Tutorial para hacer astrometría con IRAF

Julio de 2010

R. Gil-Hutton

Se necesitan los paquetes **images.imcoords** y **noao.astcat**, y mi script **astrsol.cl**.

1) Verificar que en la cabecera de la imagen existen entradas para la RA y DEC.:

```
cl> hselect xx.fit ra,dec,epoch,ut,exptime yes
21:04:51        -30:39:04       2010.5                  06:55:45        300
```

2) desplegar la imagen con ds9:

```
cl> display xx.fit 1
....
```

3) al inicio se necesita tener un WCS aproximado para lo cual se debe resolver la placa de manera aproximada. Para ello selecciono 5 estrellas bien distribuidas y utilizando **rimcursor** o **imexamine** obtengo sus valores de (x,y).

Luego utilizo **ds9** para bajar una imagen del Digital Sky Survey (DSS): en el menu de ds9 ir a *analysis - Image Servers - Digital Sky Survey (DSS)*. En la ventana que aparece ingresar las coordenadas J2000 (**OJO:** los valores de RA y DEC de la cabecera pueden venir dados para la fecha asi que podrían tener que precesar!) y las dimensiones considerando aproximadamente un 120% del tamaño de la imagen.

Luego de identificadas las estrellas se encuentra con el cursor sus coordenadas RA y DEC aproximadas y se arma un archivo donde para cada objeto tenemos (x,y,ra,dec). Por ejemplo:

```
#    x        y          ar              dec         mag
230.54 419.38 21:04:31.97 -30:41:24.36 12.56
  79.14 321.07 21:04:22.87 -30:44:47.10 13.51
282.72 259.43 21:04:15.71 -30:40:26.27 11.96
400.59 168.66 21:04:06.04 -30:38:01.21 14.09
164.73 194.75 21:04:09.76 -30:43:03.83 12.70
232.45   42.25 21:03:54.19 -30:41:46.18 13.43
```

A este archivo lo llamamos *xx.dat*. Notese que el separador usado en RA y DEC es ':'.

4) Calculamos una primera aproximación obteniendo la solución de la placa mediante la tarea **ccmap** del paquete **images.imcoords**:

cl> ccmap xx.dat ccmap.db images=xx.fit results=xx.out xcolumn=1 ycolumn=2 lngcolumn=3 latcolumn=4 insystem=j2000 refpoint=coords up+ inter-

ccmap tiene posibilidades interactivas para encontrar la solución (inter+ en la línea de comando) pero para encontrar esta aproximación no son necesarias.

Esta tarea modifica la cabecera para incluir entradas con el objeto de definir el nuevo WCS (keywords en la cabecera: crval1,crval2,crpix1,crpix2,cd1_1,cd1_2,cd2_1,cd2_2). El archivo *ccmap.db* contiene la solución de la placa para pasar de pixels a coordenadas en el cielo.

5) Para probar la solución podemos marcar en la imagen las coordenadas RA y DEC que utilizamos para resolver la placa y que se encuentran en las columnas 3 y 4 del archivo *xx.dat*. Para marcar en la imagen las estrellas podemos usar la tarea **tvmark**, pero esta sólo interpreta pixels y no coordenadas en el cielo. Entonces, debemos utilizar la tarea **cctran** del paquete **images.imcoords** para transformar de un sistema al otro. Para ello hacemos:

cl> display xx.fit 1
cl> fields xx.dat 3,4 | cctran STDIN STDOUT ccmap.db xx.fit forward- | tvmark 1 STDIN mark=circle radii=2 color=204

La tarea **fields** extrae de un archivo (*xx.dat*) la informacion contenida en un cierto campo (aquí, columnas 3 y 4 que corresponden a RA y DEC). La salida es tomada por la tarea **cctran** que interpreta estos valores de RA y DEC y los convierte a pixels (de ahi el forward-). Finalmente, la salida de esta tarea es tomada por **tvmark** que dibuja en el frame 1 círculos de 2 pixels de radio y color rojo.

Es importante notar que debido a un bug de programación **ds9** no indica los valores del WCS cuando la imagen es desplegada por IRAF (si cuando se carga directamente desde **ds9** con el menu *File - Load*).

6) Para encontrar una solución definitiva a partir de esta solución aproximada debemos primero obtener un listado de estrellas de un catálogo astrométrico. Para eso vamos a utilizar algunas tareas del paquete **noao.astcat**.

