp

3.10.5 Parameters

Table 3.10.2 lists the input parameters for UVOTRMFGEN. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotrmfgen*".

Table 3.10.2: Parameter descriptions for UVOTRMFGEN.

Parameter	Description
spectrum	XSPEC-compliant FITS table containing a 1st order grism spectrum.
outfile	A Redistribution Matrix File, normalized for the 1st order effective area of the grism filter.
areafile	Name of the calibration file containing the filter effective areas. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.

lsffile	Name of the calibration file containing the 1st order spectral line spread functions. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
(clobber)	Should UVOTRMFGEN overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTRMFGEN write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.10.6 Example

Below we provide a typical invocation of the UVOTRMFGEN tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotrmfgen.

```
Namibia> uvotrmfgen spectrum=sw00072901259ugv_sr.pha
outfile=sw00072901259ugv_sr.rsp areafile=caldb lsfile=caldb clobber=y history=y
chatter=1
```

3.10.7 Warnings and Errors

3.11 UVOTMAGHIST

3.11.1 Updates

Table 3.11.1: UVOTMAGHIST update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.11.2 Description

This tool is a labour-saving device for the analysis of UVOT image mode data. In general, provided a user has suitable region files for a specific object, the extraction of count rates from an image is a trivial exercise using, e.g. the "*counts/regionfile*" command in the HEASoft XIMAGE, while the conversion of these count rates to instrumental magnitude and flux is provided by a single call to UVOTMAG. With the large number of images produced by the UVOT, this task becomes both arduous and repetitive. UVOTMAGHIST is a perl script that performs an identical region extraction over all of the images contained in either a Level I, II or II image file and converts each count rate to magnitude and flux. Output files are a FITS table containing the magnitude history of the source, and an optional GIF file containing the filter magnitude against time.

3.11.3 Input files

UVOTMAGHIST has three input files:

1) An FITS file containing a series of image extensions, e.g., 00072901259/uvot/image/sw00072901259uvv_sk.img.gz

2) A calibration file containing the zero-points for each filter, e.g., $CALDB/data/swift/uvot/cpf/phot/swuphot20041007v001.fits.

3) A calibration file containing the coefficients of a polynomial fit to tabulated coincidence loss data versus count rate, e.g., $CALDB/data/swift/uvot/bcf/coinc/swucntcor20041007v001.fits.

3.11.4 Output files

UVOTMAGHIST has two output files:

1) A FITS table containing time, count rate, magnitude and flux data for each image in the input file. Definitions for the table columns are provided in Table 3.11.2.

Table 3.11.2: Column definitions for UVOTMAGHIST output.

Column	Definition
MET	Mission Elapsed Time of mid-exposure. Units are s and MET = 0 corresponds to midnight, 2001 Jan 1.
TIME	Time since a recent reference time. The reference is generally the BAT trigger. Units are s.
EXPOSURE	Exposure time in s.
RATE	Count rate of source. Units are counts s^{-1}.
RATE_ERR	1σ uncertainty on RATE. Units are counts s^{-1}.
MAG	Instrumental magnitude.
MAG_ERR	1σ uncertainty on MAG.
FLUX	Source flux in units of erg s^{-1} cm^{-2} $Å^{-1}$.
FLUX_ERR	1σ uncertainty on FLUX. Units are erg s^{-1} cm^{-2} $Å^{-1}$.
MAG_UPPLIM	The detection limit of a source above the background count rate.
MAG_LOWLIM	The magnitude at which a source becomes saturated. Fluxes measured to be brighter than this value will be lower limits only.

2) An optional GIF file plotting instrumental magnitude against time. Upper and lower magnitude limits are also provided.

3.11.5 Parameters

Table 3.11.3 lists the input parameters for UVOTMAGHIST. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotmaghist*".

Table 3.11.3: Parameter descriptions for UVOTMAGHIST.

