CCLRC / Rutherford Appleton Laboratory
Particle Physics & Astronomy Research Council
Starlink Project
Starlink User Note 242

SUN/242.0

Norman Gray
24 August 2001

Autoastrometry for Mosaics

Software Version 0.5-9

Abstract

We describe the `autoastrom` package, which provides an interface to the `ASTROM` application. This creates a semi-automatic route to doing astrometry on CCD images.

Document information

Document date	24 August 2001
Last revised	25 August 2003
Version number	0
Distribution ID	v0-5-8
Software version	Software Version 0.5-9

Contents

1 Introduction

THIS IS BETA SOFTWARE. Some features work only partially or unreliably. It does not have all the functionality of a production release. The interface may well change.

Starlink's `ASTROM` application provides powerful astrometry facilities, for analysing astronomical images; it is, however, rather cumbersome to use. As described in SUN/5, `ASTROM` can:

- obtain the plate centre and plate scale of a CCD image, by comparing plate positions of objects with those in a reference catalogue, and fitting with both four- and six-component distortion models;
- with enough data, obtain cubic distortion and plate tilt, using seven- to nine-component models;
- if enough information is available, it will do the reductions in observed place, correcting for atmospheric refraction.

`autoastrom` provides a shell around `ASTROM` so that, as well as the core astrometric facilities of `ASTROM`, `autoastrom` will:

- work on a CCD image provided as an NDF, as long as it has at least rough astrometry (plate centre and scale);
- automatically download appropriate reference catalogue information from catalogue servers supported by the SkyCat library;
- insert the astrometric results into the original NDF, as a WCS component;
- alternatively or additionally make the astrometry available as a set of FITS-WCS header cards.

2 Usage

Summary:

```
autoastrom [options...] <NDF>
```

The options available are described in Table 1. String or integer values are given in the form `--catalogue=usno@eso`; those marked `switch`, such as `--insert`, can be specified as either `--insert` (meaning true) or `--noinsert` (false).

Those options which take values can have those values specified as either `--option=value` or `--option value`. The latter form is rather unfortunate (but currently unavoidable), since it means that if you *forget* to give a value, it gobbles the next option on the line as its value. The options of type `switch` below take no value; *all* of the others do.

Option	Type	Description
–catalogue	string	SkyCat name of online catalogue to use
–ccdcatalogue	string	CCD catalogue
–defects	string	Remove CCD defects
–help	switch	Print usage information and exit
–insert	switch	insert WCS info into NDF at end?
–keepfits	string	retain WCS in this file (default if empty)
–keeptemps	switch	keep temporary files?
–match	string	choose matching algorithm
–matchcatalogue	string	dump catalogue of matches to this file
–maxobj	integer	maximum number of objects to process
–messages	switch	show messages from monoliths?
–obsdata	string	provide observation information, including WCS information
–skycatconfig	string	SkyCat configuration file
–temp	string	name of temporary directory
–timeout	integer	monolith timeout
–verbose	switch	verbosity
–version	switch	print version number and exit

Table 1: `autoastrom` options. Further details are in *Section 2.4*

2.1 Specifying positions

To work, `autoastrom` requires some initial estimate of the centre of the CCD frame. You can provide this with a WCS component in the NDF, or with a FITS (NDF-) extension, or with approximate astrometry given on the command line. By default, `autoastrom` searches for a WCS component, then a FITS extension, and fails if it finds neither. Because `autoastrom` ultimately sits on top of the AST library, it is able to make sense of a large variety of embedded FITS information, and get its initial astrometry from such pointing information.

You control this process using the –obsdata option.

If you wish to insert an initial approximate WCS component in the NDF, you can best do this using GAIA.

The initial calibration does not have to be particularly accurate; it need only be accurate enough that `autoastrom` is able to make a query to a catalogue server that will return a set of catalogue objects that has a substantial overlap with the imaged area of sky. The query covers the region of sky which maps to the four corners of the image, based on the initial astrometry, plus a small extra margin.

The default image scale is 1 arcsec per pixel. You will need to specify the scale if the actual scale is significantly different from this, as errors in this scale result in substantial errors in the patch of sky requested from the server. It is better to choose too small a scale than too large – that way, the area which `autoastrom` searches for will tend to lie within the observed area, rather than greatly overlap it.

