

Simultaneous astrometry

Pierre Astier

LPNHE / IN2P3 / CNRS , Universités Paris 6&7.

With : Dominique Boutigny, Johann, Philippe, Christian

LSST @ Grenoble
08/06/2016

Simultaneous astrometry: what ?

- It is about a set of images covering the same piece of sky
 - Simultaneously solving for the pixels → sky mappings (WCS's)
 - Simultaneously solving for the image to image mappings
- Required, prior to:
 - stacking
 - simultaneous measurements (fluxes, shapes)
 - image subtraction

Simultaneous astrometry: why bother ?

- We could just establish the pixel → sky mappings using reference catalogs, independently for each image
- But deep images like LSST's, contain (much) more objects than reference catalogs.
- One then can rely on all common objects to establish mappings.

How does it work ?

- We assume that our input images are equipped catalogue and a rough WCS ($\sim 0.5''$)
- Stage 1 : associate the catalogs
- Stage 2 : associate to some external catalog (to set the sidereal frame).
- Stage 3 : fit :
 - Coordinate transforms for image coordinates to some common frame. (\rightarrow WCS's)
 - Positions of all common objects.

Previous implementations of this functionnality

- SCAMP:
 - Part of astromatix (sextractor, swarp, ...)
 - THE reference code.
 - Does not fit positions of common objects, but minimizes differences between transformed positions of the same object.
- Some implementation for HSC in the stack framework.
-