Primero necesitamos conocer cuales son los catálogos disponibles. Para ello hacemos:

cl> aclist *
usnob1@noao
twomass@noao
twomass@irsa
twomass14@irsa
lan92@noao
usno2@cadc
usno1@cadc
gsc1@cadc
hipp@cadc
gsc2@stsci

que devuelve la lista de catálogos disponibles para trabajar. En este ejemplo usaremos el

catálogo usnob1@noao:

cl> agetcat xx.fit xx.fit.cat.1 catalogs=usnob1@noao

El archivo *xx.fit.cat.1* tiene la siguiente estructura:

```
# BEGIN CATALOG HEADER
# catdb astcat$lib/catdb.dat
# catname usnob1@noao
# nquery 4
#     ra 21:04:13.53 hours
#     dec -30:41:29.9 degrees
#     radius 5.1 minutes
#     qsystem J2000.0 INDEF
# type btext
# nheader 1
#     csystem J2000.0
# nfields 11
#     id 1 12 c NDEF %12s
#     ra 14 12 d hours %12.3H
#     dec 27 12 d degrees %12.2h
#     b1mag 40 5 r INDEF %5.2f
#     r1mag 46 5 r INDEF %5.2f
#     b2mag 52 5 r INDEF %5.2f
#     r2mag 58 5 r INDEF %5.2f
#     i2mag 64 5 r INDEF %5.2f
#     mura 70 6 r INDEF %6.1f
#     mudec 77 6 r INDEF %6.1f
# END CATALOG HEADER
#
0592.0896216        21:03:50.934-30:42:16.46 19.99  99.99  20.27  19.42  18.63    30.0      14.0
        8        4        10       0.08
0592.0896221        21:03:52.277-30:43:19.22 18.42  17.44  18.54  17.14  16.79     8.0      -2.0
        8        5        8        0.08
0592.0896226        21:03:53.020-30:43:51.02 99.99  99.99  20.02  20.03  18.82   280.0   712.0
        8        3        12       0.08
.........................
.........................
```

esta tarea baja de internet posiciones del catalogo indicado para objetos que se encuentran en el área de la imagen que estamos analizando y lo guarda en un archivo *xx.fit.cat.1*. Las primeras 21 líneas de ese archivo corresponden a la descripción del catálogo y después tenemos una línea para cada estrella con identificación, RA, DEC, y magnitudes en diferentes filtros.

Como el archivo es grande es mejor limitar por magnitud el número de objetos extraidos, asi que repetimos la operación:

cl> agetcat xx.fit xx.fit.cat.2 catalogs=usnob1@noao filter+ fsort=mag1 fexpr="r1mag <= 19.5"

En este caso, ordenamos la salida por la magnitud R (r1mag) y pedimos objetos más brillantes que magnitud 19.5. Hay que tener cuidado porque los campos de las cabeceras de los diferentes catálogos pueden cambiar al agregar o remover información de los mismos. En el peor de los casos cuando se incluye una condición con un campo inexistente la tarea no la considerará y devolverá todas los objetos.

En el caso que **agetcat** de error (depende de la conexión a internet, el host donde busca el catálogo, etc.) se puede bajar diréctamente el catálogo de *Vizier*. Para ello se abre un Browser cualquiera y se copia como dirección:

http://webviz.u-strasbg.fr/viz-bin/asu-tsv/?-source=USNO-B1&-c.ra=318.867&-c.dec=-16.4631&-c.bm=90/60&r1mag=%3C13&-out.max=unlimited

donde:

source=USNO-B1 es el catálogo
c.ra=318.867 es la AR aprox. del centro en grados
c.dec=-16.4631 es la DEC aprox. del centro en grados
c.bm=10/10 son los lados del área en minutos de arco
r1mag=%3C13 es un constraint que limita la magnitud (%3C es el código para "<", asi que corresponde a r1mag=<13)

Vizier devuelve un archivo con los objetos en el area solicitada que se debe guardar para continuar con la reducción (por ejemplo, con el nombre *xx.fit.cat.2*).

7) Para alinear interactivamente las estrellas de nuestra imagen con las posiciones obtenidas del catálogo vamos a utilizar un script que hice especialmente para esto llamado **astrsol**. Este script funciona de manera similar a **msctpeak** del paquete **mscfinder**, pero utiliza el sistema de coordenadas lógico de la imagen y no el físico como esta última. La ventaja de **astrsol** es que si la imagen con la que se quiere trabajar tiene un trim posiblemente la solución obtenida con **msctpeak** sea erronea. Lamentablemente aún no hay help para esta tarea pero si hay un listado de posibles funciones con "?" cuando esta corriendo.

