ECOLE POLYTECHNIQUE
PROMOTION 2009
DERAM Pierre

RAPPORT DE STAGE DE RECHERCHE

traitement d'images en comptage de photons

NON CONFIDENTIEL

Option : PHYSIQUE
Champ de l'option : Astrophysique et Cosmologie
Directeur de l'option : Mr Martin Lemoine
Directeur de stage : Mr Hervé Le Coroller
Dates du stage : du 10 Avril au 27 Juillet 2012

Nom et adresse de l'organisme :

Observatoire de Haute Provence /CNRS
F-04870 St Michel l'Observatoire
France

Résumé:/Abstract:

Pour égaliser les chemins optiques des faisceaux lumineux venant de télescopes différents, l'interférométrie moderne utilise des systèmes de 'lignes à retards'. Ces systèmes ont le désavantage de réduire la sensibilité des télescopes en multipliant le nombre de réflexions subies par la lumière avant son acquisition. De plus, les lignes à retard posent des difficultés techniques qui ralentissent le développement actuel de l'interférométrie vers des systèmes à pupilles plus riches (nombreuses sous ouvertures) et plus grandes (distances entre les sous-ouvertures). Depuis 2002, une équipe de chercheurs de l'observatoire de Haute Provence travaille à la conception d'un télescope d'un nouveau genre baptisé CARLINA. Constitué d'un miroir dilué sur le sol et de deux nacelles suspendues à un ballon d'hélium, cet interféromètre fonctionne sans ligne à retard. La caractérisation du télescope nécessitera le traitement des images stellaires attendues pour l'hiver 2013.

En interférométrie, on travaille sur les spectres des images dont on cherche à extraire une mesure de la visibilité de l'objet observé. Je démontre que dans le cas d'une pupille à N sous-ouvertures, le module au carré de la visibilité complexe le long d'une base est donné par le rapport du pic secondaire du spectre sur le pic principal le tout multiplié par le coefficient N^2.

Les images de Carlina sont acquises par l'intermédiaire d'une caméra à comptage de photons qui permet de détecter des flux faibles et effectue des temps de pose suffisamment courts (1 milliseconde) pour figer les turbulences atmosphériques. Les images obtenues par une caméra de ce type possède un défaut appelé 'biais de photons' qui introduit un bruit de fond dans le spectre. Je propose une méthode de correction de ce bruit de fond et d'extraction du module des visibilités. J'ai créé à cet effet un logiciel de traitement complet des images issues de la caméra à comptage de photons. Initialement prévue pour le traitement des images de Carlina, la méthode utilisée et la structure du logiciel sont généralisables à n'importe quelle image issue d'un interféromètre utilisant une caméra à comptage de photons.

To equalize optical path of different light beams, modern interferometry use 'delay line' systems. This method limits telescope sensibility by multiplying the number of light reflexion. Moreover, interferometry development is nowaday confronted to technical difficulties which impede the use of denser (number of sub-apertures) and longer (distance between sub-apertures) pupils. Since 2002, a research team from the observatoire de Haute Provence is working on the conception of a new type of telescope named CARLINA. Using a diluted miror on the ground and two gondolas suspended to an helium balloon, this interferometer do not use delay lines. To characterize Carlina performances, we need to be able to analyze stellar fringes which are expected in winter 2013

In interferometry, we work on the fringes' spectrum from which we tried to extract a measure of the observed star visibility. I demonstrate in a case of a N sub-apertures pupil that the square modulus of a complex visibility along a base is given by the ratio between the secondary peak of the spectrum and the main peak multiplyed by a factor N^2.

Carlina frames are obtained by a photon-counting camera which can detect weak flux of light and uses 1 millisecond exposure time to 'immobilize' atmospheric turbulence. Picture obtained by this type of camera holds a defect named 'photon biais' which creates a background noise in the spectrum. I propose an extraction method of the visibility modulus from photon-counting camera pictures which corrects photon biais noise. I create an interactive software to extract visibility square modulus from fringes. Optimized for Carlina, this extraction method and the software structure can be adapted to all interferometers which use photon counting camera.

Table des matières:

Résumé/Abstract. I

Introduction. IV

Partie I: L'interféromètre CARLINA. 1
Le principe du LDT. 2
Présentation globale de Carlina. 3
Comment stabiliser l'architecture? 5
Comment co-sphériser les miroirs au sol? 8
Comment acquérir les images? 10
Comment pointer et suivre une étoile? 10
Bilan des nuits de CARLINA 12

Partie II: Les principes fondamentaux du traitement d'image en interférométrie 13
Relations fondamentales 14
Du réseau d'ouvertures à l'équation des franges 14
De l'équation des franges à l'extraction du module de la visibilité 17
Première approche à une reconstruction de l'objet 19

Partie III: Extraction du module carré de la visibilité complexe d'une source 23
La caméra à comptage de photons 24
Le biais de photons 25
Le trou de photons 25
Correction du trou de photons 26
Traitement complet des images 29

Bibliographie 34

Annexes 35
Comment combiner la lumière en interférométrie? Le cas du LDT 35
Code du programme de création des spectres. 38
Capture d'écran du logiciel d'extraction des visibilités. 40
Code du programme d'extraction des visibilités. 41

"regardez l'univers, les étoiles qui brillent, milliards plein le firmament bluffeuses... qu'elles sont mortes des milliards d'années!... évaporées!... allez pas rire des astronomes dardés vers le ciel, à supputer, mathématiser des vides... il sont comme vous, ils gagnent leur vie..."

Louis Ferdinand Céline
Rigodon

Introduction:

Créé en 1937, l'observatoire de Haute-Provence est le plus grand site Français d'observation de l'environnement et de l'univers. Il est réputé pour avoir permis la découverte en 1995 de la planète 51 Pegasi b, la première planète extrasolaire jamais découverte par l'homme. Il abrite aujourd'hui le fameux spectrographe SOPHIE (ainsi que trois autres télescopes de haut niveau) derrière lequel des équipes d'observations se relayent près de 363 jours par ans pour traquer les exoplanètes. En parallèle, l'observatoire abrite également une station de sondage de l'atmosphère par la méthode dite LIDAR (télédétection par laser). Plus récemment, l'OHP s'est lancé dans la mise en place d'un véritable observatoire de la biodiversité associé au chêne blanc, l'O3HP, qui étudie et simule les conséquences du changement climatique sur la forêt couvrant les 95 hectares du site. Enfin, l'OHP développe de nouveaux concepts instrumentaux et accueille le projet CARLINA.

Développé depuis 2002 par Hervé Le Coroller et Julien Dejonghe, le prototype CARLINA entend démontrer à la communauté scientifique la faisabilité d'une nouvelle génération d'interféromètres géants baptisés LDT pour 'large diluted telescope'. Fonctionnant sans lignes à retard, ce nouveau concept de télescope pourrait permettre d'améliorer la sensibilité des interféromètres (considérablement limitée actuellement par les nombreuses réflexions au sein des lignes à retard) et leur pouvoir de résolution (augmentation de la distance entre les sous ouvertures de la pupille).

Dans un contexte international dominé par de grands projets instrumentaux (le futur télescope supergéant E-ELT européen par exemple), j'ai été attiré par l'audace et l'originalité du concept de Carlina qui ouvre des voies de recherche nouvelles. Hervé Le Coroller a accepté de m'accueillir au sein de son équipe du 10 avril au 27 Juillet 2012 avec pour mission la réalisation d'un logiciel de traitement des futurs images données par le télescope Carlina. Ce fut pour moi un stage très enrichissant et une première expérience scientifique autour d'un projet passionnant.

J'ai participé à 7 nuits complètes de test sur le démonstrateur Carlina au cours desquels j'ai pu apprécier l'ampleur du travail effectué et comprendre le fonctionnement du télescope. Dans la première partie de ce rapport, je présente le fonctionnement du démonstrateur Carlina et les solutions instrumentales qui ont été développées pour répondre aux enjeux scientifiques et techniques.

Dans une deuxième partie, je présente les principes du traitement des images en interférométrie et je démontre le théorème de Van Citter Zernike dans le cas d'une pupille quelconque. Des simulations programmées sous IDL illustrent les résultats. En annexe, je présente le calcul des différences de marche pour une pupille co-sphérisée.

La troisième partie présente la méthode d'extraction des visibilités à partir de franges obtenues en laboratoire. Le logiciel que j'ai développé corrige le bruit de fond du spectre et permet d'extraire le module carré des visibilités d'une source à travers n'importe quelle pupille à 3 sous-ouvertures. Son code complet est recopié en annexe.

Partie I:

Le télescope du futur?

Présentation du premier prototype de télescope dilué construit au monde: l'interféromètre CARLINA

L'astronomie moderne requiert des outils de plus en plus puissants pour explorer des objets lointains et peu lumineux. De grands projets internationaux sont en cours un peu partout dans le monde pour construire des télescopes de plus en plus grands. Actuellement, le centre européen de l'ESO au Chili détient l'un des plus grands interféromètres au monde appelé VLTI qui combine la lumière venant de 3 à 4 télescopes parfois espacés de près de 200 mètres. Du côté des miroirs monolithiques, 14 télescopes géants sont actuellement recensés à travers le monde dont les plus grands sont les deux 'Keck telescopes' américains (situés à Hawai) d'un diamètre de 10 mètres. Plusieurs projets sont en cours d'élaboration à travers le monde qui vise principalement à augmenter la taille des appareils. On ne peut oublier de citer le futur E-ELT (European Extremely Large Telescope) et son miroir de 39 mètres qui constitue le projet phare de l'astronomie actuelle. Coût estimé: un milliard d'euros. Cette course effrénée aux instruments de plus en plus grands se heurte néanmoins à de graves difficultés techniques. La réalisation d'un interféromètre géants à plusieurs dizaines d'ouvertures basée sur l'architecture actuelle des interféromètres semble encore hors de portée compte tenu des difficultés rencontrées pour piloter un grand nombre de lignes à retards. Pour surmonter ces difficultés, certains n'hésitent pas à repenser entièrement l'architecture des interféromètres et explorent des voies instrumentales nouvelles...

Remarque: la présentation qui suit ne constitue pas une documentation technique sur Carlina, mais bien un exposé de physique instrumentale présentant les principes optiques et mécaniques du fonctionnement du télescope. Il comporte de nombreuses illustrations toutes nécessaires à la compréhension de l'exposé.

1) Le principe du LDT

Élaboré par Herve Le Coroller et Julien Dejonghe depuis 2002, le projet Carlina entend démontrer à la communauté scientifique la faisabilité d'une nouvelle génération d'interféromètre géant: les LDT pour Large diluted telescope. Les innovations de ce nouveau type d'interféromètre tiennent tout entières dans son architecture 'simple' inspirée du radio-telescope Arecibo:
1) On dispose au sol un réseau de miroirs épousant la forme d'une demi-sphère virtuelle. Ils forment le miroir principal (M1) du télescope dilué.
2) Les rayons lumineux venant de l'infini qui frappe M1 forment une image sur sa demi-sphère focale (Un correcteur de Mertz corrige l'aberration de sphéricité). Au foyer de ce correcteur, il y a égalité des chemins optiques entre les rayons issus des différents miroirs du réseau formant M1.
3) Un système d'acquisition (une caméra) placée sur la demi sphère focale recueille l'image des franges obtenues.

Le schéma ci dessous reprend le principe général du télescope dilué:

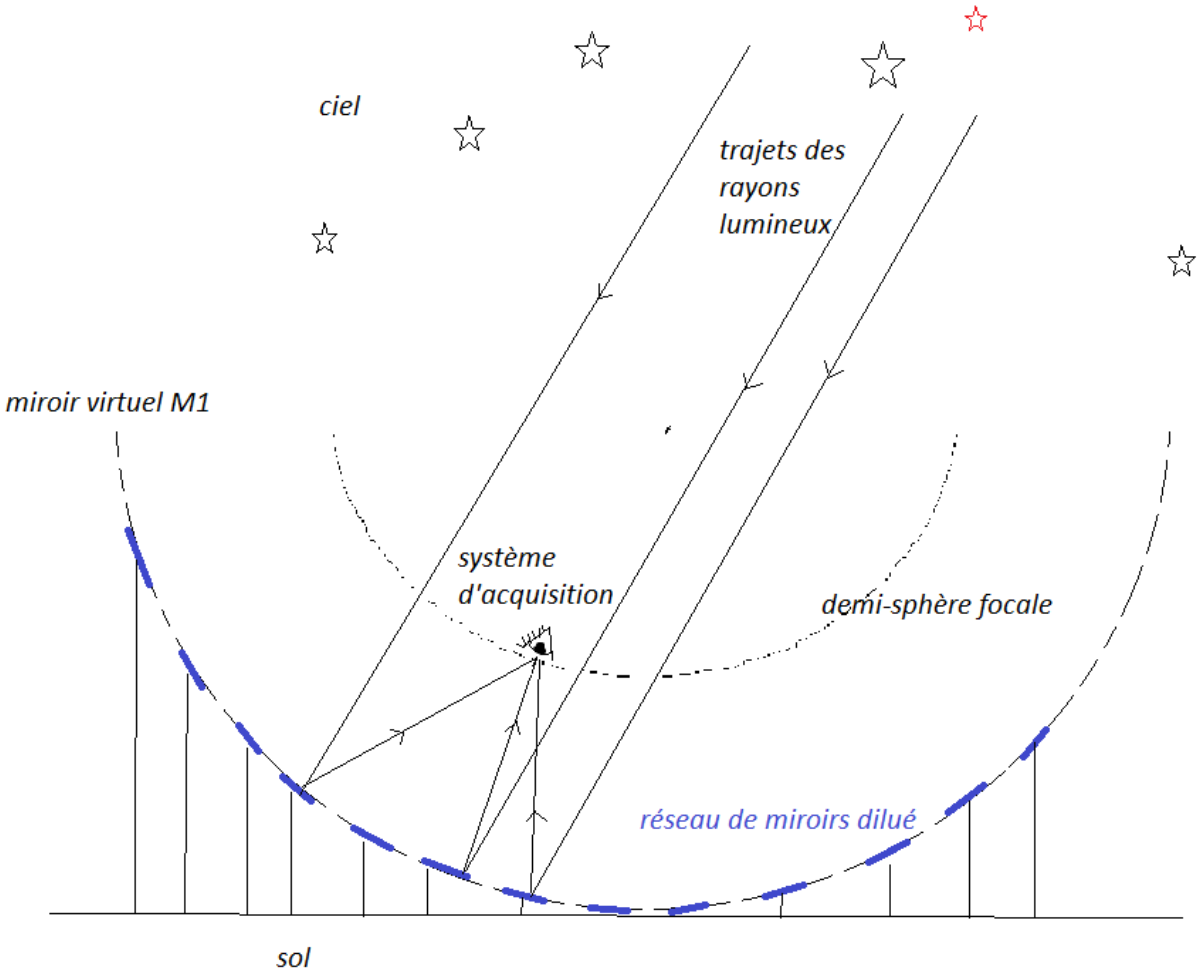


Figure 1

Les avantages d'une telles architecture par rapport à un interféromètre classique sont nombreux:

→ Le LDT ne possède aucun système de lignes à retard. Les franges se forment naturellement sur la demi-sphère focale du télescope.