Technicalities

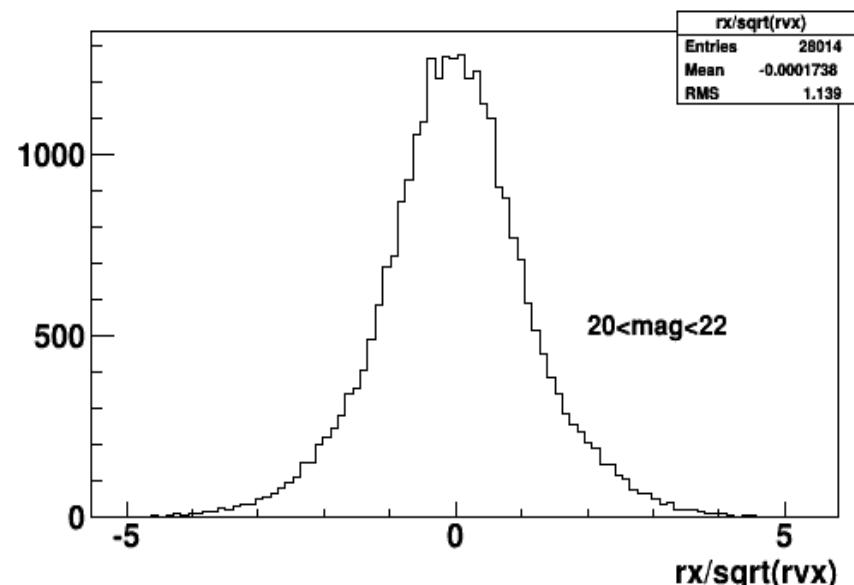
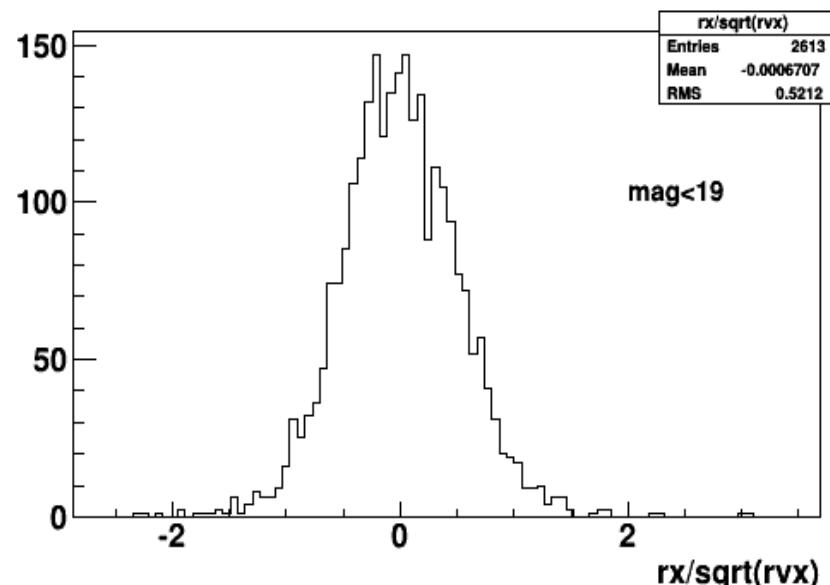
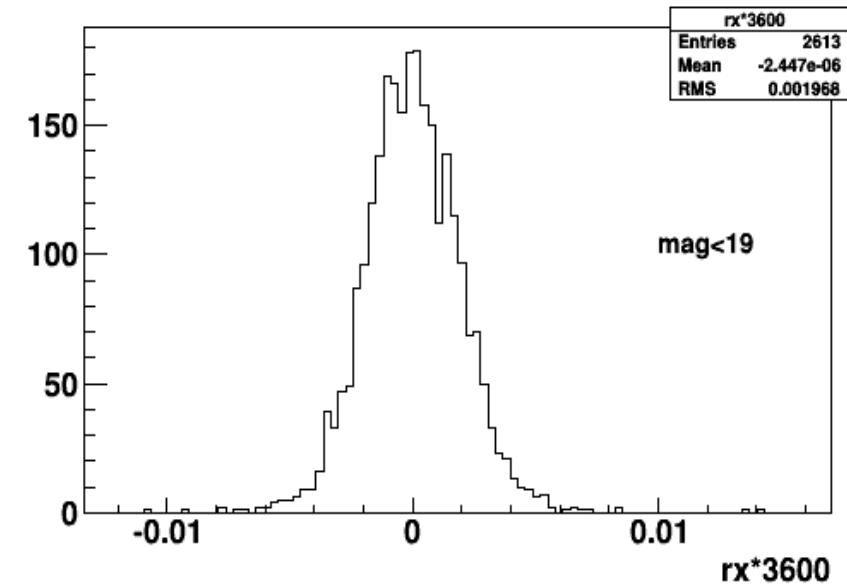
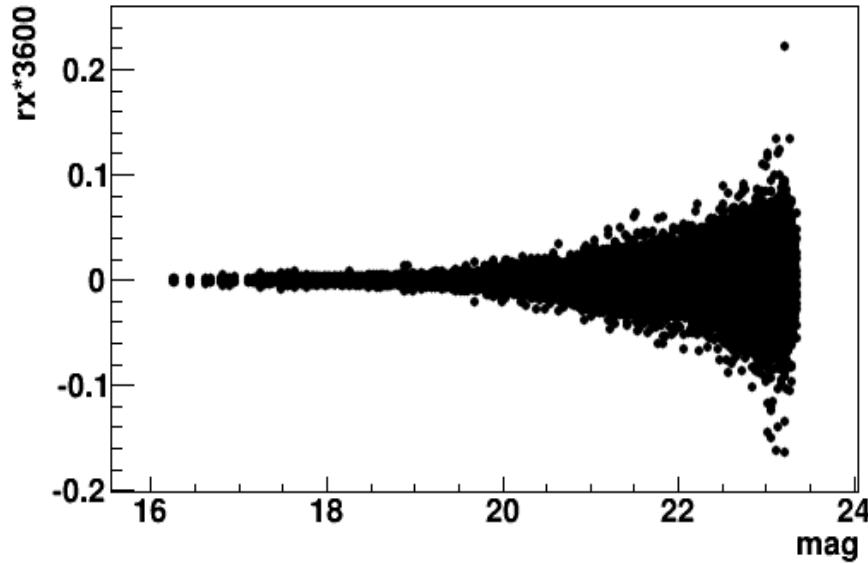
- There are a lot of fit parameters. 50,000 is common, Philippe went up to 500,000.
- A few key aspects :
 - Analytic computation of derivatives w.r.t param.
 - Sparse linear algebra (cholmod through Eigen)
 - Rely on factorization update mechanism for outlier removal (a cholmod feature).
- Never managed to spend more time fitting than loading the data.

Package history

- Proposed to the DM team in April 2015
 - Required for any shear measurement pipeline
- First working model in June 2015 (developed within the SNLS framework).
- Insertion in the stack, adjustments, further improvements, documentation.
- “Adoption” by the stack in early 2016:
 - lsst_france/meas_simastrom → lsst/jointcal

Trial (CFHT)

Internal rms residuals
are at the 2 mas level
For bright sources



Current status

- It works, provided that the input images have decent WCS's (typically at the 1" level). Is is fast.
- Over the whole CFHTLS, we get 500-700 visits per band, i.e. ~ 20000 “calexps” (data from 1 chip).
- Failures in the upstream reduction (processCCD) are expected, on this kind of scale.
- They should be addressed:
 - Devise some quality checking of the processCCD outcome (Philippe)
 - Robustify the weakest part: combinatorial astrometric matching (Pierre)

Combinatorial astrometric matching

- There are plenty of algorithms. This is a tricky business.
- One algorithm is empirically known to be very efficient: astrometry.net (web service !)
- It was adopted by the stack, and fails on images which are matched by the online version (?)
- The WCS fit is also shaky in the DM stack code
- No provision for matching whole exposures in the stack yet (which is the SCAMP way).