Parameter	Description
infile	Input FITS image file containing a series of image extensions.
ouffile	Output FITS table containing exposure times, exposure durations, count rates, instrumental magnitudes, fluxes and detection limits.
plotfile	Optional GIF file presenting instrumental magnitude over time. "*NONE*" produces no GIF output.
zerofile	A CALDB file containing filter-dependent zero-points and linear flux conversion coefficients. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the zero-point file.
coinfile	A CALDB file containing coincidence loss correction data.
ra	Epoch 2000 Right Ascension of the source in decimal degrees.
dec	Epoch 2000 Declination of the source in decimal degrees.
(srcas)	Radius of a circular extraction region for the source in units of arcsec.
(bkgas)	Radius of a circular extraction region for the background in units of arcsec.
(cleanup)	UVOTMAGHIST creates a number of intermediate files in your working directory. These are only useful for software developers. This option removes all intermediate files at the end of the routine. The default is "*yes*".
(clobber)	Should UVOTMAGHIST overwrite a file with the same name as the output? The default is "*no*".

(history)	Should UVOTMAGHIST write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.11.6 Example

Below we provide a typical invocation of the UVOTMAGHIST tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotmaghist.

```
Namibia> uvotmaghist infile=sw00072901259uvv_sk.img.gz outfile=maghist.fit
plotfile=maghist.gif zerofile=caldb coinfile=caldb ra=23.35 dec=41.823 srcas=3
bkgas=10 cleanup=y history=y clobber=y chatter=1
```

3.11.7 Warnings and Errors

3.12 UVOTIMSUM

3.12.1 Updates

Table 3.12.1: UVOTIMSUM update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

3.12.2 Description

By co-adding a series of individual images the depth of UVOT observations can be maximized. There are many tools available to perform this operation, e.g., XIMAGE, which is supplied as part of the HEASoft distribution. During data reduction pipeline development, it became clear that existing software was too slow to meet data availability requirements, therefore this tool was developed to increase the efficiency of the pipeline. By default, UVOT sky images are all oriented in the same direction which simplifies the rebinning necessary to accurately add images from slightly different pointings. While users are welcome to employ this tool, note that it was developed for speed, not flexibility. For example, there are no options to choose a subsample of images from a UVOT file. It is recommended that post-pipeline users instead use XIMAGE for image-combining tasks:

http://heasarc.gsfc.nasa.gov/docs/xanadu/ximage/ximage.html

In order to avoid the smearing effects of spacecraft aspect drift, UVOTIMSUM must translate each image so that they have a common pointing. This requires image rebinning which results in the loss of data near the detector or window edges and the smearing of bad pixels. It is therefore critical that exposure maps are constructed for each image using UVOTEXPMAP and these maps fed through UVOTIMSIM in an entirely consistent way to the data mages. By

default, the output image shares a consistent boresight position with the first exposure in the series, and has the same pixel binning as the coarsest image in the series.

There are two rebinning methods available to the user. 1) “GRID” assumes that all images are oriented in the same direction. This means that image rotations are ignored during image binning, speeding up the process to a suitable level for rapid pipeline processing. All UVOT level II data is oriented so that north is up and east is to the left. 2) “XIMAGE” is a slower process that makes no assumption concerning image rotation. Both rotation and translation are performed during this procedure.

3.12.3 Input files

UVOTIMSUM requires a single input file:

1) A FITS image file containing a series of image extensions, e.g., 00072901259/uvot/image/sw00072901259uvv_sk.img.gz

3.12.4 Output files

UVOTIMSUM has a single output file:

1) A FITS file containing a single image extension.

3.12.5 Parameters

Table 3.12.2 lists the input parameters for UVOTIMSUM. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing “*plist uvotimsum*”.

Table 3.12.2: Parameter descriptions for UVOTIMSUM.