→ Le suivi des étoiles et l'observation du ciel se fond sans modification de la position du miroir M1: C'est le système d'acquisition qui se déplace sur la demi-sphère focale.

→ Le nombre de réflexions de la lumière avant l'acquisition est minimisé (3 réflexions seulement: une sur le miroir M1 et deux dans un correcteur d'aberration de sphéricité) ce qui permet de conserver un flux de lumière maximum (possibilité d'observer des étoiles moins brillantes)

→ La taille (base) du miroir dilué M1 peut être gigantesque. Il faut seulement être capable de récupérer les images à une hauteur R/2! Un tel télescope pourrait donc largement égaler voire dépasser la taille des plus grands appareils actuels et atteindre des résolutions impossibles avec un miroir monolithique.

→ Le nombre de sous ouvertures (nombre de miroirs dans le réseau M1) peut être augmenté. Les interférences fonctionnant avec un nombre élevé de sous ouvertures sont donc possibles (ce qui n'est pas le cas avec les interféromètres actuels pour des raisons techniques: lignes à retard...).

→ Le coût d'un LDT ('large diluted telescope') est a priori inférieur à n'importe quel autre télescope équivalent (monolithique ou interféromètre).

La réalisation d'un télescope de ce type pose néanmoins des questions techniques nouvelles:

2) Comment stabiliser l'architecture au dessus du sol ?

3) Comment co-sphériser le réseau de miroir au sol?

4) Comment acquérir les images à plusieurs dizaines de mètres au dessus du sol?

5) Comment pointer et suivre une étoiles?

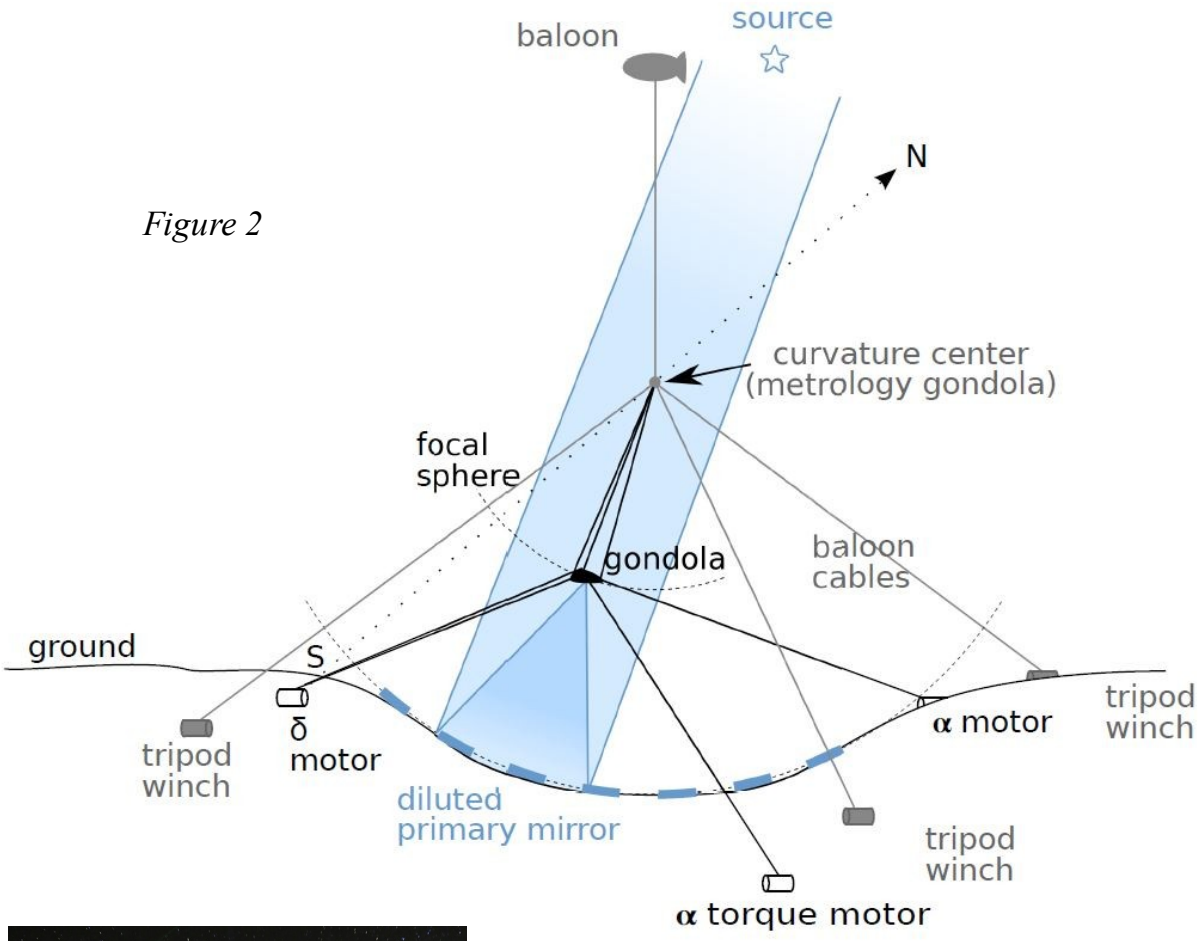
C'est à toutes ces questions et bien d'autres encore que le démonstrateur technique de l'observatoire de Haute Provence, Carlina, tentent d'apporter des réponses. Nous allons dans les pages qui suivent présenter les solutions retenues ainsi que les résultats expérimentaux. J'ai personnellement participé à chacune des nuits d'expérimentation durant la période Avril-Juillet (7 nuits au total) où l'on a pu tester en grandeur nature l'ensemble du système jusqu'au suivi des étoiles. La compréhension du fonctionnement de Carlina a donc constitué un point important de mon stage.

2) Présentation globale de Carlina:

Le prototype Carlina constitue un 'mini' LDT constitué de trois miroirs de 25 cm co-sphérisés sur un miroir virtuel de 71 mètres de rayon. Le système est constitué de deux nacelles aériennes portées par un ballon d'hélium: une nacelle dite 'de métrologie' situé au centre de courbure du miroir primaire virtuel, et une nacelle dite 'focale' se déplaçant de long de la demi-sphère focale à 35

mètres de hauteur chargée de récupérer les images du télescope. Tout le système est contrôlé et asservi par des cordes à hauts modules d'Young et des moteurs. L'ensemble est présenté dans les paragraphes qui suivent.
Ci dessous un schéma général du projet Carlina et quelques photos du prototype:

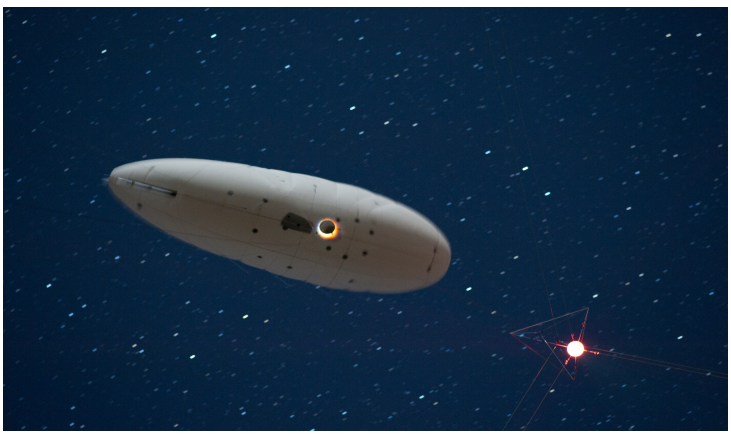
Figure 2



Ci dessus: Schéma général du projet Carlina.

Ci contre: Photo de Carlina de nuit. On aperçoit le ballon, les deux nacelles et différents lasers de réglages (photo réalisée en pose longue)

Ci dessous: Photo du ballon d'hélium vu du dessous. On aperçoit la nacelle de métrologie.



3) Comment stabiliser l'architecture ?

L'obtention de franges d'interférences stellaires nécessite un réglage très fin de l'ensemble de la structure. Le point principal étant d'être capable de positionner les nacelles dans le ciel avec une précision millimétrique. Le défi est de taille étant donné que l'ensemble est rattaché à des câbles de plusieurs dizaines de mètres suspendus à un ballon d'hélium balloté dans le vent. Pour que le démonstrateur Carlina fonctionne, il faut obligatoirement qu'un point capital du montage soit rendu matériellement immobile: il s'agit du point C, centre de courbure du miroir primaire dilué et de la sphère focal sur laquelle doit se déplacer la nacelle d'acquisition. C'est la clef de voute du télescope! Le moindre mouvement de la nacelle de métrologie au centre de courbure (point C) entraine la perte des franges et du pointage de l'étoile.

Un système de réglage et d'asservissement a été étudié et mis en place pour réussir ce tour de force. L'objectif étant le placer exactement la nacelle au centre de courbure prévu pour les miroirs primaires (à 71 mètres au dessus du sol).
La nacelle de métrologie ('metrology gondola' dans le schéma principal) est constituée d'une poutre d'aluminium de 2 mètres (on parle aussi de nacelle 'poutre' pour la nacelle de métrologie) raccordée à un miroir parabolique et à deux 'coins de cube' (Fig 3). La nacelle doit remplir plusieurs conditions: il faut que le point d'attache des fils de la nacelle focale (autour desquels elle doit se déplacer pour rester sur la demi-sphère focale) soit parfaitement immobile et parfaitement positionné au point calculé théoriquement. Il faut aussi durant le réglage des miroirs (cf paragraphe suivant) que le foyer F du miroir parabolique soit pratiquement confondu avec le point C, centre de courbure du miroir virtuel primaire.

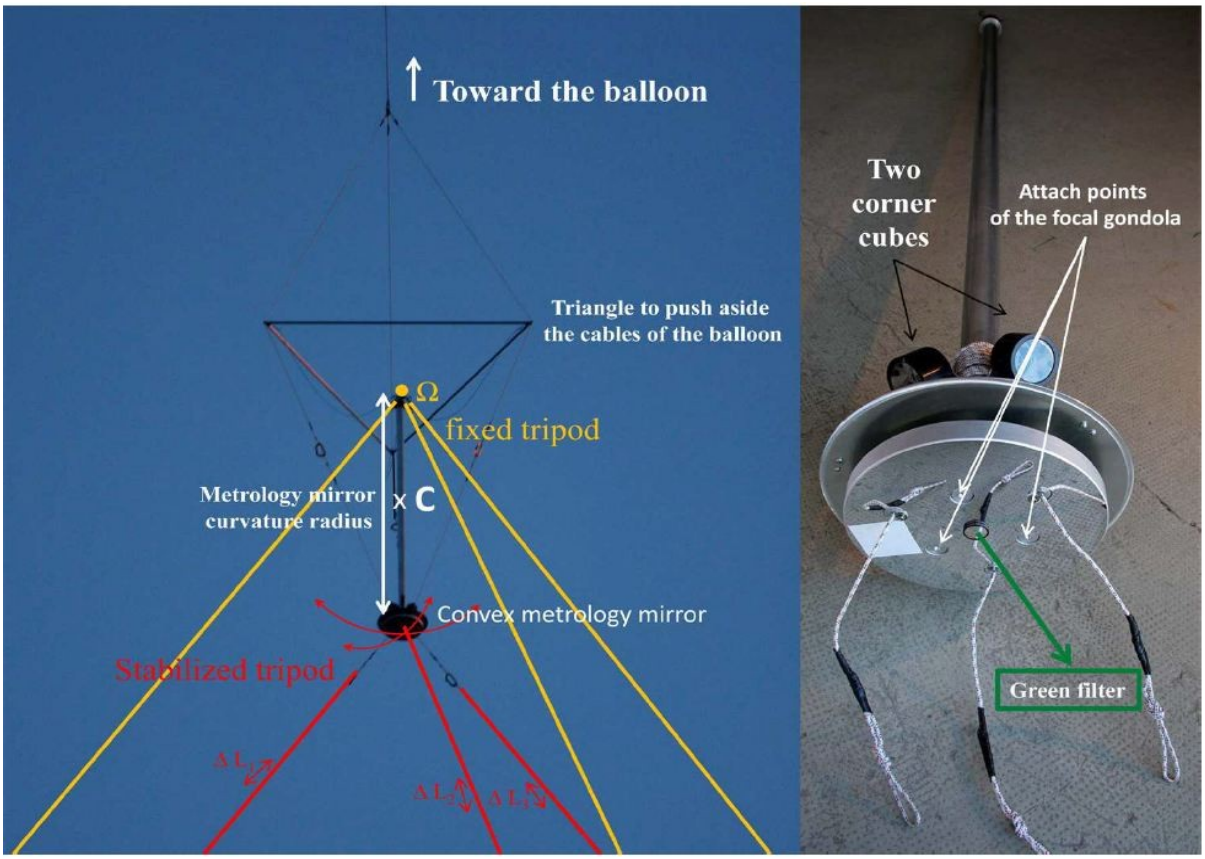


Figure 3

La nacelle est retenue au sol par un double trépied: trois câbles sont attachés en haut de la poutre au point oméga centre de courbure du miroir parabolique et trois autres en bas de la poutre au miroir parabolique (Fig 3). On utilise la verticale comme référence pour remettre toujours le télescope dilué en "station" de la même manière. La nacelle poutre est donc positionné verticalement par rapport à une table de métrologie au sol (Fig 4) et le milieu de cette poutre (foyer du miroir de métrologie) passe par le centre de courbure du miroir primaire sphérique.