One jointcal development: “matchexposure” (DM-5828)

- Matches a whole exposure using some instrument geometrical model to position the catalogs from the different chips.
- Algorithm developed for SNLS standards.
- Inserted, tested in a jointcal branch.
- Pull request hanging for a few weeks now.
- Robert Lupton is upset about that.
- The communication channels are obviously inadequate.

What comes next ?

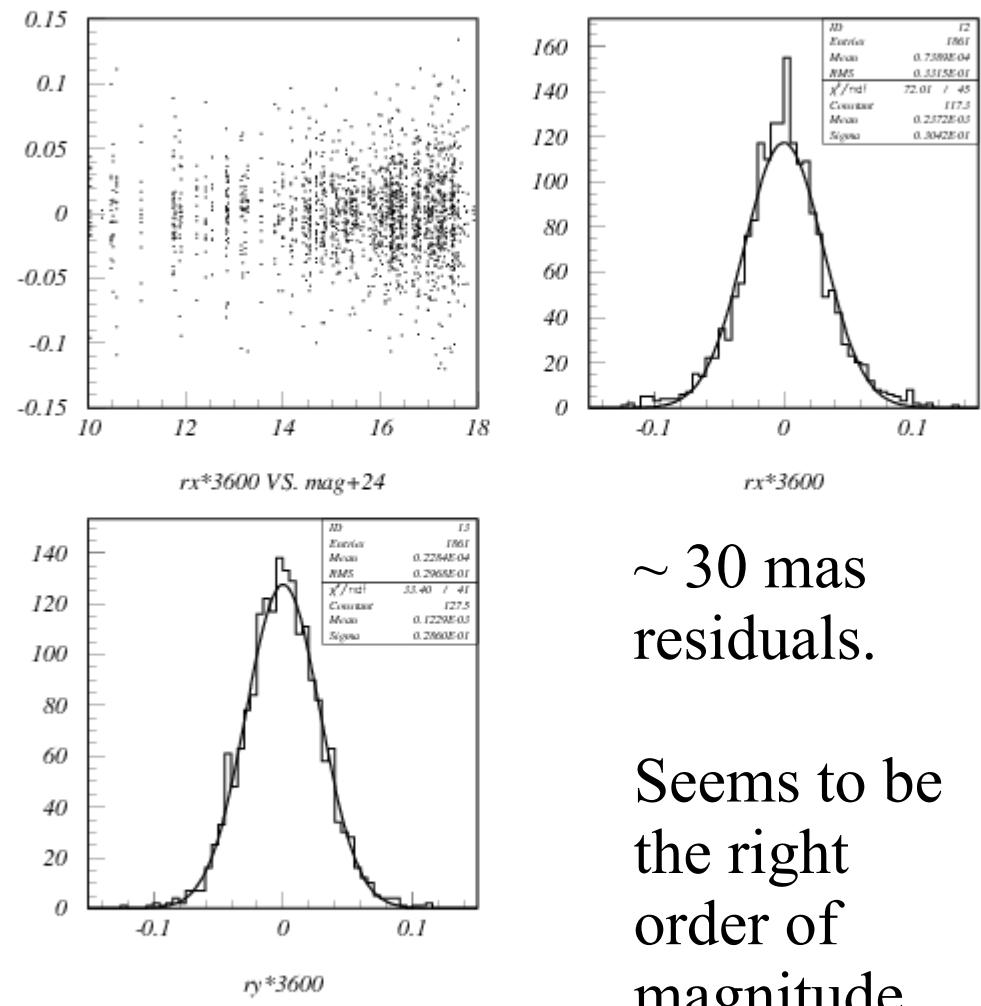
- I hope that the development is going to resume, because there is still some work needed.
- It seems that chat-like communication has some limitations, especially with 6 → 9 time zones between chatters.
- We need a plan for the next developments. In between, we just work on “detached” issues:
 - Quality assessment of ProcessCCD (Philippe)
 - Quality assessment of jointcal (Johan)
- Chat in person with RL in Belgrade (in 2 weeks).

Monocam run

- Use of an LSST sensor for astronomical observations, on modest (1.3m, 2m) telescopes.
<https://confluence.slac.stanford.edu/display/LSSTDESC/Observed+objects>
- Happened 2-3 weeks ago. Separate data streams from the CCD and the TCS....
- It worked! Nice images.
- Pre-reduction using DM done by Merlin Fisher-Levine. It hung at the astrometric match level (i.e. before jointcal).
- Output of this reduction at the CC.