`autoastrom` assumes that north is along the y-axis. The matching algorithms are generally insensitive to this angle. However, the only case where you need to specify this is if the CCD is significantly longer than it is wide, since in this case a wrong orientation would cause `autoastrom` to search for a patch of sky which was a different shape from the patch covered by the CCD. Similarly, you should specify any image inversion if you know it, but it is not always necessary.

2.2 Results

Results can be inserted into the input NDF (see *Section 2.4.5*), or else retained as a standalone FITS-WCS file (see *Section 2.4.6*). You can choose both, either, or even choose neither and throw the results away (strange person!).

The FITS files which `autoastrom` produces are in fact generated by `ASTROM`. As discussed in SUN/5, these employ a slight extension to the FITS-WCS standards, since they necessarily include distortion information which has been deferred until paper˜III, which is still at present (August 2003) in draft. Nonetheless, they are perfectly conformant. If, however, `autoastrom` had to be configured with support for an *old* version of AST, then it might have had to generate FITS headers which conformed to the drafts of paper˜II, and thus do not conform to the final version – such FITS files should not be used outside of this application, and it would be best to update your version of AST to a current one, and reinstall `autoastrom`.

The calculated astrometry is inserted into the input NDF unless you suppress it with the `--noinsert` option. If you need to insert this by hand, into this or another NDF, you can do so with KAPPA's wcscopy command (you may need to initialise the CONVERT package, so that the conversions between FITS and NDF will happen automatically).

For further details, see the FITS standard [nost-100-2], and the two published FITS-WCS papers, [fitswcs1] and [fitswcs2]; the web pages which cover the FITS-WCS process are at `<http://www.cv.nrao.edu/fits/documents/wcs/wcs.html>`.

2.3 Web proxies

In order to perform catalogue lookups, `autoastrom` needs to make HTTP (Web) queries. If your site enforces the use of proxies to get access to the web, these lookups will fail, and `autoastrom` will simply wait uselessly until its queries time out.

You can tell `autoastrom` how to use the proxies by using the standard `http_proxy` environment variable, which you set through something like

```
csh% setenv http_proxy http://wwwcache.example.ac.uk:8080/
```

or

```
sh% http_proxy=http://www.example.ac.uk:8080/; export http_proxy
```

depending on whether you use a csh-like (tcsh) or sh-like (bash, zsh) shell.

Your local computing service can advise you on the appropriate value for this environment variable, or you can examine the appropriate part of the configuration of your web browser.

2.4 Options

The options are either switches, or have a string or integer argument.

The verbosity option is an example of a switch: it can be given as `--verbose` to make the program verbose, or `--noverbose` to make it quiet.

Some options take suboptions. For example, the `--obsdata` option might be given as

```
--obsdata=ra=12:34:56,dec=-10:0:0
```

where the comma-separated list of suboptions is given as the `--obsdata` option argument.

There are no optional arguments: if an option is documented as having an argument (that is, if it is not a switch), an argument must be provided. The defaults documented below apply only if the option is *not* supplied.

2.4.1 –catalogue

This is the name of the online catalogue to use to find reference stars. The argument should be one of the catalogue names recognised by SkyCat[1]. The list of available catalogues depends on the configuration file used (see *Section 2.4.13*); the '`short_name`' entries in the file are the allowable values of this option.

Type: string; default (presently): 'usno@eso'

2.4.2 –ccdcatalogue

Specifies a pre-existing catalogue of objects in the CCD frame. The format is that produced as output by SExtractor when `CATALOG_TYPE` is set to `ASCII_HEAD`. The file starts with a sequence of lines of the form

```
# <colnum> <colname> <optional comment>
```

[1] `http://archive.eso.org/skycat/`

The *colnum* numbers run from 1 to the number of columns; the *colname* is one of a set of column named defined in the `sextractor` documentation; and *optional comment* is some (ignored) annotation. The data immediately follows, in rows with the given number of columns in them. In each of these rows, any text following a `#` is ignored, and may be used for other annotation.