La première étape consiste à régler manuellement le trépied du haut afin d'amener le centre de courbure omega sur la verticale recherchée. Pour cela, on tire un laser vert parfaitement vertical et on amène le rayon réfléchi par le miroir parabolique de la nacelle (concrètement, une tâche verte au sol) sur le point source (l'embouchure du laser): quand le laser est réfléchi sur lui même, on est certain qu'il passe (virtuellement) par le centre de courbure du miroir parabolique de la nacelle.

La deuxième étape consiste à amener le point d'attache bas de la nacelle sur la verticale du laser. Pour matérialiser optiquement ce point d'attache, un petit cercle a été dessiné au centre du miroir parabolique (cf photo précédente). Lorsque le laser touche cet endroit du miroir, le rayon réfléchi créé au sol une figure de diffraction très caractéristique. On règle grâce à des moteurs les câbles du trépied bas afin de superposer cette figure de diffraction à l'embouchure du laser (ce qui signifie que les rayons passant par le centre de courbure oméga du parabolique passent aussi par le filtre vert situé au point d'attache bas du trépied).

Lorsque les deux points sont alignés sur le laser vert, la nacelle de métrologie est parfaitement verticale et le foyer de son miroir parabolique est confondu avec le centre de courbure du primaire (cf paragraphe suivant 'comment co-sphériser les miroirs primaires').

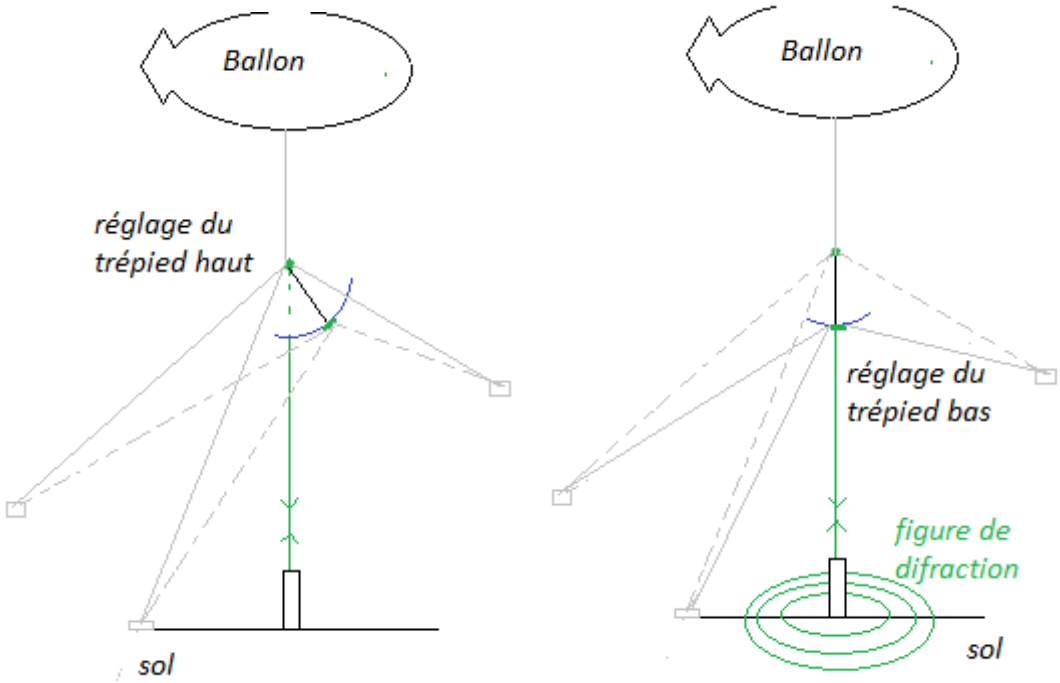


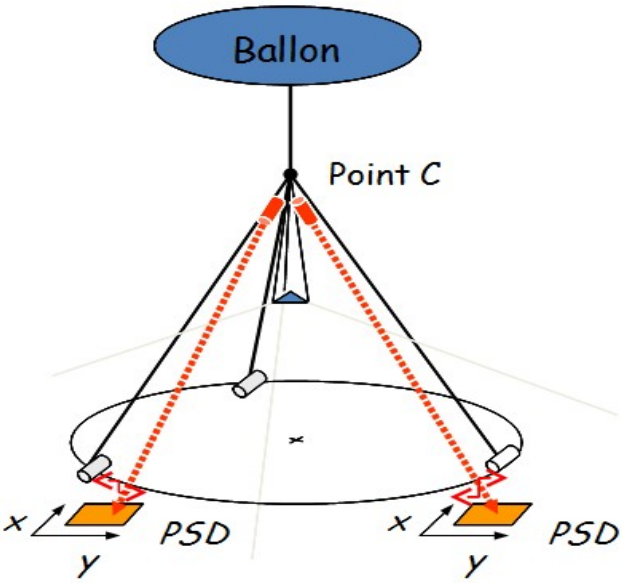
Figure 4

Une fois le réglage effectué, il faut rendre la nacelle immobile. L'architecture de la nacelle est telle que la traction du ballon s'applique sur le trépied du bas: un mouvement latéral du ballon fait tourner le miroir parabolique autour de son centre de courbure (à cause de la poutre). Par conséquent, le trépied haut reste mécaniquement quasiment immobile! Typiquement, sans asservissement, le haut de la poutre est immobile à 1 cm près tandis que le bas de la nacelle a des mouvements de 5 à 30 centimètres. Ces oscillations sont encore trop élevés pour espérer atteindre une rigidité suffisante. C'est pourquoi il a été mis en place un système d'asservissement du trépied bas.

Asservissement du trépied bas:

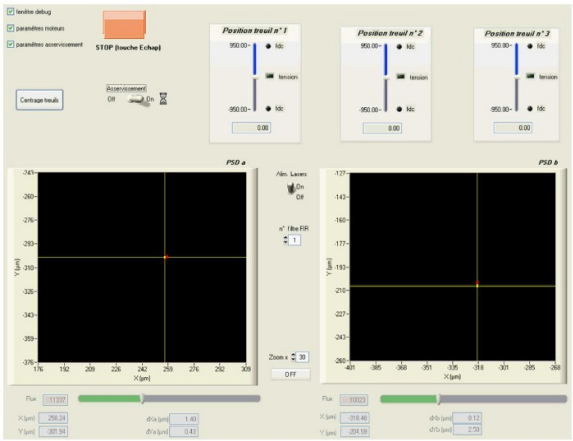
Les trois câbles du trépied bas sont reliés à trois moteurs au sol qui enroulent ou déroulent automatiquement les câbles pour maintenir la nacelle focale immobile.

Deux lasers rouges sont tirés depuis le sol en direction des deux 'coins de cube' (qui ont la propriété de réfléchir tout rayon lumineux sur eux même). Les rayons réfléchis sont recueillis sur un capteur psd (position sensitive detector). Selon la position du point d'impact, le programme d'asservissement calcule la position de la nacelle et active les moteurs-câbles pour la ramener à sa position adéquate. Cette opération s'effectue à 10 Hz et permet dans des conditions climatiques limites (jusqu'à 10 km/h de vent) d'**asservir la position de la nacelle à 5 mm pour le bas de la nacelle et 0,3 mm pour le haut.**



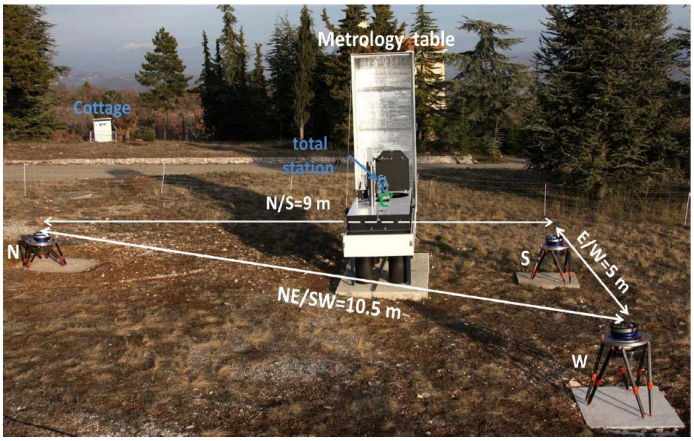
Ci contre, un schéma du système d'asservissement.

Ci dessous, une capture de l'interface du système d'asservissement. On note les points rouges sur les deux graphiques représentant l'impact des lasers rouges sur les PSD. Les deux barres vertes en dessous représente le flux sur les PSD.



4) Comment co-sphériser les miroirs au sol?

Carlina est composé de trois miroirs au sol formant une pupille triangulaire de cotés (appelés bases de l'interféromètre): 9 mètres sur 5 (hypoténuse de 10,5 mètres). La lumière stellaire se réfléchie sur les miroirs et interfère au niveau de la sphère focale.



Pour que les interférences puissent avoir lieu sur la demi-sphère focale, il faut que les miroirs primaires épousent au sol la forme de la sphère M1 virtuelle. H Le Coroller et Julien Dejonghe ont imaginé une système de métrologie interne permettant d'atteindre une précision de réglage inférieure à celle du bruit de l'atmosphère (i.e la différence de chemin optique induit par la précision du réglage des miroirs et de l'ordre de celle induite par les turbulences de l'atmosphère), L'idée étant d'utiliser les propriétés optiques d'un miroir sphérique, à savoir le stigmatisme parfait des rayons passant par le centre de courbure, et les propriétés physiques de la lumière blanche (sa longueur de cohérence de l'ordre du micron).

Dans un premier temps les trois miroirs de Carlina sont préréglés au sol grâce à une station totale (appareil de mesure des géomètres). Ce pré-réglage doit amener les miroirs à moins d'un millimètre de leur position adéquate.

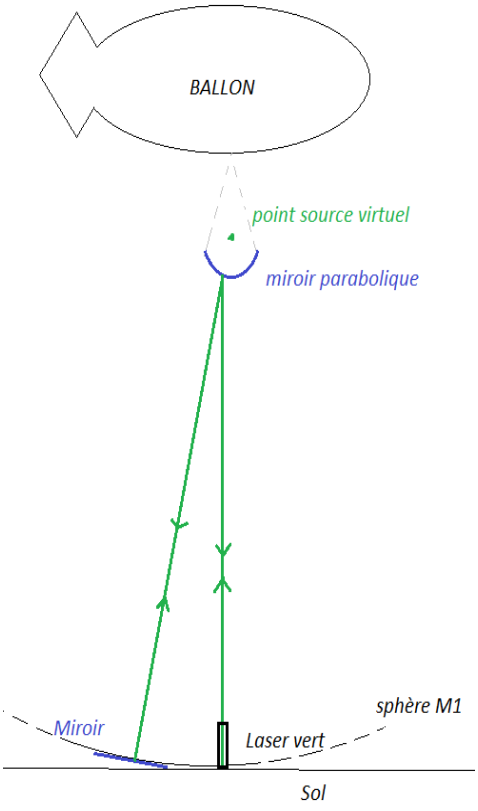
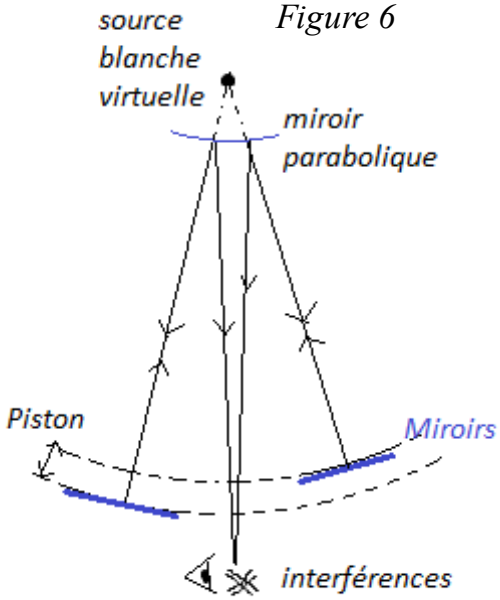


Figure 5

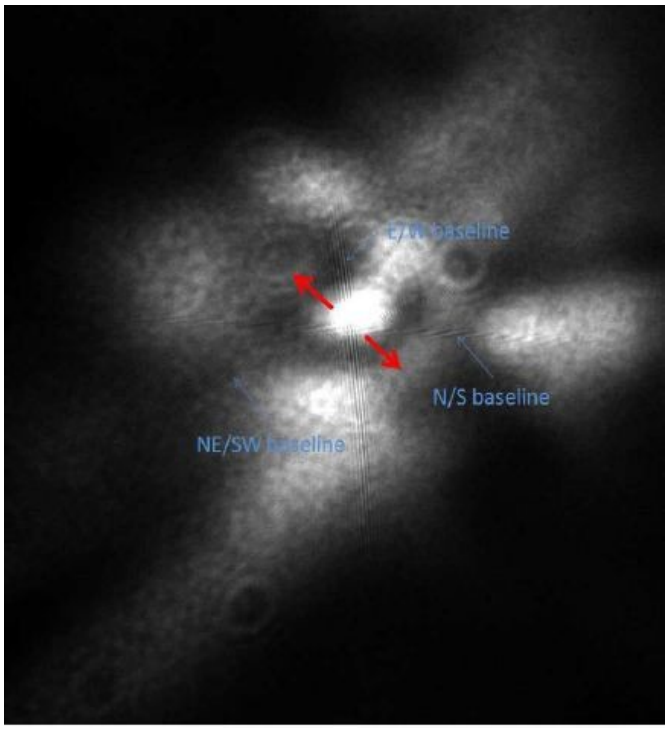
Réglage en tilt:
L'objectif de cette deuxième étape est de rendre chaque miroir tangentiel à une sphère ayant pour centre le point C (centre de courbure du miroir primaire défini par la position de la nacelle de métrologie). Pour cela, il suffit de se souvenir que tout rayon incident passant par le centre d'un miroir sphérique est réfléchi sur lui même. Ainsi, si on dispose une source laser au centre C et que l'on vise un miroir primaire, le rayon réfléchi doit passer lui aussi par le centre C. C'est cette propriété qui est exploitée dans le réglage en tilt.
Un laser vert large et très puissant est tiré à la vertical du sol en direction du miroir parabolique de la nacelle de métrologie. Le miroir parabolique créé alors un point source virtuel (cf propriété d'un miroir parabolique) en son foyer, c'est à dire au point C (cf le réglage de la nacelle). Cette source virtuelle éclaire les miroirs primaires au sol qui réfléchissent à leur tour les rayons lasers en direction du miroir parabolique de la nacelle. Le miroir parabolique de la nacelle renvoie les rayons au sol. Le réglage est correct si les trois taches correspondants aux trois miroirs se superposent après avoir été quasiment réfléchi sur eux même. Ce réglage s'effectue à la main en utilisant les vis de réglage des miroirs.