Astrometric residuals on shallow monocam exposures

- 5-s exposures on a 1.3 m. Lim. Mag : 19-20.
- Flatfielding from the stack
- SNLS pre-reductions
- Astrometric match
- Simultaneous astrometric fit



~ 30 mas residuals.

Seems to be the right order of magnitude.

More slides

Implémentation des associations

Fit : moindres carrés

$$\chi^2_{meas} = \sum_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]^T W_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]$$

Termes de mesure

c,d : calexp, detection

\mathbf{M}_c : transfo (pixel → TP), une par calexp (p.e.)

$X_{c,d}$: position mesurée des objets (pixels)

P_c : projection (ciel → TP)

$W_{c,d}$: poids de $X_{c,d}$ (1/var), transformés par M_c .

\mathbf{F}_i : (sky) position estimée de l'objet (mesurée comme $X_{c,d}$)

$$\chi^2_{ref} = \sum_j [P(\mathbf{F}_j) - P(R_j)]^T W_j [P(\mathbf{F}_j) - P(R_j)]$$

Termes de référence

P : une projection fournie par l'utilisateur

\mathbf{F}_j : position sur le ciel (ajustée) de l'objet

R_j : position sur le ciel de l'objet dans le catalogue de référence

W_j : poids de R_j (1/var), transformé par P .

Moindres carrés (2)

$$\chi^2_{meas} = \sum_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]^T W_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]$$

Termes de mesure

...

$\mathbf{W}_{c,d}$: Erreur de mesure transformée through \mathbf{M}_c .

.... → donc \mathbf{W} dépend des paramètres !!

Oui mais en pratique , l'échelle de \mathbf{M} est bien fixée et donc cette dépendance peut-être ignorée

$$\chi^2_{ref} = \sum_j [P(\mathbf{F}_j) - P(R_j)]^T W_j [P(\mathbf{F}_j) - P(R_j)]$$

Termes de référence

P : une projection fournie par l'utilisateur (?!)

Pourquoi $P(F_j) - P(R_j)$ plutôt que simplement $(F_j - R_j)$?

Parce que la distance sur la sphère n'est pas euclidienne

Moindre carrés (3)

$$\chi^2_{meas} = \sum_{c,d} [\textcolor{red}{M}_c(X_{c,d}) - P_c(\textcolor{red}{F}_i)]^T W_{c,d} [\textcolor{red}{M}_c(X_{c,d}) - P_c(\textcolor{red}{F}_i)]$$

- Cette approche s'applique aussi à l'ajustement de transformations entre images :
 - On fixe les P_c à l'identité.
 - Ainsi que l'un des M_c .
 - et on n'a pas de catalogue externe ou de termes de référence.
- On obtient ainsi les transformations optimales entre images, étant données les mesures de positions et leurs incertitudes.

Moindres carrés (4)

$$\chi^2_{meas} = \sum_{c,d} [\textcolor{red}{M}_{\textcolor{red}{c}}(X_{c,d}) - P_c(\textcolor{red}{F}_i)]^T W_{c,d} [\textcolor{red}{M}_{\textcolor{red}{c}}(X_{c,d}) - P_c(\textcolor{red}{F}_i)]$$

Pour minimiser, on développe le χ^2 au second ordre dans les paramètres,

$$\frac{d^2\chi^2}{d\theta^2} \equiv H = J J^T, \quad \frac{d\chi}{d\theta} \equiv g$$

J est une matrice rectangulaire, $N_{\text{par}} \times N_{\text{carrés}}$
et on résout pour la meilleure direction de descente :

$$H\delta\theta = -g$$

Différence avec SCAMP

- Scamp n'ajuste pas les positions des objets. Il minimise l'ensemble des distances 2 à 2. C'est statistiquement sous-optimal.
- En revanche, Scamp est bien debuggé, produit des tas de plots de contrôle, et a une bonne centaine de leviers de contrôle.

