SUN/242.1

Starlink Project
Starlink User Note 242.1

Norman Gray & Brad Cavanagh

2006 April 19

Autoastrometry for Mosaics
1.5
User's Guide

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.

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 well 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
`-bestfitlog`	string	File to hold information about the best fit
`-catalogue`	string	SkyCat name of online catalogue to use
`-ccdcatalogue`	string	Catalogue to use in place of extracting objects from CCD
`-defects`	string	Remove CCD defects
`-detectedcatalogue`	string	Dump catalogue of detected objects with updated WCS information to this file
`-help`	switch	Print usage information and exit
`-insert`	switch	Insert WCS into NDF at end?
`-iterrms_abs`	real	Absolute RMS level to reach to stop iterations
`-iterrms_diff`	real	Differential RMS level to reach to stop iterations
`-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
`-maxfit`	integer	Maximum number of fit parameter to use
`-maxiter`	integer	Maximum number of iterations to perform
`-maxobj_corr`	integer	Maximum number of objects to use for correlation
`-maxobj_image`	integer	Maximum number of objects to use from CCD
`-maxobj_query`	integer	Maximum number of objects to use from SkyCat query
`-messages`	switch	Show Starlink application messages?
`-obsdata`	string	Provide observation information, including WCS information
`-skycatconfig`	string	SkyCat configuration file
`-skycatcatalogue_in`	string	Catalogue to use in place of SkyCat query
`-skycatcatalogue_out`	string	Dump SkyCat query result catalogue to this 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 2.5

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 Iterations

autoastrom comes to its solution iteratively. After acquiring both catalogues – the SkyCat catalogue and the detected objects catalogue – it enters an iteration loop, starting off by matching objects between the two catalogues. It then calculates an astrometric solution, and if certain criteria have been met, the calculated solution is used. Otherwise, the iteration loop is entered again. This has the effect that more objects are usually matched between the two catalogues after an improved astrometric solution has been found, which leads to a more accurate solution.

Iterations can be halted in one of three ways: by reaching an absolute RMS level in the astrometric solution; by reaching a limit in the difference between the current astrometric solution's RMS and the prior solution's; and by reaching a maximum number of iterations. By default autoastrom is set to halt after ten iterations. For more information on modifying these values, see Sections 2.5.8, 2.5.9, and 2.5.16.

2.3 Results

Results can be inserted into the input NDF (see Section 2.5.7), or else retained as a standalone FITS-WCS file (see Section 2.5.10). 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 [6]. 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 [7] and the three published FITS-WCS papers, [5, FITS WCS Paper I], [4, FITS WCS Paper II] and [6, FITS WCS Paper III]; the web pages which cover the FITS-WCS process are at `http://www.cv.nrao.edu/fits/documents/wcs/wcs.html`.

2.4 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.

2.5 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.

2.5.1 –bestfitlog

When ASTROM finds a solution, information about the best fit is written to this file. This file is a textfile containing the number of terms, the central right ascension and declination, the average pixel RMS, the distortion factor (when applicable) and the FITS file containing the WCS headers for all of the fits found. For the best fit, this file also lists the number of stars used in the fit, the mean plate scale in arcseconds, and the RMS in fitted r, x, and y.

Type: string; default: none.

2.5.2 –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. The list of available catalogues depends on the configuration file used (see Section 2.5.22); the 'short_name' entries in the file are the allowable values of this option.

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

2.5.3 –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

```
# <column> <colname> <optional comment>
```

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 3.

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

2.5.4 –defects

This option is currently unsupported but will be reinstated at some point in the future. The following documentation is valid for older versions of autoastrom.

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.

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

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.

The heuristic works by assigning a 'badness' to each object on the CCD. Objects with a position variance, $< x^2 >$ or $< y^2 >$, 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 2.5.13), since that algorithm uses only the brightest objects detected.

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

2.5.5 –detectedcatalogue

Specifies a file which receives a dump of the objects detected in the NDF, with their astrometric positions corrected. The file is in the Cluster format with thirteen columns: a zero in the first column, followed by the item's ID, the right ascension and declination in space-separated format, the x and y position of the source on the CCD, an instrumental magnitude, the error in the instrumental magnitude, and extraction flags.

Type: string; default: none.

2.5.6 –help

Print usage information, and defaults, and exit.

Type: switch; default: false.

2.5.7 –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.

2.5.8 –iterrms_abs

When the absolute RMS of a fit reaches this value, iterations will cease and the current WCS will be used. For more information on `autoastrom`'s iteration process, see Section 2.2.

Type: real; default; none.

2.5.9 –iterrms_diff

When the difference between the current RMS and the previous fit's RMS reaches this value, iterations will cease and the current WCS will be used. For more information on `autoastrom`'s iteration process, see Section 2.2.

Type: real; default; none.

2.5.10 –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 2.5.7). Instead, or additionally, you can save this information as a FITS-WCS file, named by this option.

Type: switch; default: false.

2.5.11 –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: false

2.5.12 –man

If true, print a man page and exit.

Type: switch; default: false

2.5.13 –match

Specifies an alternative matching algorithm. The default algorithm is 'match', which is an implementation of the FOCAS matching algorithm [9] by Michael W Richmond [8]. 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 2.5.21). 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 3.2.