Réglage en piston:
Une fois que les trois miroirs sont sur des sphères possédant le même centre (mais ayant des rayons différent, typiquement de moins d'un millimètre), il faut amener les trois miroirs sur une seule sphère, c'est à dire modifier le réglage en piston des miroirs. Ce réglage s'effectue grâce à un laser blanc de cohérence de l'ordre du micron.
On remplace le laser vert par un laser blanc et on fait interférer deux à deux les taches superposées issues des miroirs différents (obtenues par le réglage précédent) en modifiant les vis de piston des miroirs. Lorsque les franges apparaissent et qu'elles sont centrées sur les taches, nous savons que la lumière a parcouru la même distance optique: nos trois miroirs sont co-sphérisés!

Figure 6



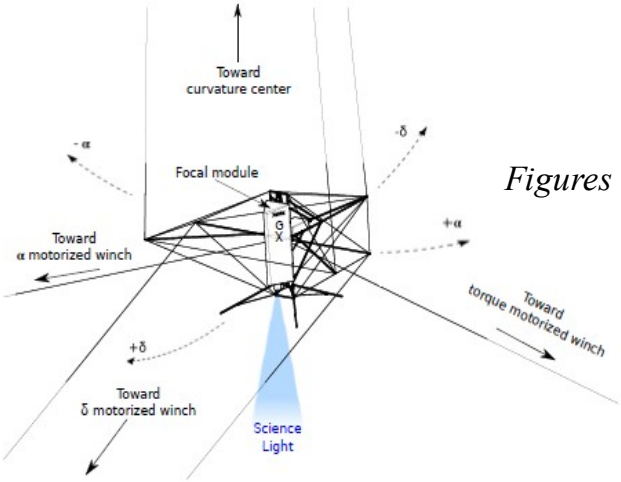
Les premières franges de métrologie ont été obtenues en février 2011 et ont fait l'objet d'un article publié dans A&A. L'obtention de ces franges blanches constituent un résultat important : tandis que le ballon oscille de plusieurs mètres dans le vent, le miroir de la nacelle de métrologie ne se déplace que d'une centaine de microns ! Ce résultat valide l'ensemble du système de métrologie interne et d'asservissement de la nacelle 'poutre'. Le prototype Carlina démontre qu'il est possible de co-sphériser la pupille par une méthode au centre de courbure et de stabiliser l'architecture à environ 100 microns près.



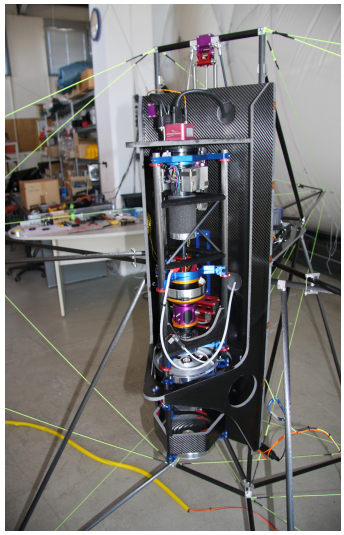
Ci-contre une photo du jeu de frange obtenu en février 2011. On reconnaît des spekles et trois jeux de franges correspondant aux trois bases. Les effets de moiré rendent les franges NE/SW difficilement visibles.

5) Comment acquérir les images?

L'acquisition des images se fait par l'intermédiaire de la nacelle focale (Fig 7 et 8). Cette nacelle se déplace le long de demi-sphère focale sous la nacelle de métrologie en tournant autour de l'axe polaire pour suivre le mouvement des étoiles. Elle est contrôlée depuis le sol par un autre trépied de câble.
Elle est composé de 3 éléments principaux:
→ un correcteur d'aberration de Sphéricité de Mertz. Il permet de corriger le stigmatisme imparfait du miroir sphérique primaire. Le correcteur de Mertz fait converger tout les rayons de l'étoile passant par le correcteur en un foyer unique.
→ un densifieur optique qui permet d'élargir les fronts d'ondes et de concentrer l'intensité lumineuse sur la caméra.
→ une caméra à comptage de photon recueille le flux lumineux entrant. Elle permet d'effectuer des temps de pose très courts (1/1000 s) nécessaires pour figer le mouvement des franges dû à la turbulence atmosphérique. Elle est aussi extrêmement sensible est permet de détecter quelques photons. Une autre caméra simple est incorporée à la nacelle pour faciliter le guidage sur les étoiles.



Figures 7 et 8



6) Comment pointer et suivre une étoile?

L'image d'une étoile se forme à l'intersection de l'axe 'centre de courbure du primaire-étoile' et de la demi-sphère focale. Il faut donc pouvoir diriger très précisément la nacelle focale sur ce point et être capable de suivre le mouvement de l'étoile dans le ciel (rotation autour de l'axe polaire).

Le pointage:
Dans un premier temps, on pointe l'étoile avec un télescope modifié positionné à coté de l'un des miroirs primaires. L'idée est d'utiliser le principe de retour inverse de la lumière: *si les rayon de l'étoile réfléchis sur le miroir primaire forme une image au foyer de la nacelle focale alors une source lumineuse placée au foyer de la nacelle focale sera réfléchi à l'infini en direction de l'étoile*.
À partir de cette idée, on place 4 LED rouges autour du foyer de la nacelle focale et un coin de cube sur le tube du télescope de pointage qui permet de capter les rayons lumineux venant en sens inverse (Fig 9 et 10). Le pointage est réussi lorsque l'on observe dans l'oculaire du télescope les 4 spots rouges issus des LED autour du spot de l'étoile. La position de la nacelle est modifiée à l'aide d'une télécommande agissant sur les câbles reliant la nacelle au sol.

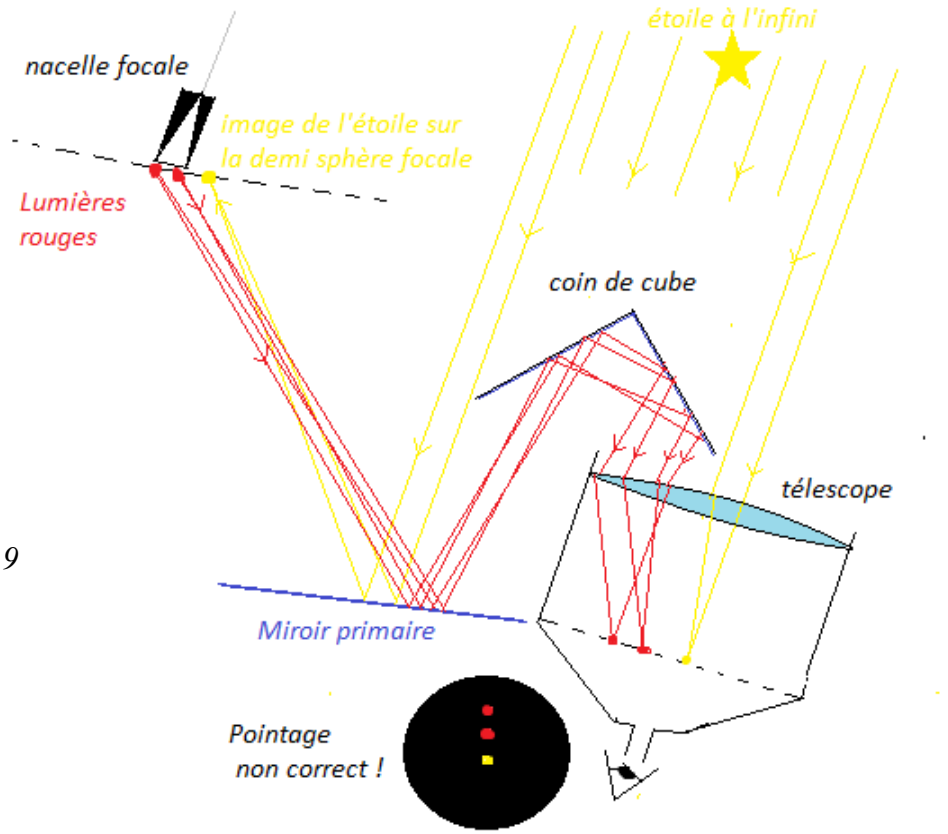


Figure 9

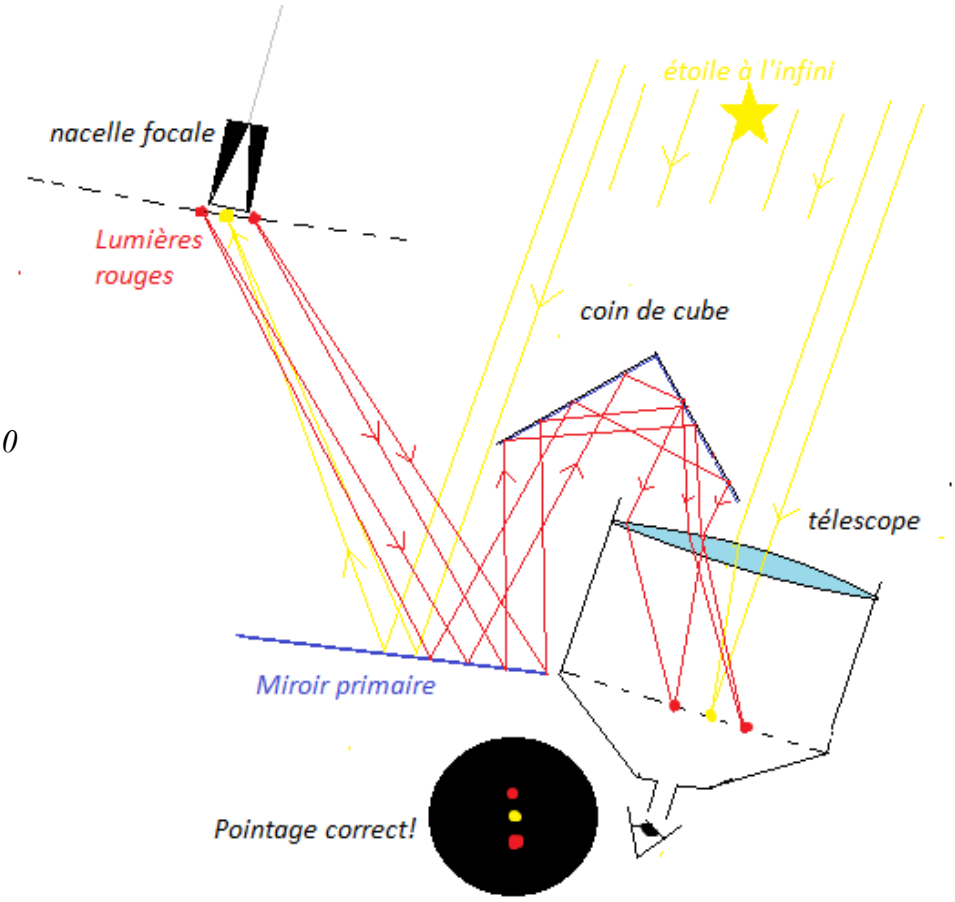


Figure 10

Le suivi de l'étoile:
Au cours de la nuit et à cause de la rotation de la terre, les étoiles 'tournent' autour de l'axe polaire. Pour que l'observation soit possible, il faut que la nacelle focale accompagne ce mouvement. La nacelle est guidée depuis le sol à l'aide d'un trépied de câble se rapprochant d'une **monture équatoriale** classique (Fig 11)

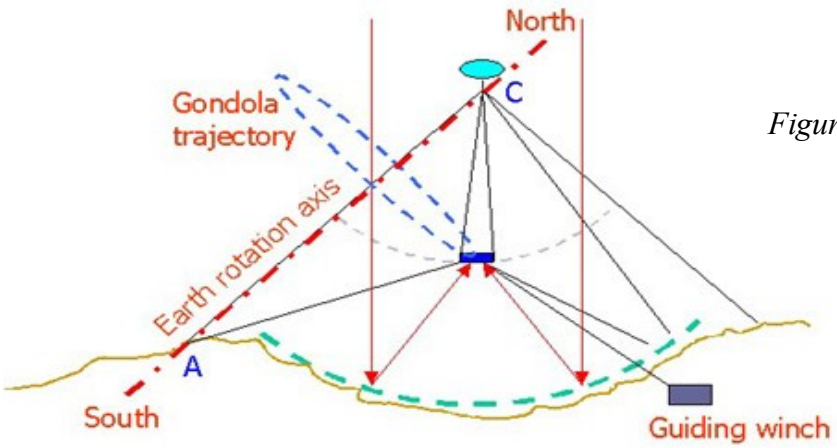


Figure 11

Sur le schéma, C représente le centre du miroir primaire virtuel où se trouve la nacelle de métrologie. La nacelle focale étant arrimée aux points A et C qui définissent l'axe polaire, elle tourne naturellement autour de cet axe équatorial. La longueur des câbles du trépied de guidage nous donne donc directement accès à la déclinaison et l'ascension droite de la direction pointée par Carlina. Le guidage se fait par le déroulement ou l'enroulement des câbles du trépied de guidage.
Le système de pointage et de guidage de la nacelle focale a été testé courant Juin 2012. les résultats obtenus ont été encourageants: le pointage fonctionne mais le guidage reste difficile. La présence du densifieur optique dans la nacelle oblige le guidage à s'effectuer dans moins d'une seconde d'arc! À titre de comparaison, le télescope de 2 mètres de l'OHP guide dans 3 secondes d'arc. Une telle performance nécessite une immobilité parfaite de la structure de Carlina ce qui n'a pu être obtenu durant les récentes nuits de test (la météo trop agitée faisant osciller le ballon de plusieurs mètres et donc la structure de quelques millimètres...).