Fit astrométrique

Les SimplePolyXXX réalisent à la fois un ajustement simultané des WCS (sans utiliser que des images peuvent provenir du même instrument) et un ajustement des transfos entre images (sans référence à un catalogue externe)

Les ConstrainedXXX implémentent un modèle dans lequel les positions relatives des CCDs sont fixes

Implémentation

- Moindres carrés avec des dérivées (principalement) analytiques (p.r. aux positions et paramètres).
- Algèbre linéaire creuse (Eigen3 et Cholmod)
- Environ 2000 lignes de code C++ nouveau (~ 10 classes) qui implémente le fit et les modèles. Le reste est du recyclage de code existant.
- Nombreuses routines liées au python

Eviction des outliers

- Il faut détecter et éliminer les intrus. A la fois les mesures et les références.
- C'est une procédure itérative : une mauvaise mesure dégrade tous les résidus de l'objet auquel elle est associée.
- Plutôt que de répéter la résolution de systèmes linéaires très proches, on peut plutôt utiliser la mise à jour de la factorisation de Cholesky proposée dans Cholmod.
- L'outillage nécessaire est dans le paquet micro_cholmod (fabriqué par Christian)
24

Qualité des résultats

- On descend à 2 ou 3 mas d'erreur systématique (i.e. non expliquée par le bruit de grenaille) sur Megacam, Suprime-Cam, Hyper-suprime-cam.
- Scamp est plutôt vers ~ 10 mas.

Détails techniques

- github.com/lsst-france/meas_simastrom
- La doc est dans doc
- Cholmod :
- github.com/lsst-france/micro_cholmod
- Développements en cours :
 - Encore un peu de debugging.
 - Modèles plus complexes
 - Plots de contrôle,
 - fichier TODO dans le source.

Des questions ?

Le « pré-conditionnement » avant factorisation

Cholesky :

$$A = LL^T$$

Chol(A)

$$(ou LDL^T)$$

L triangulaire

A

P : permutation

$$P^TAP$$

Chol(P^TAP)

A : pos-def.

Temps d'exécution

- Aujourd'hui plus de la moitié du budget est consommée dans la lecture des catalogues. On espère que ça va changer.
- Pour ~ 500 calexps (10^5 paramètres):
 - Calcul des dérivées : $<\sim 1$ s
 - JJ^T : $>\sim 1$ s
 - Factorisation solution update : ~ 6 s
 - Mise à jour de la factorisation : ~ 0.5 s

WCS : coordonnées sidérales

Calabretta, M.R., & Greisen, E.W., (2002), A & A, 395, 1077-1122

En pratique :

CRPIX_i

CD_{i,j}

ou CDELT_i

CTYPE_i

CRVAL_i



Etablir le WCS d'une image

- Mécanique du télescope : en aveugle, au mieux 5" en général moins bien.
- On peut améliorer en associant le catalogue de l'image à un catalogue de référence, le plus souvent moins profond que notre image.
- Donc :
 - Associer le catalogue de l'image et le catalogue de référence, par combinatoire.
 - Ajuster les paramètres (certains...) du WCS
 - Encoder dans le header FITS.

Ajuster le WCS

$$\begin{pmatrix} x_{TP} \\ y_{TP} \end{pmatrix} = \begin{pmatrix} CD1_1 & CD1_2 \\ CD2_1 & CD2_2 \end{pmatrix} \begin{pmatrix} x_p - CRPIX1 \\ y_p - CRPIX2 \end{pmatrix} \leftarrow$$

Pour de petits angles, la déprojection devient :

$$\begin{pmatrix} \alpha \\ \delta \end{pmatrix} = \begin{pmatrix} 1/\cos\delta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_{TP} \\ y_{TP} \end{pmatrix} + \begin{pmatrix} CRVAL1 \\ CRVAL2 \end{pmatrix}$$

Et donc, on ne peut pas ajuster

- les CD (4)
- les CRPIX (2)
- les CRVAL (2)

En pratique, le plus simple est de fixer le plan de projection (CRVAL{1,2})

Distorsions

- Le modèle impose une relation linéaire entre coordonnées sur le senseur et plan de projection.
- Ce n'est vrai que si l'image d'une droite sur le senseur est une droite sur ce plan.
- En général les optiques produisent des distorsions importantes.
- Il faut donc un moyen de les encoder.

Distorsions : plusieurs % entre centre et bord

Variation relative du jacobien
i.e. $(\text{Plate-scale})^2$ sur Megacam
(Regnault+ 2009)

Plate-scale sur Suprime-cam
(von der Linden+ 2014)

