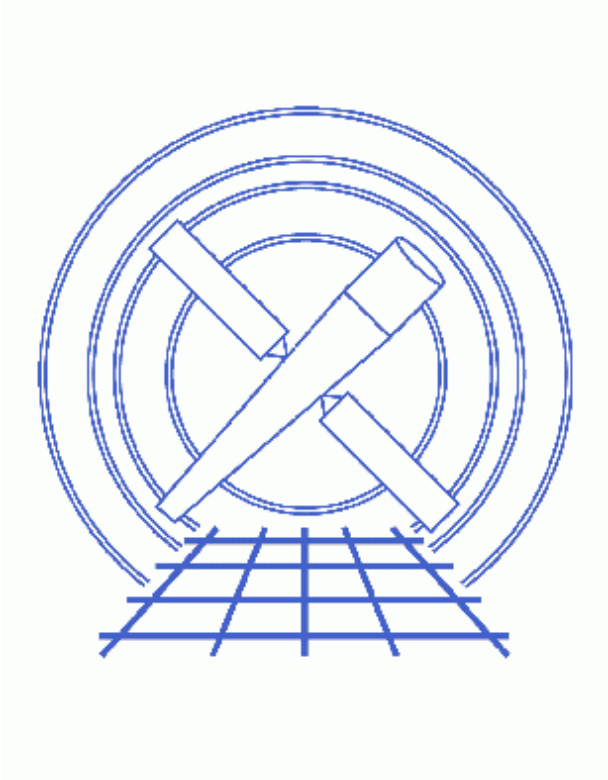
Overview: Detecting Sources in Imaging Observations



CIAO 3.4 Science Threads

Table of Contents

Overview: Detecting Sources in Imaging Observations

CIAO 3.4 Science Threads

Overview

Last Update: 1 Dec 2006 − reviewed for CIAO 3.4: no changes

Synopsis:

An overview and comparison of the three Detect tools: `celldetect`, `vtpdetect`, and `wavdetect`. A complete description of these algorithms and their uses is available from the Detect manual.

Related Links:

- Analysis Guide: Extended Sources

Proceed to the HTML or hardcopy (PDF: A4 | letter) version of the thread.

The CIAO Detect Package

The detection of significant features in 2−D images is at the heart of much of astronomy, as well as of other areas of science. X−ray astronomy poses some special challenges (e.g. large sparse arrays, Poisson statistics) that mandate specially developed methods for source detection. Moreover, the extremely small beam area of the Chandra mission can resolve tight clusters of sources. However, the small beam also means that more diffuse sources will also be resolved, making them more difficult to detect.

The CIAO Detect package includes three different source detection programs which identify statistically significant brightness enhancements, deriving from both unresolved (point) and resolved (extended) x−ray sources:

- `celldetect` uses a sliding square detect cell whose size is matched to the instrument PSF (or fixed by the user) to search for statistically significant enhancements over the background. This tool works well in simple fields with well−separated point sources.

  More detailed information about `celldetect` is available from the Detect manual, including:

  - Running `celldetect`
  - False Detection Rates

  - ♦ Theory
  - ♦ Parameters and Data Products Reference
- `vtpdetect` uses a Voronoi tessellation to determine individual densities, or fluxes, for every occupied pixel. The tool then analyzes the distribution of densities for significant source enhancements. This is the only detect tool that is scale−free.

  More detailed information about `vtpdetect` is available from the Detect manual, including:

  - ♦ Running `vtpdetect`
  - ♦ Theory
  - ♦ Parameters and Data Products Reference
- `wavdetect` correlates the image with wavelets of different scales (selected by the user) and then searches the results for significant correlations. Note that `wavdetect` is a wrapper for the tools `wtransform` and `wrecon`.

  More detailed information about `wavdetect` is available from the Detect manual, including:

  - ♦ Running `wavdetect`
  - ♦ Theory
  - ♦ Parameters and Data Products Reference

Comparison of the Detect Tools

In general, `wavdetect` and `vtpdetect` offer more capability for complex fields with both point and extended sources, but place a significantly heavier demand on computer resources. For `wavdetect`, it is recommended that you restrict the image space as much as possible; see Section 1.5 − Data Manipulation (DM) Filtering − of the Detect manual for information on how to do so.

Method	*Strengths*	*Weaknesses*
`celldetect`	• Long heritage (EINSTEIN, Rosat); well understood. • Good for faint point sources, outside crowded fields.	• Requires fine tuning of parameters for extended sources. • Divides extended sources into multiple point sources. • Has difficulty separating closely spaced point sources.
`vtpdetect`	• Finds faint, low surface brightness features. • Extended sources found as single source in visually "correct" way, regardless of their actual shape. • Detection process is scale−free.	• Combines closely spaced point sources. • Combines diffuse emission with embedded point sources. • Slow for more than about ~10^5 photons.

`wavdetect`	• Separates closely spaced point sources. • Finds extended sources so long as wavelet scales are chosen appropriately.	• Slower than `celldetect`.

Comparison of Detect Methods

A. Factors Affecting Performance And Runtimes

- ***`celldetect`***

  The primary factor that influences the runtime of `celldetect` is the size of the dataset. Two secondary factors are the number of cell sizes to examine and the actual number of detected sources.
- ***`wavdetect`***

  The three most important factors that affect the runtime of `wavdetect` are:

  1. the size of the dataset
  2. the number and size of wavelet scales
  3. the number of background cleaning iterations.

  The first two factors influence both `wtransform` and `wrecon`, the third only affects `wtransform` .

  For large wavelet sizes, the dataset needs to be padded. This causes the size of the data to be artificially augmented.