You control which columns SExtractor generates by specifying them in the SExtractor `.param` file. The catalogue must have all of the fields `NUMBER`, `FLUX_ISO`, `X_IMAGE`, `Y_IMAGE`, `A_IMAGE`, `B_IMAGE`, `X2_IMAGE`, `Y2_IMAGE`, `ERRX2_IMAGE`, `ERRY2_IMAGE`, `ISOAREA_IMAGE`, of which `X2_IMAGE`, `Y2_IMAGE`, `A_IMAGE`, `B_IMAGE`, `ERRX2_IMAGE`, `ERRY2_IMAGE` are not generated by default.

For further details about SExtractor, and details of these required columns, see *Section 4*

If you include the option `--keeptemps` to avoid deleting the temporary work directory, you will be able to scavenge the extractor output from there if this is useful to you.

Type: string; default: none

2.4.3 –defects

If this option is present, then when the application extracts the list of objects from the CCD, it will try to remove CCD blemishes. The algorithm is currently rather crude, and will likely change in future.

Some of the 'objects' detected by EXTRACTOR are in fact CCD defects, or readout errors, or the like. These are very bright, so they can confuse a matching program which examines only or preferentially the brightest objects.

By default, `autoastrom` will warn of the existence of anything it thinks is a defect, based on a plausible heuristic, and you can control this using this option. The options takes a list of keywords, which it processes as shown in Table 2.

Keyword	Description
ignore	Completely ignore defects.
warn	Warn about possible defects, but do nothing further. This is the default.
remove	Remove any suspected defects from the catalogue of CCD objects.
badness	Provide the threshold for defect removal and warnings. Any objects with a badness greater than the value specified here are noted or removed. The default threshold is 1.

Table 2: Keywords for `--defects` option

The heuristic works by assigning a 'badness' to each object on the CCD. Objects with a position variance, $\langle x^2 \rangle$ or $\langle y^2 \rangle$, smaller than one pixel, and objects whose flux density (counts/pixel) is significantly higher than the average, are given high scores. One can observe that line and point defects score high with this, with a badness greater than 1, but this is not completely reliable.

We can afford a few uncaught defects, and we can afford to discard a few real, but very small, sources, since the match algorithms will generally simply ignore these. What we need to avoid is a CCD catalogue which is dominated by bright sources which have no counterpart on the sky.

We emphasise that this defect-removal algorithm is not particularly sophisticated – if the object-extraction is producing spurious objects, you may need to mask the defects out by hand, using GAIA or similar.

If you have any observations on the reliability, or indeed usefulness, of this option, the author would be interested to receive them.

This option is not always necessary when used with the default FINDOFF matching algorithm, but it is vital when used with the 'match' matching algorithm (see *Section 2.4.8*), since that algorithm uses only the brightest objects detected.

Type: string; default: `--defects=warn`

2.4.4 –help

Print usage information, and defaults, and exit.

Type: switch; default: false.

2.4.5 –insert

If true, the final fit to the astrometry is inserted into the input NDF as an AST WCS component. If false (`--noinsert`), the insertion is not done, so presumably you have decided to handle the WCS information in a different way, and have set the `--keepfits` option to retain the WCS information.

Type: switch; default: true.

2.4.6 –keepfits

The normal mode of operation is for the script to insert the final astrometry into the input NDF, as an AST WCS component (see *Section 2.4.5*). Instead, or additionally, you can save this information as a FITS-WCS file, named by this option.

Type: switch; default: false.

2.4.7 –keeptemps

If true, temporary files will not be deleted at the end of processing. This is useful for debugging, both at the low level when there is something wrong with the script itself, but also when you wish to examine the sequence of calls to `findoff` and `ASTROM`, or examine the `ASTROM` input and output files.

All the temporary files are created in a temporary directory which is reported during the script's execution.

Type: switch; default: true (but this *will* change)

2.4.8 –match

Specifies an alternative matching algorithm. The default algorithm is 'match', which is an implementation of the FOCAS matching algorithm [valdes95] by Michael W Richmond [match-home]. A version of this is distributed with `autoastrom`, but it should be compatible with any later version.

Alternatively, you can use the 'findoff' application, which is part of CCDPACK. This does work, but is rather slow, and is completely thrown by a CCD image with unequal X and Y scales, making it essentially unusable if the image NDF does not have some astrometry already, or if you try to specify a position with `--obsdata` (see *Section 2.4.12*). Since it has different characteristics, though, it may work in some circumstances where 'match' does not. Again, the author of `autoastrom` would welcome feedback on this point.