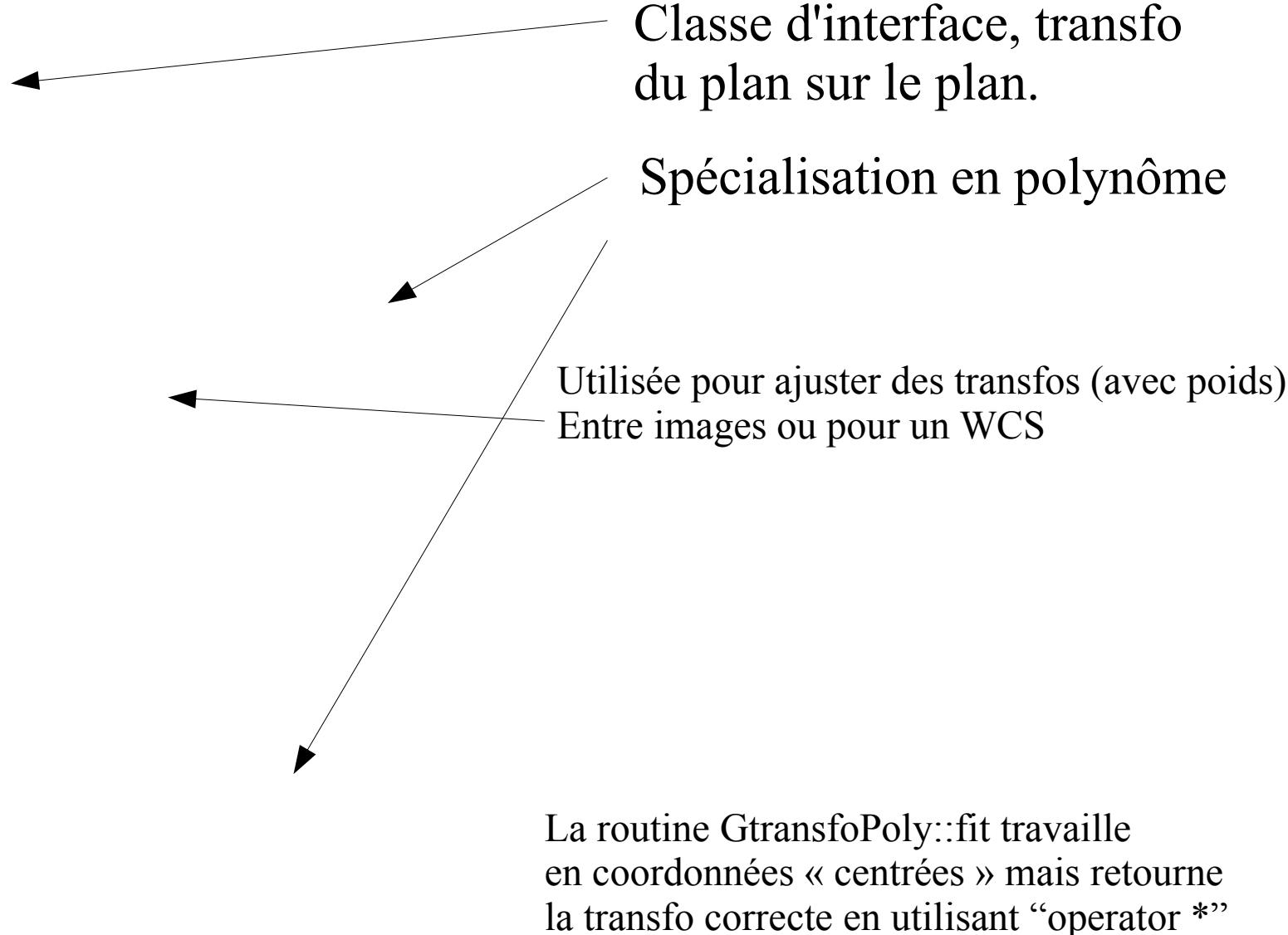
Distorsions

- Pour l'encodage des distorsions, il n'y a pas vraiment de norme universelle. Deux schémas sont « courants », il utilisent tous les deux un polynôme pour décrire les distorsions.
 - SIP : $\text{Pix2TP} = \text{Lin}([\text{CD}, \text{CRPIX}], \text{Pol}(\text{X}_{\text{ccd}}))$
 - PV : $\text{Pix2TP} = \text{Pol}(\text{Lin}([\text{CD}, \text{CRPIX}], \text{X}_{\text{ccd}}))$
- Dans les deux cas, le résultat est un polynôme de même degré que celui utilisé pour les distorsions
- La séparation est arbitraire : on choisit d'avoir le meilleur WCS si les distorsions sont ignorées

Distortions

- Implémentations :
 - SIP : lsst-stack, ds9, ~~weslib~~
 - PV : scamp, swarp, Poloka, ~~weslib~~
- En pratique l'un ou l'autre est un détail d'implémentation. Le SIP est facile à implémenter « par dessus » la wcslib, comme c'est fait dans le stack.
- Un détail pratique : la wcslib n'est pas très rapide. Swarp la contourne pour le ré échantillonnage.