Parameter	Description
infile	Input FITS image file containing a series of image extensions.
outfile	Output FITS image with a single image extension.
method	Image rebinning method. Options are GRID or XIMAGE.
(pixsize)	Pixel size for the output image. The default (or if pixsize=0) is to rebin the input images to match the coarsest image in the series. Units are degrees.
(cleanup)	UVOTIMSUM creates a number of intermediate files in your working directory. These are only useful for software developers. This option removes all

	intermediate files at the end of the routine. The default is "*yes*".
(clobber)	Should UVOTIMSIM overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTIMSIM write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

3.12.6 Example

Below we provide a typical invocation of the xxxxxx tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotimsum.

```
Namibia> uvotimsum infile=sw00072901259uvv_sk.img.gz outfile=v_total.img
method=GRID pixsize=0 cleanup=y history=y clobber=y chatter=1
```

3.12.7 Warnings and Errors

4. Event Tools

4.1 UVOTSCREEN

4.1.1 Updates

Table 4.1.1: UVOTSCREEN update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

4.1.2 Description

UVOTSCREEN reads a FITS event table, determines the quality of each event, grades each event as 'good' or 'bad' according to user-defined criteria, and rejects all bad events from the output FITS table. Screening criteria can be related to internal data, performing logical calculations on, e.g., the arrival times of events and event position, but can also be related to external data, such as the orbit and attitude data of the spacecraft during the observation.

Event quality can be deemed to be poor based on two tests. The first is to compare the pixel position with a list of known cosmetic defects over the detector that is stored in the caldb. UVOTSCREEN updates the values of the QUALITY column in the FITS table for cosmetic defects. The second test is to check pixels for compression damage. UVOT telemettry may or may not come down in a compressed format, depending on the volume of the data. The compression schemes can potentially corrupt data if events from the same readout frame are widely separated or too numerous, or events occur to infrequently in time. Compression damaged is already flagged in the QUALITY column by the telemetry-to-FITS conversion software executed in the pipeline.

Table 4.1.2: Types of event badness found in Event mode tables.

QUALITY	Description
0	Good event
1	Dead pixel
2	Cold pixel
4	Hot pixel
8	Flickering pixel
16	Compression damaged event position
32	Compression damaged event time
64	Compression has caused loss of events from frame

Data is screened by calling the FTOOLS MAKETIME and EXTRACTOR internally. Screening arguments are a string of operators passed to the tool as input parameters. For example, to screen by both time and pixel quality, an argument resembling:

evexpr="time > 123456789 && quality == 0"

is appropriate. External parameters, from, e.g., tbe orbit and attitude filter file provided with yout data can also be used to screen events. For example, the following argument will EXCLUDE events from the output file that were obtained within 10 degrees of the bright Earth limb or from within the South Atlantic Anomaly:

aoexpr="BR_EARTH > 10 && SAA == 0"

The full functionality and syntax of these operators is discussed in the MAKETIME and EXTRACTOR on-line helps, e.g., type fhelp extractor on your command line.

UVOTSCREEN was developed as a pipeline tool rather than a user tool. it is recommended that general Swift users screen their UVOT data using the general HEASoft tools MAKETIME and XSELECT:

http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/ftools/xselect/xselect.html

The one exception to this rule is if the bad pixel table in the Swift caldb has been updated since the pipeline processing of your data. In this case, UVOTSCREEN should be run in order to populate the QUALITY column in the event tables correctly.

4.1.3 Input files

UVOTSCREEN requires three input files:

1) A FITS table containing UVOT events, e.g., 00072901259/uvot/event/sw00072901259uvvpo_uf.evt.gz

2) A FITS table containing time-tagged spacecraft orbit and attitude data for the observation, e.g., 00072901259.012/auxil/sw00072901259sao.fits.gz

3) The caldb product containing a cosmetic bad pixel list, e.g., $CALDB/data/swift/uvot/bcf/badpix/swubadpix20041007v001.fits

4.1.4 Output files

UVOTSCREEN has a single output file:

1) A FITS table containing screened UVOT events, e.g., 00072901259/uvot/event/sw00072901259uvvpo_cl.evt

4.1.5 Parameters

Table 4.1.3 lists the input parameters for UVOTSCREEN. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotscreen*".