`autoastrom` supports 'match' through the more general plugin mechanism described in *Section 4.2*.

Type: string; default: 'match'.

2.4.9 –matchcatalogue

Specifies a file which receives a dump of the set of positions matched by the matching process. The file is formatted like a `sextractor` output file, with five columns, containing (1) running object number, (2 and 3) RA and Dec of the source on the sky, and (4 and 5) x and y positions of the source on the CCD.

Type: string; default: absent – no file written by default

2.4.10 –maxobj

The maximum number of objects to fetch from the catalogue server. The default is generous, and it's unlikely you'd need to change this.

Type: integer; default: 500

2.4.11 –messages

If true, the script will pass on messages from the Starlink applications which the script uses. These might be reassuring, but if you don't like the chatter, they can be suppressed with `--nomessages`.

Type: switch; default: true

2.4.12 –obsdata

Specifies a source for the observation data, including WCS information. The keyword has the multiple role of specifying the source of WCS information, supplying approximate WCS information directly, and supplying the observation data (time, observatory, temperature and pressure, and colour) which `ASTROM` needs if it is to attempt one of its higher-order fits. This information is only needed if it is missing from the NDF or any FITS extension it incorporates, and if you wish to attempt the slight increase in accuracy of the more elaborate fits.

Do not specify a keyword more than once with distinct values.

The value of the `--obsdata` option can specify approximate astrometry through a comma-separated list of `key=value` pairs. For example, to supply a value for the centre of a CCD, overriding any astrometry in the file, you might write

```
--obsdata=ra=14:24:00,dec=-12.34
```

This value indicates the centre of the CCD even when this is not the centre of distortion, which would be the case when, for example, the CCD represents an off-centre section of a Schmidt plate.

The program will take its astrometry from only one of the possible sources (see `source` below) – so that it will not, for example, take a position from the `obsdata` option and a plate scale from the file's own astrometry. By default, a position given here overrides any other WCS information; thus to supply a position as a backup, in case there is *no* astrometry elsewhere, you could write

```
--obsdata=source=AST:FITS:USER,ra=14:24:00,dec=-12.34
```

The available astrometry keywords are in Table 3.

Keyword	Description
source	This is a colon-separated list of sources of WCS information. The values may be 'AST', indicating that the information should come from the AST WCS component of the NDF, or 'FITS', indicating that it should come from any FITS extension there, or 'USER', indicating that the following `obsdata` keywords should be used. The default is `source=USER:AST:FITS`, so that any WCS information given in the `obsdata` keyword has precedence. The keywords are not case-sensitive. If the program gets through this list without finding any WCS information, it exits with an error.
ra	Right ascension of the centre of the pixel grid, in colon-separated HMS or decimal degrees. Required.
dec	Declination of the centre of the pixel grid, in colon-separated DMS or decimal degrees. Required.
angle	Position angle of the pixel grid. This is the rotation, in degrees anti-clockwise, of the declination or latitude axis with respect to the 2-axis of the data array. Default: 0.
scale	The plate scale, in units of arcsec/pixel. Default: 1.
invert	If true (that is, specified as `invert=1` or just `invert`), the axes are inverted; otherwise (`invert=0` or `noinvert`) the axes are unflipped. Default: 0.

Table 3: Keywords for the `--obsdata` option.

The position angle is the same as the AIPS `CROTA2` convention. Specifically (and by definition), these definitions map to the `CDn_n` conventions of [fitswcs2] through

$$\begin{aligned}
\texttt{CD1_1} &= \pm s\cos(\phi) \\
\texttt{CD1_2} &= \pm s\sin(\phi) \\
\texttt{CD2_1} &= -s\sin(\phi) \\
\texttt{CD2_2} &= s\cos(\phi),
\end{aligned}$$

where s is `scale`/3600 (ie, the scale in units of degrees/pixel), ϕ is the position angle `pa`, and $\pm$ is $+$ when `invert` is false, and $-$ when `invert` is true.