Fit du WCS : sketch simplifié de l'implémentation dans Poloka



Associer l'ensemble d'une exposition

- On prend un ensemble d'expositions (complètes) et on évalue la moyenne des transformations pixel-> plan tangent.
- Pour une nouvelle exposition, on transporte l'ensemble des catalogues vers le plan tangent avec ces transformations moyennes.
- On associe (combinatoire) avec le catalogue de référence.
- On récupère les associations dans chaque CCD et on ajuste un WCS par CCD.
- Bénéfices :
 - Une seule recherche combinatoire/exposition
 - Les poses courtes peuvent être équipées d'un WCS
- Fonctionne sur Megacam et SuprimeCam.

Plusieurs expositions

- Quand on a plusieurs images du même champ, on peut ajuster les transformations pour minimiser la dispersion entre images.
- On tire parti :
 - D'une résolution de position en général meilleure que les catalogues de référence. Ça va changer avec Gaia.
 - Du fait que bien des objets dans les images ne sont pas dans le catalogue de référence. Ça ne va guère changer avec Gaia.

Associating detections of the same object

The simple way :

- For each image
 - Load catalog and apply quality cuts
 - Match it to the “Object catalog” (in the tangent plane)
 - Add the unmatched objects to the “Object catalog”
- I know it is fast and efficient (with a 2D $O(N_1 \log(N_2))$ matching)
- One could be worried by the outcome depending in principle of the order of input images.
- In practice, this is not serious if the WCS's are accurate enough (1-2 pixels) and blends are ignored (as they should).
- Faster than reading the catalogs.

Fit : Least Squares

$$\chi^2_{meas} = \sum_{c,d} [\mathbf{M}_c(\mathbf{X}_{c,d}) - \mathbf{P}_c(\mathbf{F}_i)]^T \mathbf{W}_{c,d} [\mathbf{M}_c(\mathbf{X}_{c,d}) - \mathbf{P}_c(\mathbf{F}_i)]$$

Measurement terms

c,d : calexp, detection

\mathbf{M}_c : mapping (pixel → TP), one per calexp

$\mathbf{X}_{c,d}$: measured position of the object (pixels)

\mathbf{P}_c : projection (sky → TP)

$\mathbf{W}_{c,d}$: Measurement weight (1/var), transformed through \mathbf{M}_c .

\mathbf{F}_i : (sky) position of the object (measured as $\mathbf{X}_{c,d}$)

$$\chi^2_{ref} = \sum_j [\mathbf{P}(\mathbf{F}_j) - \mathbf{P}(\mathbf{R}_j)]^T \mathbf{W}_j [\mathbf{P}(\mathbf{F}_j) - \mathbf{P}(\mathbf{R}_j)]$$

Reference terms

\mathbf{P} : some (user-provided) projection

\mathbf{F}_j : (fitted) sky position of the object

\mathbf{R}_j : sky position of the object fitted (reference catalog)

\mathbf{W}_j : \mathbf{R}_j weight (1/var), transformed through \mathbf{P}

Least Squares (2)

$$\chi^2_{meas} = \sum_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]^T W_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]$$

Measurement terms

...

$\mathbf{W}_{c,d}$: Measurement error, transformed through \mathbf{M}_c .

.... → so \mathbf{W} depends on some fitted parameters !

Yes, but in practice, the scale of \mathbf{M} is extremely well known, so this can be ignored.

$$\chi^2_{ref} = \sum_j [P(\mathbf{F}_j) - P(R_j)]^T W_j [P(\mathbf{F}_j) - P(R_j)]$$

Reference terms

P : some (user-provided) projection

Why $P(F_j)$ - $P(R_j)$ rather than just (F_j-R_j) ?

because the distance on the sphere is **not** Euclidean

Least Squares (3)

$$\chi^2_{meas} = \sum_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]^T W_{c,d} [\mathbf{M}_c(X_{c,d}) - P_c(\mathbf{F}_i)]$$

- This setup accommodates the fit of mappings between images:
 - All the P_c are set to identity.
 - One of the M_c is set to identity.
 - No external catalog nor “reference terms”.
- This yields the optimal mapping between images, given position measurements and their uncertainties.

Différence avec SCAMP

- Scamp n'ajuste pas les positions des objets. Il minimise l'ensemble des distances 2 à 2. C'est sous-optimal.
- En revanche, Scamp est bien debuggé, produit des tas de plots de contrôle, et a une bonne centaine de leviers de contrôle.