Table 4.1.3: Parameter descriptions for UVOTSCREEN.

Parameter	Description
infile	Name of the input UVOT event file.
outfile	Name of the screened output event file.
attorbfile	The attitude and orbit data file. This is calculated in the Swift pipeline and supplied with each observation in the archive, but can also be generated by the

	user using the PREFILTER tool.
badpixfile	Name of the calibration file containing a list of cosmetic defects over the detector. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
aoexpr	The filtering expression, based on the attitude and orbit data, with which to screen the events.
evexpr	The filtering expression, based on internal data within the event file, with which to screen the events.
(cleanup)	UVOTSCREEN creates a number of intermediate files in your working directory. These are only useful for software developers. This option removes all intermediate files at the end of the routine. The default is "*yes*".
(clobber)	Should UVOTSCREEN overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTSCREEN write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

4.1.6 Example

Below we provide a typical invocation of the UVOTSCREEN tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotscreen.

```
Namibia> uvotscreen infile=sw00072901259uvvpo_uf.evt.gz
outfile=sw00072901259uvvpo_cl.evt badpixfile=caldb
attorbfile=sw00072901259sao.fits.gz aoexpr="SUN_ANGLE > 20" evexpr="QUALITY ==
0" cleanup=y history=y clobber=y chatter=1
```

4.1.7 Warnings and Errors

4.2 UVOTEVGRISM

4.2.1 Updates

Table 4.2.1: UVTEVGRISM update history.

HEADAS Version	Date	Description of updates
8.0		Released to Swift science team.
9.0	2004-10-07	Released for Burst Advocate training.

4.2.2 Description

UVOTEVGRISM performs two functions. Firstly it screens events according to position, keeping only those that coincide approximately with the 0th and 1st order light of a specific source. The resulting list is output to a new FITS event table. By default this table will also include a contribution from the background sky and possibly neighbouring sources that are confused with the target. Secondly this tool uses the UVOT dispersion equations to calculate wavelengths for each event. Results are written in units of Angstroms (Å) to a column called WAVELENGTH. The dispersion equation is stored as polynomial coefficients in the caldb.

The user must supply a source position in RA_{2000} and Dec_{2000} coordinates, or alternatively a location of the 0th order source in detector coordinates (DETX,DETY) in units of mm. They must also supply the angle of the dispersion axis relative to the DETX direction, the cross-dispersion width of the screened region and the wavelength range.

In order to improve the accuracy of the wavelength calculations the tool uses the attitude history file to track spacecraft pointing during the grism exposures and correct event positions on the detector. In the absence of a series of records in the attitude history, the tool will interpolate linearly between two disparate records.

4.2.3 Input files

UVOTEVGRISM requires four input files:

1) A FITS event table, e.g., 00072901259/uvot/ievent/sw00072901259uvupo_cl.evt.gz

2) A spacecraft attitude file covering the duration of the UVOT exposure, e.g., $CALDB/data/swift/misc/sw00072901259sat.fits.gz

3) A CALDB file containing polynomial coefficients that describe the grating equation. This equation defines the mapping between DET pixel and wavelength e.g., $CALDB/data/swift/uvot/bcf/grism/swuvgrism20041007v001.fits

4) A telescope definition file. In order to convert from SKY to DET coordinates. The data required to perform this operation contained within a single CALDB file, e.g., $CALDB/data/swift/uvot/bcf/teldef/swu20041007.teldef

4.2.5 Output files

UVOTEVGRISM has a single output file:

1) A FITS event table, screened by region to contain 0th and 1st order events from a specific source, and with an additional column containing the wavelength of each event in units of Å.

4.2.5 Parameters

Table 4.2.2 lists the input parameters for UVOTEVGRISM. Parantheses indicate parameters that are not compulsory. If these parameters are not specified on the command line, the tool will look-up the current value in the parameter file. Users can inspect the parameter file by typing "*plist uvotevgrism*".