Additionally, there are keywords which specify observation data: time, station coordinates, meteorological data and colour. These are described in Table 4. For fuller details on these, see the `ASTROM` documentation, SUN/5. Note that the `source` keyword discussed above does not affect the priority of these keywords: if they are specified, they override any values obtained from the NDF or its FITS extensions.

Keyword	Description
time	An observation time, given as a Julian epoch (format `r`), or a local sidereal time (format `i:i`) or UT (`i:i:i:i:r` specifying four-digit year, month, day, hours and minutes).
obs	An observation station, given either as one of the SLALIB observatory codes; or else in the format `i:r:i:r[:r]`, giving longitude, latitude and optional height. Longitudes are east longitudes – west longitudes may be given either as minus degrees or longitudes greater than 180.
met	Temperature and pressure at the telescope, in degrees Kelvin and millibars. The defaults are 278K and a pressure computed from the observatory height. Format `r[:r]`.
col	The effective colour of the observations, as a wavelength in nanometres. The default is 500nm.

Table 4: Observation data keywords. In these specifications, `i` represents an integer, `r` a real, optional entries are in `[...]`, and the separator `:` may be either a colon or whitespace.

Type: string; defaults: see above.

2.4.13 –skycatconfig

The SkyCat library has a default configuration file, which you can examine at `<http://archive.eso.org/skycat/skycat2.0.cfg>`. For some applications, it may be appropriate to use an alternative configuration file. If so, you can point to it using this option. For more information on SkyCat, catalogues, and the configuration file, see the SkyCat FAQ[2].

Type: string; default: built-in defaults

2.4.14 –temp

This option, if present, specifies the directory to be used for temporary files.

This has a double function: firstly and most straightforwardly, it indicates where temporary files should be created if, for some reason, the default location is unsuitable; secondly, if the script finds there a file (from a previous run) which it was about to generate, it does not regenerate it, but instead simply reuses it. The first function may be useful in some circumstances; the second is primarily a debugging facility, is subtle, and its use is discouraged.

Type: string; default: generated

[2] `http://archive.eso.org/skycat/skycat-faq.html`

2.4.15 –timeout

If present, this specifies how long, in seconds, the script should wait for its slave applications to complete. The default is three minutes. In some circumstances – such as if you have a huge number of stars to match up – this might be too short, so that the script loses patience before the slave application has finished, and produces a rather messy error message. In this circumstance, you can increase the timeout here.

Type: integer; default: 180

2.4.16 –verbose

If present, the script produces a good deal of chatter to the standard error; if false (`--noverbose`), this is suppressed.

Type: switch; default: false

2.4.17 –version

If present, write the version number to the standard output and exit.

Type: switch; default: false.

3 Building the software

Build the software by unpacking the distribution tar archive, and using the standard `./mk build` to build it, then `./mk install` to install it.

If you wish to use a non-default Starlink tree to build the software (that is, not the standard `/star`), then you can specify this by setting the environment variable `STARLINK`: for example `setenv STARLINK /stardev`. If you need to use more than one Starlink tree (perhaps because you have installed updated versions of some software outside the main tree; specifically, note that `autoastrom` requires AST version 1.8-1 or better), then use the `STARLINKS` variable, setting it to a colon-separated list of directories: for example `setenv STARLINKS /path/to/my/star:/star` (this facility is intended primarily for developers, and you really shouldn't need to use it in 'normal' building of the software).

4 Auxiliary software

The script uses other Starlink software tools. Although the use of these tools should be transparent, it might be of use to have some indication of sources of further detail. All of these tools should be up-to-date if you are using a Starlink Summer 2003 distribution or later.

The script is largely a wrapper for `ASTROM`. This is documented fully in SUN/5.

The pattern matching facilities which the script depends on come from the `findoff` application within CCDPACK. There are further details in SUN/139. You need version 4.0-1 at least, and the program does, of course, respect any setting of the environment variable `CCDPACK_DIR`).

The WCS information is largely handled by ATOOLS. This does not have a SUN yet. You need ATOOLS 1.3 or better (`ATOOLS_DIR`).

`autoastrom` requires AST 1.8-1 or better to build. Depending on precisely which earlier version of AST you have, the moggy tool will fail to compile, or if it compiles, fail to link. If you have AST installed in a non-standard place, you should use the `STARLINKS` environment variable at build time (see *Section 3*), to tell the build script where to find it.