Type: string; default: 'match'.

2.5.14 –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.

2.5.15 –maxfit

The maximum order fit to use. If not set, then 9 will be used.

Type: integer; default: 9.

2.5.16 –maxiter

The maximum number of iterations to perform. For more information on `autoastrom`'s iteration process, see Section 2.2.

Type: integer; default: 10.

2.5.17 –maxobj_corr

The maximum number of objects to use from either catalogue to perform correlation on. This will take the brightest objects in the catalogues.

Type: integer; default: 500.

2.5.18 –maxobj_image

The maximum number of objects to use from the image catalogue. This will take the brightest objects in the catalogue.

Type: integer; default: 500;

2.5.19 –maxobj_query

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

2.5.20 –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

2.5.21 –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
```

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 `-obsdata` option

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.

The available astrometry keywords are in Table 3.

The position angle is the same as the AIPS `CROTA2` convention. Specifically (and by definition) these definitions map to the `CDn_n` conventions of [4] 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 unites 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 ASTROM. 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.

Type: string; defaults: see above.

2.5.22 –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.

Type: string; default: built-in defaults.

2.5.23 –skycatcatalogue_in

Specifies a pre-existing catalogue of objects as retrieved from SkyCat. The format of this catalogue must be the Cluster format as described in Section 2.5.5.

Type: string; default: none.

2.5.24 –skycatcatalogue_out

After the SkyCat query has been performed, the catalogue returned can be saved to disk in this file. The format of this catalogue will be the Cluster format as described in Section 2.5.5.

If used in conjunction with the `-skycatcatalogue_in` option (see Section 2.5.23), this can be used as a poor-man's cache, avoiding potentially timely SkyCat queries.

Type: string; default: none.

2.5.25 –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.

2.5.26 –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.

2.5.27 –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.

2.5.28 –version

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

Type: switch; default: false.

3 Auxiliary Software

The autoastrom script relies primarily on the Starlink::Autoastrom Perl module, which relies on a number of other Perl modules which, in turn, rely on a number of Starlink software tools. All of these tools should be up-to-date if you are using a Starlink distribution as obtained from the Starlink CVS repository.

The pattern matching facilities which autoastrom relies 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 respects any setting of the CCDPACK_DIR environment variable.

autoastrom requires AST 1.8-1 or better. See SUN/211 on information about AST.

Object detection is performed 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, the home of SExtractor on the web [3], the distribution location [2], and the original SExtractor article [1].

If you need to supply the inital astrometry which autoastrom 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.

3.1 Environment

The environment variable ASTROM_DIR points to the installed location of ASTROM, CCDPACK_DIR points to CCDPACK, and EXTRACTOR_DIR points to SExtractor. All of these environment variables will be set up for you when you do the initialisation to use Starlink software.

3.2 Plugins

autoastrom has, through its use of the Astro::Correlate Perl module, a flexible extension mechanism for the matching of objects between those found on the CCD and those downloaded from an online catalogue.

For further information on creating new plugins, see the documentation for the Astro::Correlate Perl module, which can be found on CPAN.

4 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.

4.1 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, Section 5):

> ASTROM aims to deliver results better than 1″ from typical Schmidt plate measurements, and better than 0.1″ 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.

References

[1] Bertin, E. Sextractor: Software for source extraction. *Astron. Astrophys. Suppl. Ser.*, 117:393, 1996. 3

[2] Bertin, E. Sextractor distribution [online]. 2001 [cited 25 April 2005]. URL: <`ftp://ftp.iap.fr/pub/from_users/bertin/sextractor/`>. 3

[3] Bertin, E. SExtractor program [online, cited 25 April 2005] URL: <`http://terapix.iap.fr/rubrique.php?id_rubrique=91/`>. 3

[4] Calabretta, M.R. & Greisen, E.W. Representations of celestial coordinates in FITS ("Paper II"). *Astronom. Astrophys.*, 395:1077-1122, 2002. 2.3, 2.5.21

[5] Greisen, E.W. & Calabretta, M.R.. Representations of world coordinates in FITS ("Paper I"). *Astronom. Astrophys.*, 395:1061-1075, 2002. 2.3

[6] Greisen, E.W., Calabretta, M.R., Valdes, F.G., & Allen, S.L. Representations of spectral coordinates in FITS ("Paper III"). *Astronom. Astrophys.*, 446:747-771, 2006. 2.3

[7] Hanisch, R.J., Farris, A., Greisen, E.W., Pence, W.D., Schlesinger, B.M., Teuben, P.J., Thompson, R.W., & Warnock, A. Definition of the flexible image transport system (FITS). *Astron. Astrophys.*, 376:359-380, September 2001. 2.3

[8] Richmond, M.W. Matching lists of stars [online, cited April 2005] URL: <`http://spiff.rit.edu/match/`> 2.5.13

[9] Valdes, F.G., Campusano, L.E., Velasquez, J.D., & Stetson, P.B. FOCAS automatic catalog matching algorithms. *Publications of the Astronomical Society of the Pacific*, 107:1119, November 1995. 2.5.13