Lo primero que se necesita es un listado con solo RA y DEC que se obtiene del catálogo que hicimos en el paso anterior:

```
cl> fields xx.fit.cat.2 2,3 > xx.fit.cat.3
```

A continuación cargamos la tarea y la hacemos correr. **Astrsol** desplegará la imagen en el frame 1 de **ds9**:

```
cl> task astrsol = ./astrsol.cl
cl> astrsol xx.fit xx.fit.cat.3 olddatab=ccmap.db newdatab=solucion.db box=15
...
...
```

Esta tarea marca en la imagen los objetos de catálogo que no estan centrados en **rojo**, los que si estan centrados en **verde** y otros objetos de interés en **cyan**. Como sólo estamos

desplegando el catálogo sobre la imagen y no tenemos equivalencia entre uno y otro, todos los círculos son rojos.

Para registrar una equivalencia entre una estrella y un objeto de catálogo tenemos que usar la tecla 'a' sobre la imagen de una estrella y la tarea registrará el círculo rojo mas cercano (y lo dibujará en verde). Si no existe un buen registro o es dudoso cuál círculo rojo es el más cercano podemos relacionar una estrella con un círculo usando la tecla 'r' al presionar primero en un círculo rojo y luego sobre una estrella. Si creemos que la diferencia entre el catálogo y la imagen es una traslación podemos utilizar la tecla 'g' o 't' que producen el efecto de utilizar 'a' o 'r', respectivamente, sobre todos los objetos de catálogo al mismo tiempo. Los círculos rojos originales no son borrados despues de encontrar una equivalencia con algún objeto debido a que para eso se requiere desplegar la imagen nuevamente.

El comando '?' nos da un listado de comandos disponibles. No hay inconveniente en hacer zoom hacia adelante o atras en la imagen durante el proceso interactivo de ajuste o, recentrarla para hacer un blink con otra imagen en el frame 2 o 3.

Una vez que tenemos unas cuantas estrellas centradas podemos presionar 'f' para encontrar un ajuste. La tarea hace correr interactivamente la tarea **ccmap** y nos muestra los gráficos de ajuste de diferentes parámetros. Presionando '?' sobre el gráfico nos da un listado de opciones. El gráfico inicial es una representación (x,y) de los objetos a ajustar. Presionando 'x', 'y', 'r', y 's' nos muestra los residuos en funcion de x o y. En estos gráficos uno puede presionar 'd' cerca de algun punto discordante (mal identificado, mal centrado o estrellas con gran movimiento propio) para borrarlo y luego presionar 'f' para recalcular. Cuando terminemos el ajuste presionando 'q' nos regresa a la imagen donde se muestra el ajuste obtenido. En cualquier momento podemos presionar 'd' sobre un círculo verde para borrarlo o 'n' para comenzar de nuevo. Para terminar presionamos 'q' sobre la imagen si **no queremos** guardar la solucion obtenida o 'w' **si queremos** guardarla en el header de la imagen.

Antes de salir podemos hacer astrometría sobre los objetos de interés que estan en la imagen pero no en el catálogo. Con la tecla 'o' el script centra un círculo cyan sobre el objeto y calcula sus coordenadas. Si el objeto es muy débil o si sólo se quiere saber las coordenadas en algun punto, con la tecla 'p' se dan las coordenadas del punto donde se presionó la tecla.

El archivo *solucion.db* contiene la transformación final para pasar de coordenadas en la imagen a coordenadas en el cielo. Para verificar el resultado podemos marcar los objetos en la imagen del mismo modo que lo hicimos al iniciar el proceso:

```
cl> display xx.fit 1
cl> fields xx.dat 3,4 | cctran STDIN STDOUT solucion.db xx.fit forward- | tvmark 1 STDIN mark=circle radii=2 color=204
```

8) Llegado a este punto tenemos una solución astrométrica para la imagen. Si por otro lado obtenemos coordenadas (x,y) en un archivo *xx.coo.1* para objetos de interés utilizando otras tareas (por ejemplo, **daophot.daofind** o **imcoords.starfind**) podemos encontrar sus coordenadas en el cielo mediante:

```
cl> fields xx.coo.1 1,2 | cctran STDIN STDOUT solucion.db xx.fit forward+
```

o sacar la informacion a un archivo *xx.sol*:

cl> fields xx.coo.1 1,2 | cctran STDIN xx.sol solucion.db xx.fit forward+

9) En el caso que se quiera borrar de la cabecera de una imagen la solución astrométrica encontrada se puede utilizar la tarea **imcoords.wcsreset** :

cl> wcsreset <imagen>.fts world

Y si se quiere copiar la solucion astrométrica de una imagen a otra se puede utilizar la tarea **imcoords.wcscopy**:

cl> wcscopy <imagen-sin-wcs>.fts <imagen-con-wcs>.fts