It also needs StarPerl (see SUN/228).

The concepts underlying WCS, and its interaction with FITS-WCS, are described in the AST documentation. See SUN/211.

Extraction of the object catalogues from the CCD is done by Extractor, which is the Starlink version of Emmanuel Bertin's SExtractor. Extractor is documented in SUN/226. See also the SExtractor manual MUD/165[3], the home of SExtractor on the web [sextractor-web], the distribution location [sextractor-ftp] and the original SExtractor article [sextractor-article].

If you need to supply the initial astrometry which this script requires, you can do that using GAIA. This is documented in SUN/214, but it's very easy to use, and you might be best to simply start it up (give the command `gaia image_name`, presuming you have already done the initialisation to use Starlink software) and experiment.

4.1 Environment

The environment variable `AUTOASTROM_DIR` points to the installed location of the software and its auxiliary programs.

If you need to use a proxy to get to the web, then you should define the environment variable `http_proxy` to the appropriate value for your site.

The program moggy uses the ESO SkyCat library for its catalogue access. However, note that, for the sake of consistency, it does *not* honour the SkyCat `CATCONFIG` variable directly. If you need to specify an alternative SkyCat configuration file, you should do that with the `--skycatconfig` option instead (see *Section 2.4.13*).

4.2 Plugins

`autoastrom` has a flexible extension mechanism, implemented through plugins. At present, the only area of the code which takes advantage of this is the part which finds matches between the objects found on the CCD and those in the downloaded object catalogue.

[3] `file:/star/docs/mud165.ps`

The program searches the directory pointed to by the environment variable `AUTOASTROM_DIR` (see *Section 4.1*) for files which match the pattern `plugin-*.pl`, and reads in all it finds. Each such file must define a single subroutine, and assign a reference to that subroutine to the appropriate member of the hash `%helpers`.

`autoastrom` comes with a single plugin, to use Michael W Richmond's `match` application for object matching. To use this, specify 'match' as the argument to the 'match' option, as `--match=match` (see *Section 2.4.8*). The plugin checks at runtime to see if the `match` application is present in the directory pointed to by `AUTOASTROM_DIR`, and the plugin will not be installed if the application is not found. `autoastrom` has been tested with match versions 0.3, 0.4 and 0.7.

If you wish to add further plugins, see the documentation for the function `match_positions`. That is given a plugin name, say 'mymatch', and searches for the entry `$helpers{plugin-match-mymatch}`.

5 Future developments

As a longer-term goal, it *might* be possible to automate even the discovery of the initial pointing, if there is some reasonable way of estimating plate-centre and plate-scale information *ex nihil*, from the pattern of objects in the plate.

At present, only one CCD can be processed at a time. It would be useful if the application could process images from multiple CCDs, arranged in a mosaic.

5.1 Restrictions and limitations

The script inherits some limitations from `ASTROM`, and some from CCDPACK's `findoff`.

The limitations due to `ASTROM` are as follows (quoted from SUN/5, sect. 5):

`ASTROM` aims to deliver results better than 1 arcsec from typical Schmidt plate measurements, and better than 0.1 arcsec from carefully measured JKT and AAT plates *etc.* Astrometric specialists will, nonetheless, be aware of a number of shortcomings, including the following:

- The fit is limited to a 6-coefficient linear model plus cubic distortion and plate tilt. Colour effects – arising for example from chromatic aberrations in the camera optics – are not allowed for, no magnitude or image shape terms are included in the model, and the refraction cannot be adjusted automatically.
- The zonal distortions of the reference catalogues are neglected.
- There is no provision for the simultaneous fitting of more than one plate. This prevents an extended area being modelled via overlapping plates, and the determination of proper motion and parallax from plates taken at different epochs.
- Only rudimentary error information is produced.

Despite these limitations, which stem mainly from the need for simplicity of use, the accuracy of the result tends in practice to be dominated by the quality of the input data rather than by `ASTROM` itself.

Further, `ASTROM` requires that at least 10 reference stars are available. Since `autoastrom` obtains its reference stars from well-stocked catalogues, this is not a problem in practice.