  `wavdetect` also uses a lot of computer memory. At present, 1024 x 1024 pixels should be considered the practical limit for the size of the input dataset on most machines.
- ***`vtpdetect`***

  The runtime of `vtpdetect` does not depend on the spatial size of the dataset, but depends on many other factors and is rather unpredictable. The most important factors are the number of unique photon locations and the overall number of events.

  `vtpdetect` will run quickly if the number of photons is low and there is a high contrast between background and sources. In the opposite scenario (i.e. when one has a large number of photons and a large number of faint sources), the `vtpdetect` run can be very long, since fitting background becomes an arduous task. In these situations, our tests have shown that `vtpdetect` becomes very slow if the observation contains more than ~10^5 photons.

B. Sample Runtimes

To give the reader a rough idea on the performance, we ran all three tools on various subsets of two simulated HRC−I observations. One simulation contained a set of sources on top of a flat background, the other contained only a flat background, but for a 10 times longer exposure, i.e. containing roughly 10 times more background events.

The following table shows the runtimes on a Sun Ultra 1 with a 167 MHz processor and 124 MB of RAM. The same runs on a Sun SPARCstation 5 with 70 MHz CPU and 64 MB of RAM were ~3 times slower. The runtimes

of `celldetect` and `wavdetect` are not affected by the increase in the number of events; the major factor is instead the size of the dataset. In the case of `vtpdetect`, however, the number of events is the primary factor. The tool slows down considerably if the number of events is large.

Some things to keep in mind about the runs reported in the table:

- only one wavelet scale was analyzed for `wavdetect`
- only one background iteration was performed for both `wavdetect` and `vtpdetect`

On account of these two items, actual runtimes will normally be longer than indicated. Also:

- to a first approximation, the time required for execution of `wrecon` and `vtpdetect` increases linearly with the number of iterations.
- each additional scale in `wavdetect` increases the runtime by roughly the same amount of time.
- The ``recursive blocking scheme'' was used in `celldetect` runs for 32k x 32k data.

Size (pixels)	N_{evt}	Approximate run time (min:sec)		
		`celldetect`	`wavdetect`	`vtpdetect`
Short exposure, bkg+sources				
512 x 512	400	0:25	3:45 (3:00+0:45)	0:03
1024 x 1024	1300	0:50	11:40 (9:30+2:10)	0:05
2048 x 2048	3800	2:00	−−	0:16
32768 x 32768	$1.1e10^5$	8:50	−−	9:40
Long exposure, bkg only				
512 x 512	1800	0:25	3:35 (2:55+0:40)	0:15
1024 x 1024	7300	0:50	11:30 (9:30+2:00)	0:50
2048 x 2048	28000	2:00	−−	1:40
32768 x 32768	$1.1e10^6$	8:50	−−	−−

C. Comparative Results For The Chandra Deep Field South

As described in section 2.4 of the Detect manual, we chose to use ObsID 2405 (part of Chandra Deep Field South) as a test field because it contributes 10% of the total exposure time of the Deep Field South. From running detect on the full exposure, we know the locations of all real sources which could be detected from the short exposure.

For this comparison, we have run the tools with their default parameter values. For `celldetect`, we used a S/N threshold of 3, an encircled energy of 0.9, and the PSF library of 2001 April. For `wavdetect`, we used scales of 1,2,4,8, & 16 and `sigthresh = E-6`. We found 71 detections for the block=2 image and 26 for the inner 1024

x 1024 area at block=1; of these, only 4 were unique. For `vtpdetect`, we used `scale = 1` and a maximum probability of being a false source (limit) equal to E−5 and E−6. The results are shown in the following table.

Tool	# of Detections	# of Edge Detections	# of Spurious Detections	# of Valid Detections	Percent Spurious
`celldetect`	36	1	1	34	3%
`wavdetect`	75	19	0	56	0%
`vtpdetect` (E−5)	61	0	6	55	10%
`vtpdetect` (E−6)	48	0	2	46	4%

The distinction between "spurious" and "edge" is somewhat subjective. In the case of `wavdetect` , the edge detections were close to, but not actually on, the edge. The percent spurious assumes that these detections don't count; i.e. if we had used an exposure map, they probably would not have been detected. By adjusting various parameters, each tool could be made to perform better or worse; the values presented here are indicative of the norm.

Running The CIAO Detect Tools

Separate threads have been created for each of the CIAO Detect tools. Each thread illustrates running the tool, selecting appropriate parameter values, etc.:

- Running `celldetect`
- Running `vtpdetect`
- Running `wavdetect`

Output of Detect: _src Files

All three Detect programs have a common output format, a `_src` file. (The CXC archive also contains such files from standard data processing. They are labelled in the style `acis*_src2.fits` where the "2" indicates that they were built during Level 2 processing.)

For each source these FITS files contain the RA, Dec of each source (with uncertainties), the net source counts and background (with uncertainties), signal−to−noise ratio, the region descriptor for the source and a number of parameters that depend on the detect method employed. The Detect manual gives a complete list of the file columns and their meanings in the "Data Product Description" sections.

The Using the Output of Detect Tools thread illustrates how to use the source list in CIAO data analysis.

History

14 Dec 2004	reviewed for CIAO 3.2: no changes
03 Jun 2005	updated links for CIAO 3.2 version of Detect manual
12 Dec 2005	reviewed for CIAO 3.3: no changes
01 Dec 2006	reviewed for CIAO 3.4: no changes

URL: http://cxc.harvard.edu/ciao/threads/detect_overview/ Last modified: 1 Dec 2006