Least Squares (4)

$$\chi^2_{meas} = \sum_{c,d} [\textcolor{red}{M}_c(X_{c,d}) - P_c(\textcolor{red}{F}_i)]^T W_{c,d} [\textcolor{red}{M}_c(X_{c,d}) - P_c(\textcolor{red}{F}_i)]$$

Pour minimiser, on développe le chi² au second ordre dans les paramètres,

$$\frac{d^2\chi^2}{d\theta^2} \equiv H = J J^T, \quad \frac{d\chi}{d\theta} \equiv g$$

J est une matrice rectangulaire, N_{par} x N_{carrés}
et on résout pour la meilleure direction de descente :

$$H\delta\theta = -g$$

Implementation

- Least squares with (mostly analytical) computation of derivatives (w.r.t positions and parameters).
- Sparse matrix algebra (Eigen3 package). Similar performance with Cholmod.
- About 1500 lines of new C++ code (~ 10 classes) to implement the fit and the model. Used existing (home-made) classes for everything else.
- The fit talks to the model via two abstract classes.

Fit astrométrique

Les SimplePolyXXX réalisent à la fois un ajustement simultané des WCS (sans utiliser que des images peuvent provenir du même instrument) et un ajustement des transfos entre images (sans référence à un catalogue externe)

Les ConstrainedXXX implémentent un modèle dans lequel les positions relatives des CCDs sont fixes

Le « pré-conditionnement » avant factorisation

Cholesky :

$$A = LL^T$$

Chol(A)

$$(ou LDL^T)$$

L triangulaire

A

P : permutation

$$P^TAP$$

Chol(P^TAP)

A : pos-def.

Outlier removal

- I have not found a **canned** small-rank update of Cholesky factorizations. The only one I know about is Cholmod providing a rank-1 update.
- So, I have used the following trick: do not remove in a single pass 2 outliers that constrain the same parameter.
- Would require a lot of iterations to come to zero outlier removed.
- I was not that patient: I ran it only 4 iterations.

Trial run

- 15 Megacam r-band 300-s exposures on D3, observed over the same lunation. 540 Calexp's.
- Use USNO-A2 for the reference catalog.
- Ignore proper motions.
- Use Gaussian-weighted positions and associated errors.
- Strict selection of measurements (no flag at all, S/N>10), average of ~400 measurements/calexp
- All calexps have their own mapping parameters as if they all came from different instruments

Trial run

- ~32,500 objects, ~210,000 measurements.

Trial run : residuals

Residuals of the “measurement terms”, these are internal residuals

Residuals vs mag

Use simplistic error model $V = V_{\text{meas}} + (0.02 \text{ pix})^2$

Beware : residuals
are **not**
Studentized, and
The number of
measurement is
not that large

A smaller constant
term would help for
the bright side, but
not for the faint side

Overall : it could
be much worse!

Second trial run: Supreme cam

120-s (i and z)-band exposures reduced by Augustin Guyonnet.
12 exposures, guessed photometric scale. Exactly the same code.

Computer resources

- Execution time: for the 15 Megacam exposures:
 - Reading the (540) catalogs : ~ 100 seconds at 33% CPU
 - Associating : negligible
 - Fitting: $<\sim 20$ s per iteration
 - Computing the derivatives : ~ 1 s
 - “Squaring” the Jacobian, i.e. $H = JJ^T$: ~ 3 s
 - Factorize-solve-update (dim=75,510, nnz=17,164,700) ~ 13 s.
 - Partial fits (positions OR mappings) are solved instantly.
 - Total : 125.137u 19.769s 3:40.44 65.7% (Xeon 2.3 GHz)
- Memory reaches ~ 1 GByte (not completely sure though)

To be done (at least):

- I still have to test the code of the mapping model for a “rigid instrument”. I would not be surprised if the residuals come out much larger. Have to think about relaxing rigidity.
- Study dispersions of faint objects (galaxies mostly).
- Proper motions, parallaxes ? the code to handle proper motions is there, but I have not implemented anything to detect “moving” stars.
- Output of mappings (i.e. WCS's)... Which format? Output of the catalog.

To be done (suite)

- Identifier les possibilités de parallélisme de la factorisation.
- Liaison avec le stack :
 - Header FITS (standardisation).
 - Catalogues.
- Lecture du code.

Code

- <http://gitlab.in2p3.fr/astier/gastro>
- A titre documentaire : ne peut tourner sans poloka-core
- La branche dans laquelle je travaille s'appelle « sparseshell »
- Doc sur <http://supernovae.in2p3.fr/~astier/gastro/>
- Les dépendances sont installées au CC
- L'insertion dans le stack va nécessiter du découpage

(Simplified) class diagram

Input data

Fit stuff