6) Bilan des nuits de Carlina:

Durant mon stage, j'ai pu participer aux nuits de tests de Carlina. L'objectif était de valider les choix techniques effectués et de tester chacune des phases de réglage (co-sphérisation des miroirs, pointage, guidage, acquisition par la caméra à comptage de photons). Les nuits difficiles de Juin (trop courtes et trop agitées) ont rendu la tâche éprouvante mais les résultats obtenues ont été significatifs: la co-sphérisation des miroirs (franges de métrologie en lumière blanche) est désormais routinière, le système d'asservissement de la nacelle poutre a montré son efficacité malgré quelques problèmes techniques et des vents défavorables, le pointage de l'étoile par le télescope fonctionne mais se heurte occasionnellement à un problème de masquage par le ballon, le guidage en revanche s'avère difficile et pas assez précis. Une des solutions explorée actuellement serait de s'affranchir du densifieur optique présent dans la nacelle qui réduit considérable le champs pour observer les franges. Les franges stellaires escomptées n'ont donc pas pu être observées durant la période Mai-Juin. La prochaine période de test est prévue pour l'hiver 2012.

Partie II:

Les principes fondamentaux du traitement d'images en interférométrie.

On cherche à combiner la lumière stellaire capturée par des ouvertures (le plus souvent des télescopes) éloignées d'une distance B appelée Base. L'étude des franges obtenues et notamment de leurs contrastes permet de remonter aux caractéristiques de l'objet observé. Historiquement, la première mesure d'interférométrie stellaire permit à Michelson et Pease en 1920 de mesurer le diamètre de l'étoile Beltégeuse en utilisant la variation du contraste des franges en fonction de la distance entre deux sous-ouvertures. Dans les années 70, Antoine Labeyrie a réalisé pour la première fois des franges d'interférence dans le visible à l'aide de deux télescopes indépendants (I2T). L'interférométrie moderne permet d'observer des objets avec une résolution bien plus fine que celle des plus gros télescopes: Un interféromètre composé de deux petits télescopes éloignés d'une distance D a une résolution théorique égale à celle d'un télescope monolithique de diamètre D.

La structure des interféromètres actuels est composée d'un réseau de télescopes reliés par des 'lignes à retard' permettant d'égaliser les chemins optiques avant la recombinaison de la lumière (Fig 12). Durant le suivi de l'étoile, les lignes à retard se déplacent pour conserver l'égalité des chemins optiques entre les rayons lumineux provenant des différents télescopes.

Dans cette partie, je présente la base théorique de l'interférométrie. Je montre que l'étude des franges permet la mesure de la visibilité complexe de l'étoile observée. La démonstration est illustrée par des simulations programmées sous IDL. En annexe, je propose un calcul des différences de marches dans le cas d'une pupille co-sphérisée (de type Carlina).

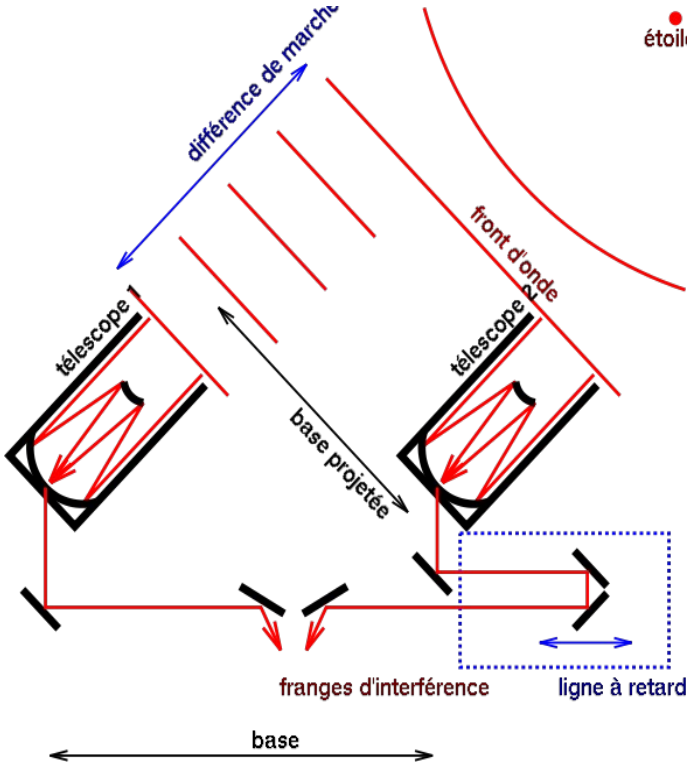


Figure 12

1) Relations fondamentales:

L'image d'une source étendue à travers un appareil optique est la superposition de l'image de chacun des points de la source (considérés comme incohérent entre eux).

$$I(x)=\int psf(x,y)\ \ o(y)\ \ dy$$

où I(x) est l'image au point x, psf(x,y) est la réponse impulsionnelle (*Point Spread Function*) de l'instrument, c'est à dire l'image créée en x par une source ponctuelle située dans la direction y, et o(y) est la valeur de l'intensité de l'objet dans la direction y.

Pour la plupart des appareils optiques, il existe un champ angulaire 'd'isoplanétisme', c'est à dire un domaine pour lequel la réponse impulsionnelle (PSF) de l'appareil+atmosphère est la même dans toutes les directions (ce qui signifie notamment que la lumière y traverse les mêmes couches d'atmosphère). Dans ce cas, la relation précédente devient simplement:

$$I(x)=\int psf(x-y)\ \ o(y)\ \ dy\ =\ psf(x)\ \circ\ o(x)$$

C'est la convolution de la réponse impulsionnelle de l'instrument avec l'objet observé.(' $\circ$ ' est l'opérateur convolution)
Cette relation fondamentale se traduit dans le domaine fréquentiel par la passage d'une convolution à un produit simple (la transformée de Fourier de l'opérateur convolution est le produit usuel)

$$TF(I(x))\ =\ TF(psf\circ o)\ =\ TF(psf)\ *\ TF(o)\ =\ PSF(u)*O(u)$$

Une simple division du spectre de l'image par la PSF (obtenue expérimentalement ou théoriquement en prenant le spectre de l'image d'un point infini) permettrait en théorie d'obtenir O(u) et donc de 'reconstruire' l'objet observé. Cette perspective séduisante est néanmoins impossible en pratique:

> La PSF prend en compte les turbulences atmosphériques. Elle est donc, elle aussi, une fonction très rapide du temps. Les relations précédentes ne sont valables que de manière instantanée. C'est toute la difficulté de la théorie des Speckle.

> La PSF(u) n'est non nulle que pour une gamme de fréquences restreinte. Nous verrons que les fréquences spatiales 'explorées' par un interféromètre sont les $B/\lambda f$ où les B représentent les longueurs des bases de l'interféromètre (distances entre les ouvertures de la pupille). On ne 'sonde' donc qu'une partie de la fonction O. En terme de résolution, λ/B est la largeur angulaire du détail le plus fin théoriquement visible par le télescope. On remarque immédiatement l'intérêt des interféromètres à bases très éloignées et la course actuelle aux télescopes géants.

2) Du réseau d'ouvertures à l'équation des franges

Le réseau d'ouverture au sol forme la pupille de diffraction à travers laquelle sera vue l'étoile. (c'est l'équivalent d'un écran percé de sous-ouvertures dans une expérience d'interférométrie classique).
Pour une étoile éloignée non résolue, l'image obtenue à l'infini (condition de Fraunhofer) par une tel dispositif (c'est à dire la psf=point spread function) est égale au module au carrée de la transformée de Fourier de la pupille:

$$I(x)=|TF(pupille)|^2$$

Si on néglige dans un premier temps la largeur des ouvertures, la pupille d'un interféromètre quelconque peut être vu comme une fonction P(x,y) valant 1 à l'endroit des ouvertures et 0 dans tout le reste du plan pupille:

$$P(x,y) = \sum_{1}^{N} \delta(x-x_k, y-y_k) e^{i\phi_k}$$

où les (x_k, y_k) sont les coordonnées des N ouvertures (des télescopes dans le cas classique, des miroirs au sol dans le cas de Carlina). Une phase est ajoutée pour chaque ouverture. La transformée Fourier complexe de la pupille est alors:

$$TF(P)(u,v) = \int\int P(x,y)\ e^{-i\frac{2\pi}{\lambda f}(xu+yv)}\, dx\, dy = \sum_{1}^{N} e^{i\phi_k - i\frac{2\pi}{\lambda f}(x_k u + y_k v)}$$

f est la distance écran-pupille. Les conditions de Fraunhofer exige que f>>Base
On calcule le module au carré:

$$|TF(P)|^2(u,v) = TF(P)\overline{TF(P)} = \left(\sum_{1}^{N} e^{i\phi_k - i\frac{2\pi}{\lambda f}(x_k u + y_k v)}\right)\left(\sum_{1}^{N} e^{-i\phi_k + i\frac{2\pi}{\lambda f}(x_k u + y_k v)}\right)$$

d'où:

$$psf(\vec{U}) = N + \sum_{(i,j)}^{N(N-1)/2} 2\cos\left(\frac{2\pi}{\lambda f}\ \vec{U}.\vec{B}_{ij} + \phi_{i,j}\right)$$

où U est le vecteur position sur l'écran image (u,v) (à ne pas confondre avec le plan (u,v) du spectre) et les B sont les vecteurs 'Base' du réseau d'ouverture. (Les B relient les ouvertures deux par deux).
L'image est donc la superposition d'une composante continue proportionnelle au flux total du signal (c'est à dire à la surface collectrice totale, ici, le nombre de sous-ouvertures) et de N(N-1)/2 jeux de franges sinusoïdaux correspondant aux recombinaisons par paire de base. L'orientation de chaque jeux de franges est perpendiculaire au vecteur B dont il est issu.

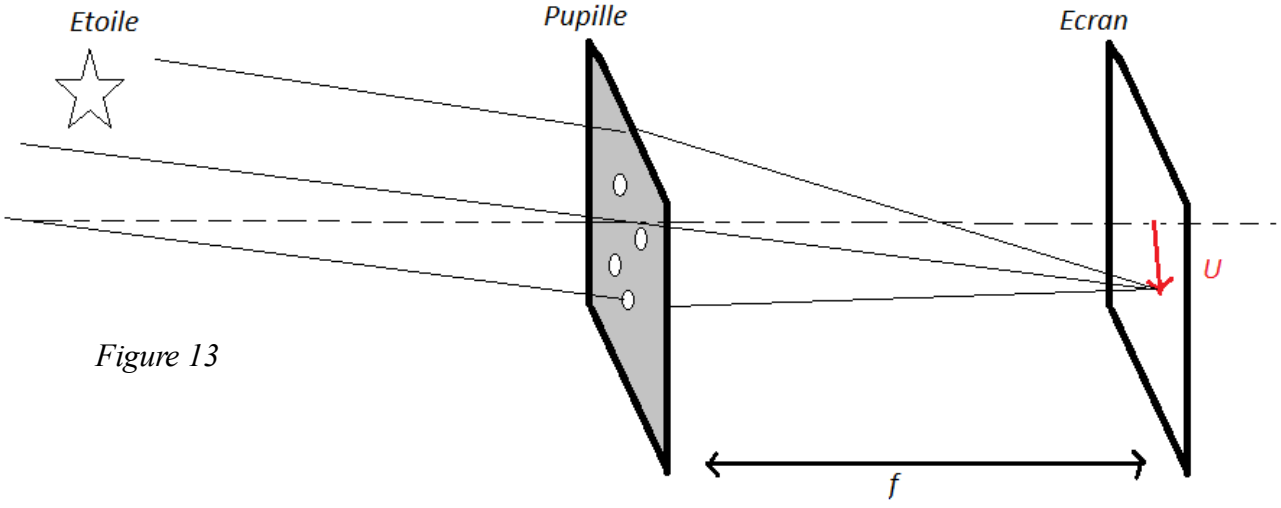


Figure 13

Quand l'objet est résolu et dans l'hypothèse d'isoplanisme, nous avons prouvé que l'image était donnée par la convolution de la psf précédente avec la fonction 'objet' o donnant la valeur de l'intensité lumineuse 'locale' en fonction du point de l'objet visé.