The restrictions arising from `findoff` are similarly slight, and should not be a problem if the initial astrometry is good enough. It's not at present *completely* clear exactly what 'good enough' means.

6 Release Notes

6.1 Version 0.5-8

This is release 0.5-8, 2003 August 25

Changes in this version:

- Various stability improvements
- Added the –ccdcatalogue and –matchcatalogue options
- Documentation updated and tidied

6.2 Version 0.5-7

This is release 0.5-7, 2003 July 28

Changes in this version:

- In 0.5-7: Fixed crashing error in 0.5-6; now works without 'match' being installed; now distribute 'match' (`<http://spiff.rit.edu/match/>`) with autoastrom.
- Fixed typo in function name which made some informational messages fatal.
- Extractor sometimes claims to detect objects with a negative flux – such objects are now silently ignored.
- Assorted changes to configuration to make it more robust.
- All of autoastrom that originates purely within Starlink has been released under the GPL. This does NOT include the bundled version of ASTROM, nor the decompose˙transform routine in autoastrom.pm, both of which are Copyright 1995 Patrick Wallace. See README.Licence.
- Uses bundled ASTROM 3.7, which emits FITS WCS headers which extend the draft 'Paper II' proposals. See ASTROM docs for discussion.

6.3 Version 0.5-3

This is release 0.5-3, 2003 May 2

Changes since version 0.5-2:

- Better behaved in the case where an input file has no date or time information.
- Uses version 3.7 of ASTROM

6.4 Version 0.5-2

This is release 0.5-2, 2003 April 8

Changes since version 0.5-1:

- Fixed sexagesimal-to-decimal bug, so that negative Decs are now converted correctly (thanks to Peter Draper for diagnosing this).
- More adjustments to configuration code, to make it more robust against potential version-skew (it now builds correctly on RH8).

6.5 Version 0.5-1

This is release 0.5-1, 2003 February 20

Changes since version 0.5:

- The bulk of the code is now under the terms of the GPL. However, not all of it is: see the notes in `README.Licence`.
- Fixed typo in function name which made some informational messages fatal.
- Extractor sometimes claims to detect objects with a negative flux – such objects are now silently ignored.
- Assorted changes to configuration to make it more robust.

6.6 Version 0.5

This is release 0.5.

THIS IS BETA SOFTWARE. Some features work only partially or unreliably. It does not have all the functionality of a production release. The interface may well change.

See the version 0.4 release notes for building guidelines.

Changes since version 0.4:

- `autoastrom` now supports plugins, and there is one plugin supplied, for the 'match' matching application available at `<http://acd188a-005.rit.edu/match/>`. This is not distributed with `autoastrom`, but it is easy to download, build, and install in the directory `$AUTOASTROM_DIR`.
- Now, if you type Control-C, `autoastrom` will delete the monoliths it is using, and tidy up after itself.

6.7 Version 0.4

This is release 0.4, Date: 2001/11/05 15:58:21

THIS IS BETA SOFTWARE. Some features work only partially or unreliably. It does not have all the functionality of a production release. The interface may well change.

At present, you'll probably need to define the `STARLINK` environment variable to point to the /stardev tree. If you have `CCDPACK_DIR` and `ATOOLS_DIR` defined, make sure that these are pointing to at least versions 4.0-1 and 1.3 respectively.

To build, give the sequence of commands

```
./mk build
./mk install
```

There's a pre-installation regression test:

```
# Define AUTOASTROM_DIR to point to a temporary directory, which
# must exist.
./mk test-install
./mk pretest
```

A couple of regression tests fail on DUX, apparently due to rounding errors somewhere.

Autoastrom requires:

AST 1.8-1
Depending on precisely which earlier version of AST you have, the moggy tool will fail to compile, or if it compiles, fail to link. If you have AST installed in a non-standard place, you should use the `STARLINK` environment variable at build time (see *Section 3*), to tell the build script where to find it.

ATOOLS 1.3
This will be on the next Starlink CD, but is available at present from David Berry.

CCDPACK 4.0-1
This will be on the next Starlink CD, but is available at present from Mark Taylor.