Table 4.2.2: Parameter descriptions for UVOTEVGRISM.

Parameter	Description
infile	Name of the input grism event file.
outfile	Name of the output grism event file, with wavelength column added.

wavefile	Name of the calibration file containing the pixel-to-wavelength conversion factors. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
teldeffile	Name of the calibration file containing the telscope definition data, such as pixel size and optical distortion map. If the CALDB environment variable is set, "*caldb*" will point the tool to the most relevant version of the file.
attfile	Name of the attitude history table for the observation.
(attlim)	The number of seconds beyond a particular entry in the attitude history file after which the tool is allowed to interpolate pointing data. The tool will warn the user if gaps in the attitude file exceed this limit.
ra	The Right Ascension (epoch 2000) of the source.
dec	The Declination (epoch 2000) of the source.
(uvang)	UV grism filter only. The angle in degrees subtended by the DETX axis and the dispersion direction of the 1st order light.
(uvwid)	UV grism filter only. The size of the spatial filter region in cross-dispersion direction. The units are pixels.
(uvmin)	UV grism filter only. The lower limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(uvmax)	UV grism filter only. The upper limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(vang)	V grism filter only. The angle in degrees subtended by the DETX axis and the dispersion direction of the 1st order light.
(vwid)	V grism filter only. The size of the spatial filter region in cross-dispersion direction. The units are pixels.
(vmin)	V grism filter only. The lower limit of the spatial filter region in the dispersion direction. The units are Angstroms.
(vmax)	V grism filter only. The upper limit of the spatial filter region in the dispersion direction. The units are Angstroms.

(clobber)	Should UVOTEVGRISM overwrite a file with the same name as the output? The default is "*no*".
(history)	Should UVOTEVGRISM write HISTORY keywords to the output file? This creates a record of the processing performed on the file. The default is "*yes*".
(chatter)	Verbosity of the tool (0-5). This parameter control how chatty the tool is (0 = quiet. 5 = noisy).

4.2.6 Example

Below we provide a typical invocation of the UVOTEVGRISM tool. Example input and output files can be copied from ftp://heasarc.gsfc.nasa.gov/docs/swift/foo/uvotevgrism.

```
Namibia> uvotevgrism infile=sw00072901259uvupo_cl.evt.gz outfile=vgrism.evt
wavelfile=caldb teldeffile=caldb attfile=sw00072901259sat.fits.gz attlim=32
vang=212.5 vwid=20 vmin=2900 vmax=5500 history=y clobber=y chatter=1
```

4.2.7 Warnings and Errors

5. Glossary

Provided below are list of common acronyms used in this document.

Table 4.1: List of acronyms

Acronym	Big words
BAT	Burst Alert Telescope
CALDB	Calibration Database
FITS	Flexible Image Transport System
GSFC	Goddard Space Flight Center
HEASARC	High Energy Astrophysics Science Archive and Research Center
HEASoft	High Energy Astrophysics Software
HK	Housekeeping
NASA	National Aeronautics and Space Administration
OGIP	Office of Guest Investigator Programs
SDC	Swift Data Center
SSC	Swift Science Center
UVOT	Ultraviolet and Optical Telescope
XRT	X-ray Telescope

6. References

ds9:
http://hea-www.harvard.edu/RD/ds9

HEAdas:
http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/headas/developers_guide

HEASARC:
http://heasarc.gsfc.nasa.gov

HEASoft:
http://heasarc.gsfc.nasa.gov/docs/software/lheasoft

sextractor:
http://terapix.iap.fr/rubrique.php?id_rubrique=91

SDC:
http://swift.gsfc.nasa.gov/sdc

ßSSC:
http://swift.gsfc.nasa.gov

XIMAGE:
http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/xanadu/ximage/ximage.html

XRONOS:
http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/xanadu/xronos/xronos.html

XSPEC:
http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/xanadu/xspec/index.html