$$Image(\vec{U}) = \int o(\vec{V}) \times \left(N + \sum_{(i,j)}^{N(N-1)/2} 2\cos\left(\frac{2\pi}{\lambda f} (\vec{U}-\vec{V}).\vec{B_{ij}} + \phi_{i,j}\right)\right) dV$$

¿

[Notons que le vecteur V ne donne pas directement une direction d'observation. L'angle correct est (V/f). cf Fig 13]

o(V) donne l'intensité lumineuse en un point de l'objet situé dans la direction V (en fait V/f). On la note I(V). l'intensité totale est I0. On a la relation suivante entre I et I0:

$$I_0 = \int I(\vec{V}) dV$$

En développant l'expression de la fonction image précédente, on a:

$$Image(\vec{U}) = N \times I_0 + \sum_{(i,j)}^{N(N-1)/2} \int I(\vec{V}).2\cos\left(\frac{2\pi}{\lambda f} (\vec{U}-\vec{V}).\vec{B_{ij}} + \phi_{i,j}\right) dV$$

que l'on peut écrire sous forme complexe:

$$Image(\vec{U}) = N \times I_0 + \sum_{(i,j)}^{N(N-1)/2} \Re\left\{ \int I(\vec{V}).2e^{i\phi_{i,j} + \frac{i2\pi}{\lambda f} \vec{U}\vec{B_{i,j}}}.e^{\frac{-i2\pi}{\lambda f}\vec{V}.\vec{B_{ij}}} dV \right\}$$

On sort de l'intégrale le terme de frange:

$$Image(\vec{U}) = N \times I_0 + \Re \sum_{(i,j)}^{N(N-1)/2} 2e^{i\phi_{i,j} + \frac{i2\pi}{\lambda f} \vec{U}\vec{B_{i,j}}} \left|\gamma(B_{i,j})\right|.e^{\frac{-i2\pi}{\lambda f}\alpha(B_{i,j})}$$

où

$$\int I(\vec{V}).e^{\frac{-i2\pi}{\lambda f}\vec{V}.\vec{B_{ij}}} dV = \left|\gamma(B_{i,j})\right|.e^{\frac{-i2\pi}{\lambda f}\alpha(B_{i,j})} \qquad \textbf{(Eq. 1)}$$

est **la visibilité complexe de l'étoile fonction du vecteur B.**

On peut désormais revenir dans le domaine réel. **L'équation des franges est:**

$$Image(\vec{U}) = N I_0 + \sum_{(i,j)}^{N(N-1)/2} 2.\left|\gamma(B_{i,j})\right| \cos\left(\frac{2\pi}{\lambda f} \vec{U}.\vec{B_{ij}} + \phi_{i,j} - \alpha(B_{i,j})\right)$$

On cherche à calculer la visibilité complexe de l'étoile, c'est à dire, comme indiqué par l'équation 1, la transformée de Fourier de sa distribution spatiale en intensité aux fréquences B/lambda.

Plus on possède de bases, plus on a de fréquences spatiales explorées. Les interféromètres utilisent le plus souvent la rotation de la terre pour augmenter le nombre d'angles de vue (tout se passe comme si les bases B tournaient). C'est ce que l'on appelle **la synthèse d'ouverture**.

3) De l'équation des Franges à l'extraction du module de la visibilité:

Décomposons nos franges en série de Fourier. Pour cela , il suffit d'écrire les cosinus en écriture complexe:

$$image(\vec{U}) = N I_0 + \sum_{(i,j)}^{N(N-1)/2} |\gamma(B_{i,j})| \cdot \left(e^{i\alpha_{i,j} - i\phi_{i,j}} e^{i\frac{2\pi}{\lambda f}\vec{U}.\vec{B_{i,j}}} - e^{-i\alpha_{i,j} + i\phi_{i,j}} e^{-i\frac{2\pi}{\lambda f}\vec{U}.\vec{B_{i,j}}} \right)$$

On reconnaît immédiatement la décomposition en série de Fourier dans un spectre à deux dimensions. Les pics apparaissent aux fréquences spatiales B/(lambda*f).
La transformée de Fourier des franges a donc comme équation:

$$TF(image)(\vec{v}) = N I_0 \delta(\vec{v}) + \sum_{(i,j)}^{N(N-1)/2} \delta\left(\vec{v} \pm \frac{\vec{B_{i,j}}}{\lambda f}\right) \cdot |\gamma(B_{i,j})| e^{\pm(i\alpha_{i,j} - i\phi_{i,j})}$$

On s'intéresse au module de cette TF:

$$|TF(image)|^2(\vec{v}) = (N I_0)^2 \delta(\vec{v}) + \sum_{(i,j)}^{N(N-1)/2} |\gamma(B_{i,j})|^2 \delta\left(\vec{v} \pm \frac{\vec{B_{i,j}}}{\lambda f}\right)$$

Le module au carré de la TF des franges nous donne donc une structure en 'pic'. Si les B sont tous différents (bases non redondantes), on peut obtenir une estimation du module carré de la visibilité dans une direction B en effectuant le rapport d'un des deux pics secondaires associés sur le pic principal, le tout multiplié par N^2:

$$\frac{N^2 \times PIC(B_{i,j})}{PIC0} = \frac{|\gamma(B_{i,j})|^2}{I_0^2}$$

C'est la relation fondamentale du calcul de visibilité. Nous venons en fait de démontrer le **théorème de Van-Citter Zernike** (équation 1) et de montrer que **l'étude du spectre des franges permet d'obtenir une partie de la transformée de Fourier de l'étoile. C'est le principe du traitement des images en interférométrie!**

remarque: Il est possible de démontrer cette formule sans passer par l'expression analytique des franges après convolution. Nous savons en réalité que:

$$image = psf \circ objet$$

La transformée de Fourier d'une convolution étant un produit, nous avons:

$$|TF(image)|^2 = |TF(psf)|^2 \times |TF(objet)|^2$$

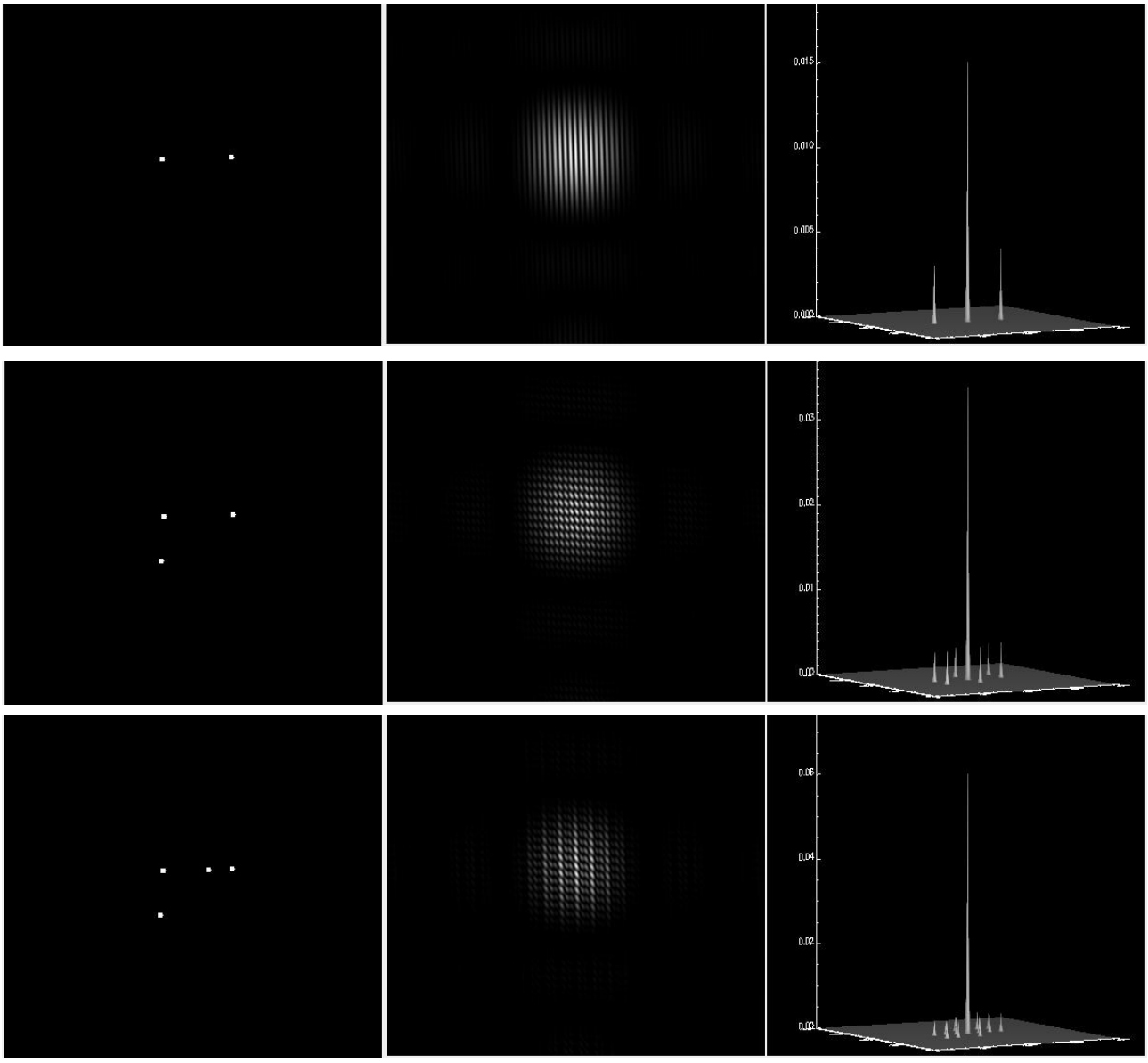
Cette fonction est non nulle uniquement à l'endroit des pics de |TF(psf)|. On effectue ensuite le rapport des pics secondaires sur le pic central:

$$\frac{|TF(image)|^2(PIC\,n)}{|TF(image)|^2(PIC\,0)} = \frac{|TF(psf)|^2(PIC\,n)}{|TF(psf)|^2(PIC\,0)} \times \frac{|TF(objet)|^2(PIC\,n)}{|TF(objet)|^2(PIC\,0)}$$

=N^2

cqfd

L'image ci dessous présente les résultats d'une simulation effectuée sous IDL. On crée sur l'écran de gauche un réseau d'ouvertures non redondant et on lit sur les écrans qui suivent la psf et le spectre obtenus. Puisque les franges ne sont convoluées avec aucun objet, **leur visibilité est de 1**. La vue 3d du spectre permet de juger de la hauteur relative des pics. Le calcul numérique de ce rapport confirme la loi d'évolution du rapport pic secondaire sur pic principal en N^2



4) Première approche à une reconstruction de l'objet:

La connaissance du module de la visibilité complexe ne suffit pas en théorie a obtenir les informations suffisantes pour reconstruire l'objet. Il faut en effet connaître la 'phase' de cette visibilité complexe. Hors l'atmosphère ajoute un terme de phase dans les visibilités complexes. L'information sur l'objet observé est donc en partie perdue. Des méthodes mathématiques existent pour récupérer une partie de l'information, notamment la méthode dite de Knox-Thompson (à base de triples corrélations des franges). Nous ne présenterons pas ici ces méthodes qui font l'objet de thèse; les objectifs premiers des observations stellaires avec Carlina étant de prouver la faisabilité du projet (obtenir des franges et montrer que l'on peut suivre une étoile) et caractériser les performances du télescope (la qualité des observables: stabilité, module de la visibilité etc).