Changes since v0.3:

- The distribution builds successfully on Alpha (with the cxx compiler), Solaris (with CC) and Linux (with gcc). There is no post-installation test yet, but you can run a pre-installation self-test with `./mk pretest`.
- The astrometry can now come from a FITS extension or from the commandline (`--obsdata` switch).
- Handles FITS dates more robustly, specifically including pre-2000 FITS dates.
- Notes in the documentation and within the program chatter about `http_proxy`.
- Fixed a silly bug which means that the script does now work for negative Decs.
- Report more coherent information to the user.

6.8 Version 0.3

This is release 0.3, Date: 2001/08/29 17:09:52

THIS IS BETA SOFTWARE. Some features work only partially or unreliably. It does not have all the functionality of a production release. The interface may well change.

This release is still lacking some functionality. In particular

- the initial approximate astrometry must be supplied as a WCS component in the input NDF – a future version will allow this to be either specified on the command line or picked up from any FITS headers in the file;
- this can handle only single CCDs, rather than mosaics;
- I need to do some experimentation to discover how robust the whole kit is to errors in the initial astrometry.

Bibliography

References

[sextractor-web] Emmanuel Bertin. Sextractor program [online, cited 15 August 2003]. URL: `<http://www.tass-survey.org/tass/software/software.html#sextract>`.

[sextractor-article] Emmanuel Bertin. Sextractor: Software for source extraction. *Astron. Astrophys. Suppl. Ser.*, 117:393, 1996.

[sextractor-ftp] Emmanuel Bertin. Sextractor distribution [online]. 2001 [cited 15 August 2003]. URL: `<ftp://ftp.iap.fr/pub/from_users/bertin/sextractor/>`.

[fitswcs2] Mark Calabretta and Eric W Greisen. Representations of celestial coordinates in FITS ("Paper II"). *Astron. Astrophys.*, 395:1077–1122, 2002. URL: `<http://www.cv.nrao.edu/fits/documents/wcs/wcs.html>`. URL: `<http://www.cv.nrao.edu/fits/documents/wcs/wcs.html>`.

[fitswcs1] Eric W Greisen and Mark Calabretta. Representations of world coordinates in FITS ("Paper I"). *Astron. Astrophys.*, 395:1061–1075, 2002. URL: `<http://www.cv.nrao.edu/fits/documents/wcs/wcs.html>`. URL: `<http://www.cv.nrao.edu/fits/documents/wcs/wcs.html>`.

[nost-100-2] R. J. Hanisch, A. Farris, E. W. Greisen, W. D. Pence, B. M. Schlesinger, P. J. Teuben, R. W. Thompson, and A. Warnock. Definition of the flexible image transport system (FITS). *Astron. Astrophys.*, 376:359–380, September 2001. URL: `<http://ukads.nottingham.ac.uk/cgi-bin/nph-bib_query?bibcode=2001A%26A..%.376..359H&db_key=AST>`. URL: `<http://ukads.nottingham.ac.uk/cgi-bin/nph-bib_query?bibcode=2001A%26A..%.376..359H&db_key=AST>`.

[match-home] Michael W. Richmond. Matching lists of stars [online]. 2003 [cited August 2003]. URL: `<http://spiff.rit.edu/match/>`.

[valdes95] Francisco G. Valdes, Luis E. Campusano, Juan D. Velasquez, and Peter B. Stetson. FOCAS automatic catalog matching algorithms. *Publications of the Astronomical Society of the Pacific*, 107:1119, November 1995. URL: `<http://adsabs.harvard.edu/cgi-bin/nph-bib_query?bibcode=1995PASP..107.1% 119V&db_key=AST&high=39463d35aa24090>`. URL: `<http://adsabs.harvard.edu/cgi-bin/nph-bib_query?bibcode=1995PASP..107.1% 119V&db_key=AST&high=39463d35aa24090>`.

Change history

Distribution v0-5-8

Norman Gray, 25 August 2003

Stability improvements; extra options; enhanced documentation.

Distribution v0-5-2

Norman Gray, 8 April 2003

Bugfix release

Distribution v0-5-1

Norman Gray, 20 February 2003

Release of v0.5-1

Distribution v0-4

Norman Gray, 5 November 2001

Release of v0.4, tagged autoastrom-0-4.

Version 0

Norman Gray, 24 August 2001

Changes in version 0

Norman Gray, 24 August 2001

Initial release