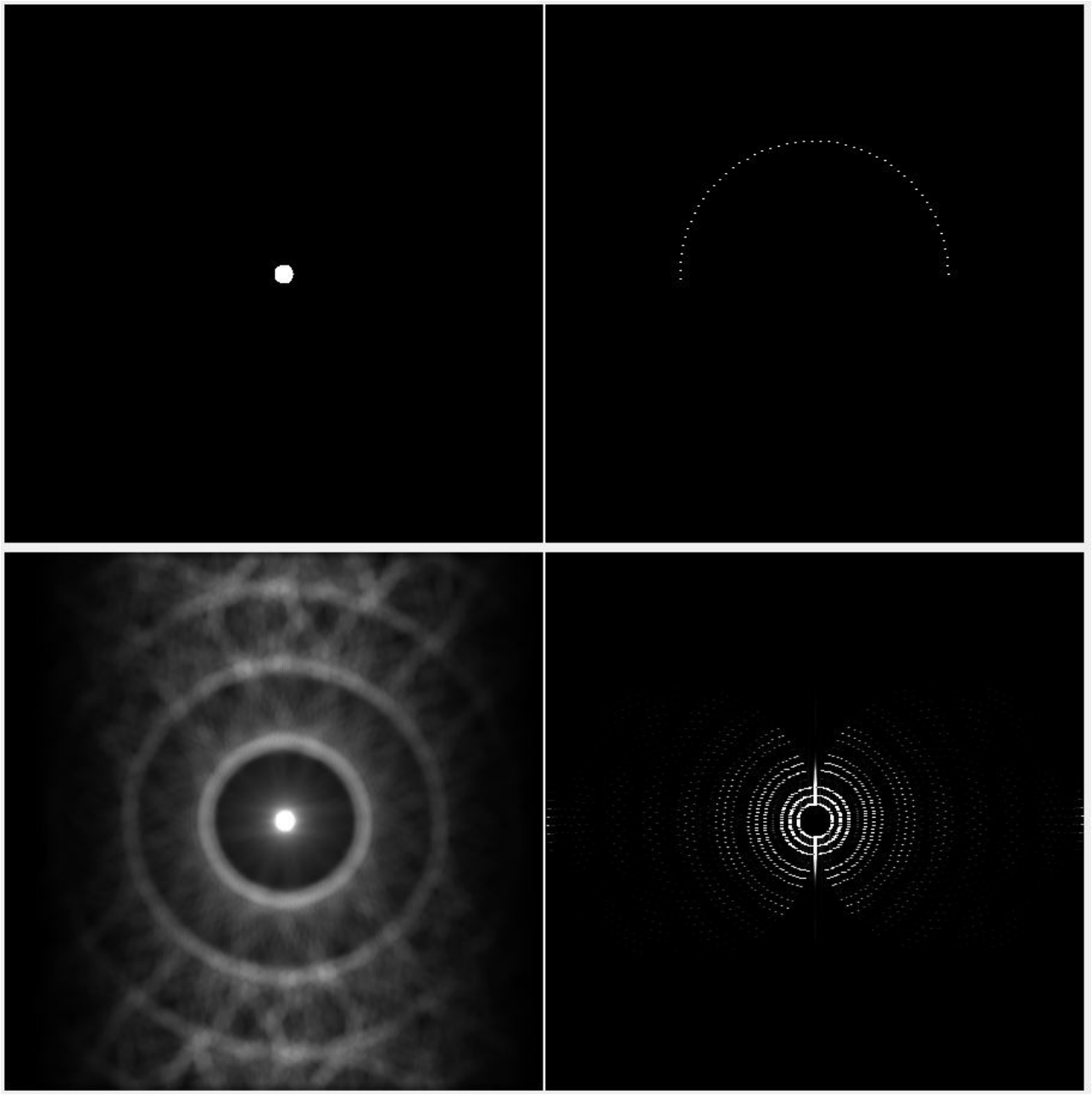
Si nous obtenons la visibilité complexe de l'étoile (c'est à dire sa phase et son module) pour un grand nombre de base, nous pouvons espérer reconstruire une image de l'objet observé. Rappelons en effet que la visibilité complexe n'est autre (Eq. 1) que la transformée de Fourier de la distribution spatiale en intensité de l'étoile. Dans le cas limite où le nombre de bases est infini, nous obtenons entièrement cette transformée de Fourier (toutes les fréquences B sont 'balayées'). Une simple TF inverse nous donne l'image de l'objet observé (ce qui est logique puisque dans le cas des ouvertures infinis, le jeu de franges obtenu est en fait l'image directe de l'objet).

Dans certains cas, le module de la visibilité complexe suffit à caractériser l'objet. On utilise le plus souvent un modèle théorique de l'objet observé que l'on ajuste en fonction du module de la visibilité observé.
Les simulations qui suivent illustrent la démonstration précédente.

Premier exemple: un disque:

L'objet est un disque lumineux (une étoile) d'intensité totale I0. Dans ce cas, la transformée de Fourier de sa distribution en intensité est une fonction de Bessel d'ordre 1. En mesurant par interférométrie une valeur de visibilité, on peut par exemple remonter au diamètre de l'étoile.

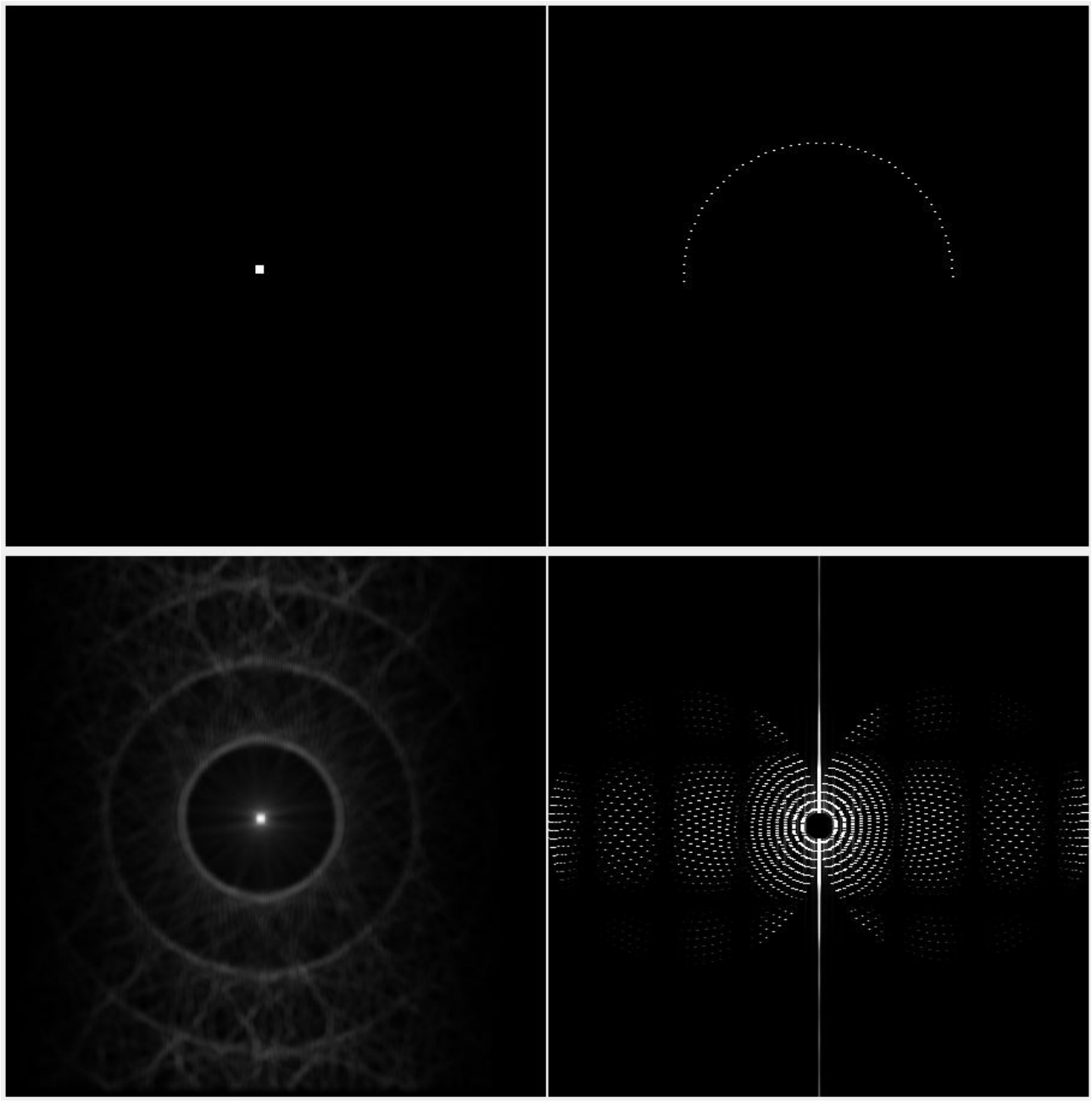
Méthode de simulation: pour chaque pupille je calcule la psf par transformée de Fourier (j'utilise la fonction FFT d'IDL). On obtient un jeu de frange similaire à ceux obtenus dans les simulations de la page précédente. On convolue ensuite ces franges avec l'objet (le disque) conformément à la théorie de interférences. On obtient alors une image qui constitue une simulation de l'image observée à travers l'interféromètre.
Le spectre est quand à lui obtenu en calculant la transformée de Fourier des Franges. Puisque le module de la visibilité est égal à la visibilité elle même (objet symétrique → visibilité réelle), on 'lit' directement sur le spectre la transformée de Fourier de l'objet (à un facteur multiplicatif près).



L'objet est le cercle (non parfait à cause de la forme des pixels) situé en haut à gauche. Le réseau d'ouverture est composé de 50 pixels blancs disposés en demi cercle (motif non redondant) en haut à droite. L'image obtenue est en bas à gauche et son spectre en bas à droite (Le disque noir au centre du spectre est un cache ajouté par le programme pour éliminer le pic central non significatif)

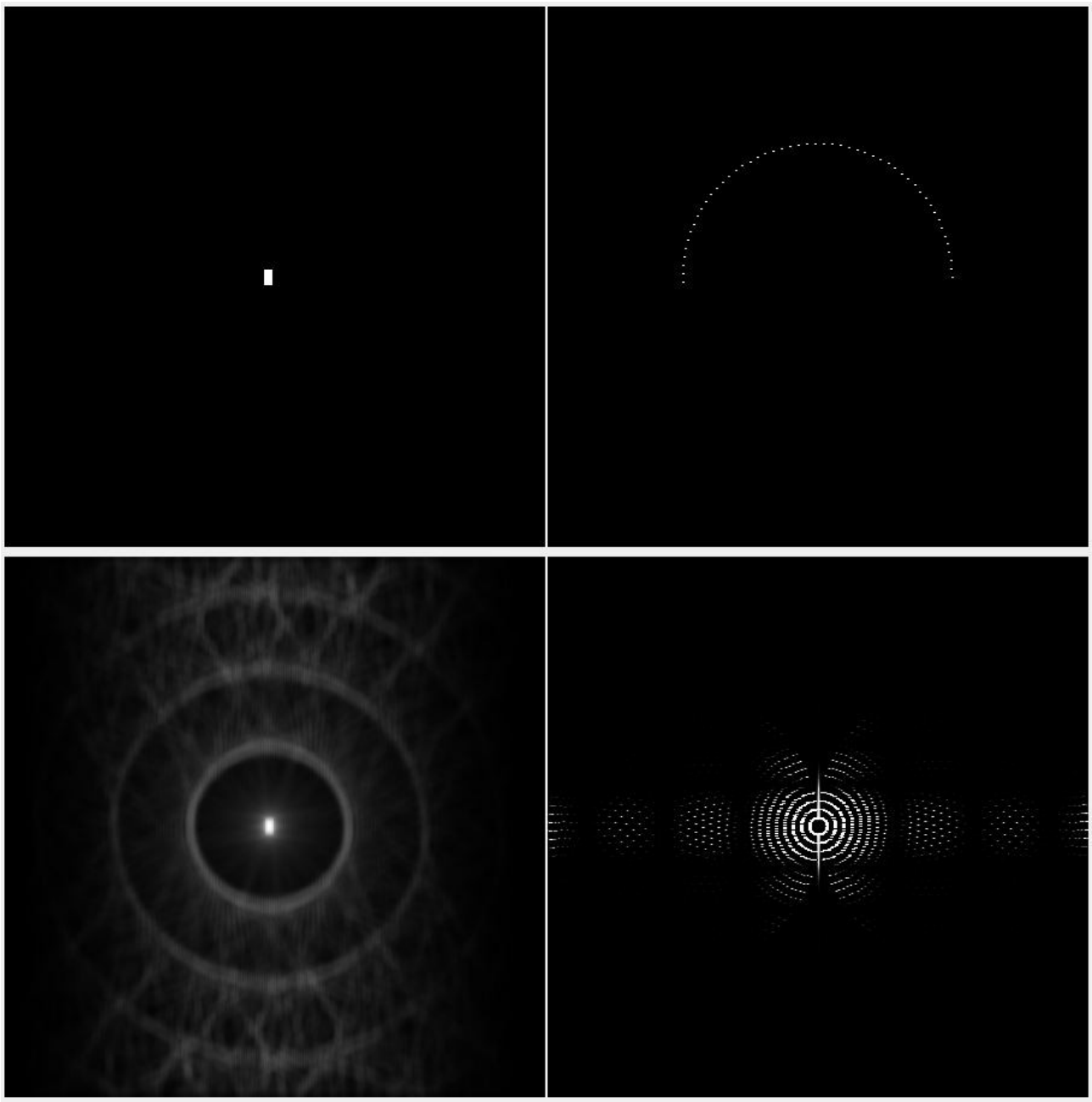
On voit apparaître dans le spectre la forme d'un tâche d'Airy, transformée de Fourier de l'objet observé. Chaque point du spectre correspond à une 'base' de la pupille. Dans le cas présent d'une pupille a 50 miroirs, cela fait 50*49 = 245 pics! La fonction de Bessel n'est donc ici représentée que sur 245 points.

Deuxième exemple, un carré:



Le spectre des franges fait apparaître la transformée de Fourier de l'objet carré (similaire à une tache de diffraction par une ouverture carrée).

Troisième exemple: un rectangle:



Partie III:

Extraction du module carré de la visibilité complexe:

Création d'un logiciel de traitement d'images.

L'observation d'interférences stellaires depuis le sol est rendue difficile par le phénomène de tavelures. Les turbulences atmosphérique induisent à chaque instant une variation aléatoire des indices de réfractions de l'air qui entraine une variation à l'échelle de la longueur d'onde des chemins optiques parcourus par les rayons lumineux traversant l'atmosphère. La perte de la cohérence spatiale des faisceaux lumineux (la déformation du front d'onde) crée sur l'image des motifs 'tachetés' (speckle) rapidement variables (typiquement de l'ordre de 1000 Hz) issus des interférences entre les rayons qui ont traversé différentes couches de l'atmosphère. Dans une expérience d'interférométrie, tout se passe alors comme si le facteur de transmission de la pupille voyait sa phase varier quasi-aléatoirement en chaque point et à chaque instant (de 100 à 1000 fois par seconde). En terme d'acquisition de données, cela implique qu'il faut utiliser un temps de pose assez court pour 'figer' l'atmosphère et rendre le traitement de l'image possible. Dans le cas contraire, l'image devient rapidement inexploitable: Supposons que la fréquence 'speckle' soit de 1000 Hz et que l'on choisisse un temps de pose de 0,1 seconde, notre image sera alors la somme en intensité de 100 images qui ont bougé les unes par rapport aux autres!

En 1974, A. Labeyrie obtient des franges d'interférence sur l'étoile Vega avec un interféromètre à deux téléscopes (I2T à Calern): C'est le début de l'interférométrie moderne.

Dans cette partie, je présente la méthode d'extraction des visibilités à partir d'images issues d'une caméra à comptage de photon (simulant les franges stellaires de Carlina). Nous verrons que ce type d'instrument s'il permet un temps de pose d'une milliseconde crée aussi une anomalie dans l'image qui doit être corrigée: c'est le 'trou de photons'. Je présente la méthode que j'ai mise en place dans le cas particulier de Carlina pour pallier cette difficulté et extraire le module carré de la visibilité complexe d'une étoile à partir des images de la caméra. En annexe, vous trouverez le code complet du logiciel que j'ai développé à cet effet ainsi qu'une capture d'écran présentant son fonctionnement.

1) La caméra à comptage de photons.

L'acquisition des franges stellaires nécessite une caméra à temps de pose ultra-court (1/1000 s) et capable de travailler à de faibles flux (typiquement 1000 photons par pose) sans que l'image ne soit dominée par le bruit de la caméra (bruit thermique, bruit de lecture etc). Le prototype Carlina de l'observatoire de Haute Provence utilise une caméra à comptage de photon placée dans la nacelle focale et reliée au sol par une fibre optique. L'image obtenue donne la position des 'impacts' photons-écran sans considération d'énergie.
La caméra est composée de deux parties principales: un intensificateur permettant de transformer l'impact d'un photon sur le capteur en une vague d'électrons (gain de 10^6) et un réducteur à fibre optique associé à un CCD permettant de reconstituer l'image et de calculer les coordonnés des impacts.

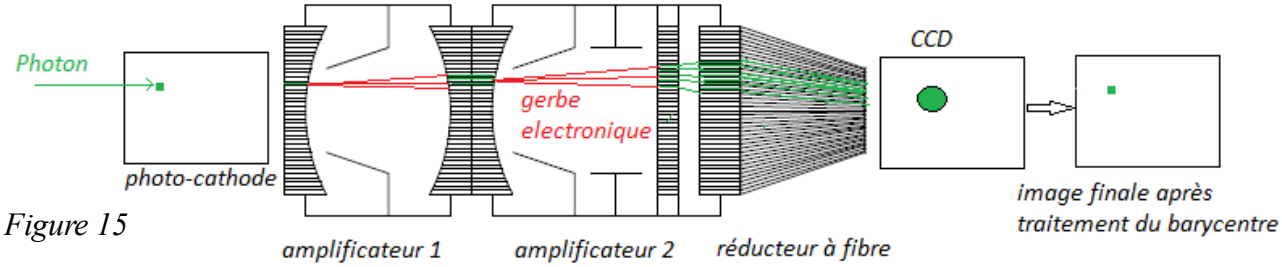


Figure 15

L'intensificateur (Fig 15):
il est composé de 3 parties:

- une photo-cathode d'entrée. Lorsqu'un photon percute la photo-cathode, il 'arrache' un ou plusieurs électrons secondaires à l'entrée de l'amplificateur. Son rendement quantique est de l'ordre de 10%.
- l'amplificateur. Il accélère la gerbe d'électrons (par application d'une tension) et la focalise à l'intérieur du tube. Le gain obtenue est typiquement de 10^5
- Le 'phosphore'. La gerbe d'électron accéléré percute l'anode composé d'une plaque électro-luminescente qui émets des photons (typiquement, un électron accéléré libère 500 photons)

Le réducteur à fibre optique et le CCD:
Les photons émis par le phosphore de l'amplificateur sont canalisés dans des fibres optiques qui permettent d'adapter le flux de sortie de l'amplificateur à la taille des pixels de la CCD.
Chaque photo-évènement détecté par la caméra se traduits sur le CCD par une 'tâche' d'énergie (typiquement de 10^6 photons) s'étalant sur plusieurs pixels (typiquement sur 3*3). Une électronique permet de calculer le barycentre de cette tâche avec une précision d'un quart de pixel ce qui permet d'obtenir la résolution réelle de l'appareil (celle de la photo-cathode)

Ci-contre (Fig 16): image obtenue en laboratoire avec la caméra à comptage de photons de Carlina. Elle se présente comme un tableau de pixels valant 1 (pixel allumé, photon détecté à la position (x,y) du pixel), ou 0 (pixel éteint, pas de photons détecté).

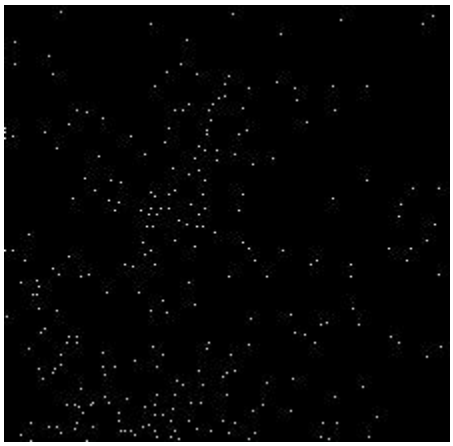


Figure 16

2) Le biais de photons:

Lorsque deux photons proches heurtent la caméra à comptage de photon, leurs tâches d'énergie reçus sur le CCD peuvent se 'chevaucher' et le système ne 'voit' alors qu'un seul photon auxquels il associe une coordonnée (x,y) correspondant au barycentre des deux impacts réels. Il en résulte une erreur sur le nombre de photons total et leurs coordonnées appelé *biais de photon (Fig 17).*

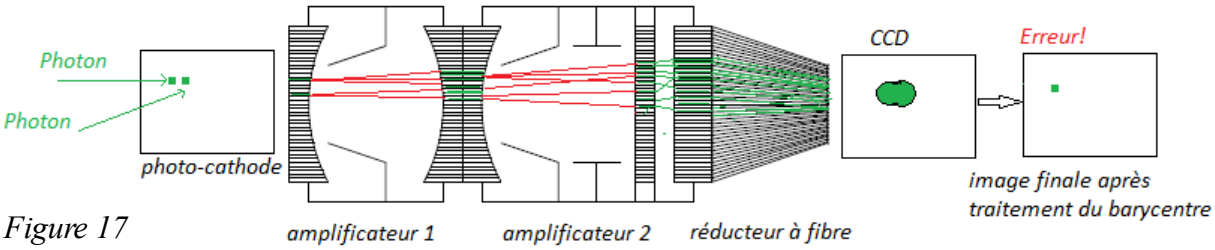


Figure 17

Pour exploiter les images de la caméra avec précision (ce qui est le cas en traitement d'images d'interférométrie) il faut donc limiter sinon résoudre ce problème. Une première mesure nécessaire est de travailler dans des régimes de faible flux photonique afin de réduire la probabilité d'avoir deux photons proches. On appelle ce régime *photon-starved regime*.
Néanmoins, le biais de photon est toujours présent est sa conséquence la plus visible apparait au moment de l'auto-corrélation des images sous la forme d'un 'trou de photon'.

3) Le trou de photons.

Le trou de photon désigne la forme caractéristique apparaissant au centre de l'auto-corrélation des images de la caméra à comptage de photons et qui s'apparente à une gaussienne inversée. Le trou de photon dans l'autocorrélation est une conséquence du biais de photon dans l'image.

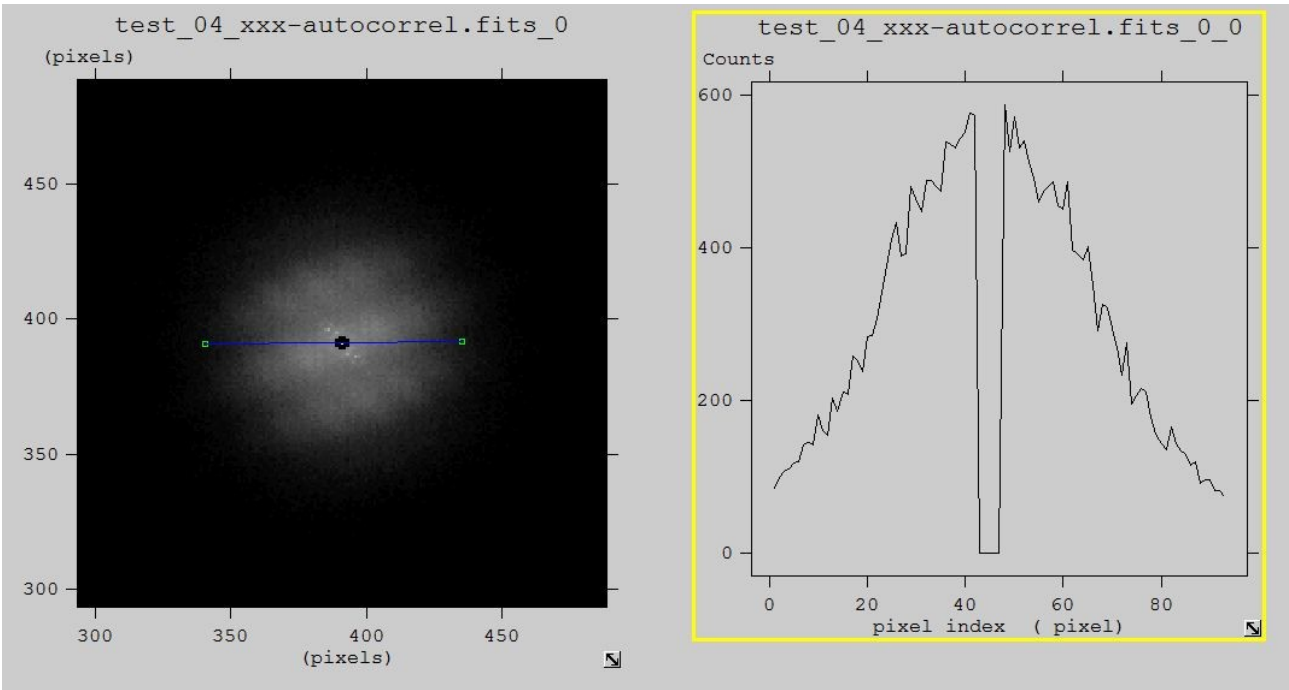


Fig 18: Profil de l'autocorrélation d'une somme d'image obtenues avec la caméra à comptage de photon. Un 'trou de photon' apparaît au centre de l'image.

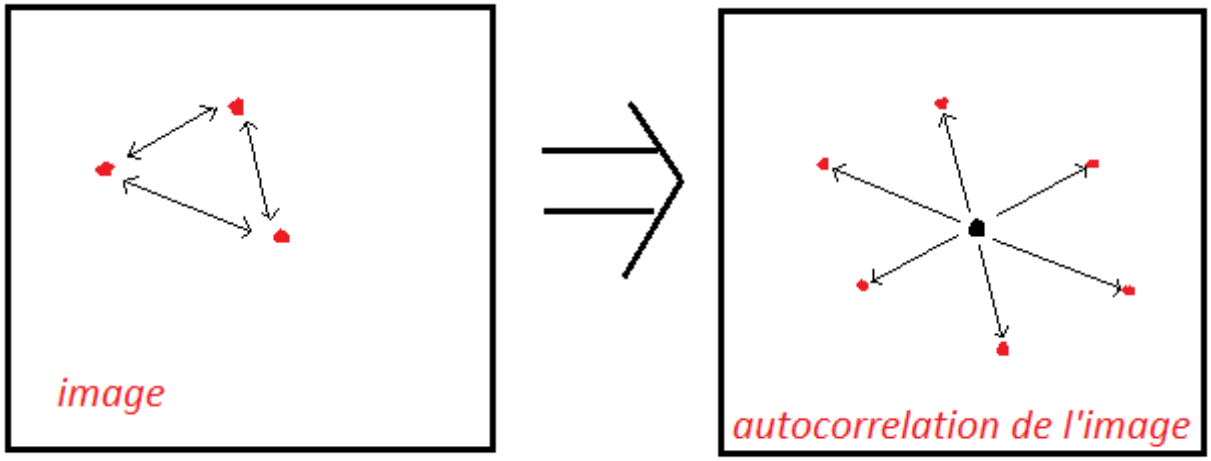
Rappel sur l'auto-corrélation:

L'auto-corrélation d'une image f(x,y) est donnée par $a(u,v)=\iint f(x,y)\overline{f(x-u,y-v)}\,dx\,dy$.

Dans le cas d'une caméra à comptage de photon, l'image f(x,y) peut être écrite comme étant la somme de fonction de Kronecker: $f(x,y)=\sum_{1}^{n}\delta(x-x_i\ ,y-y_i)$

où les xi, yi sont les coordonnées des photons détectés et n le nombre de photons détectés.
Dans ce cas, on remarque que la fonction d'auto-corrélation a(u,v) n'est non nulle que si (u,v) représente un 'vecteur' reliant deux impacts existants: $a(u,v)\neq 0 \quad \Leftrightarrow \quad \exists\ i,j \quad (x_i-x_j, y_i-y_j) = (u,v)$
(Si je glisse l'image sur elle même d'un vecteur (u,v), au moins deux impacts se superposent)

La valeur prise par l'auto-corrélation en un point (u,v) est égal au nombre de redondance du vecteur (u,v) dans le motif des (xi,yi).
L'auto-corrélation d'une image de la caméra à comptage de photons peut donc être vu comme étant l'histogramme de présence des vecteurs (u,v) dans le motif formé par les impacts (xi, yi).
Notons que la valeur en (0,0) est a(0,0)=n où n est le nombre d'impacts.



L'apparition du 'trou de photon' au centre de l'auto-corrélation signifie qu'il n'existe pas dans l'image de départ de points d'impacts plus proches que la largeur du trou de photon. Cette limitation de la résolution de la caméra est due au phénomène de biais de photon qui interdit la distinction entre deux impacts proches. Ce défaut dans l'auto-corrélation des images de la caméra à comptage de photon doit absolument être corrigé au moment du traitement des données. Nous verrons quelle est l'influence de ce 'trou de photons' sur le module carré du spectre de l'image utilisé pour extraire la visibilité (c'est à dire sur la TF de l'autocorrélation)

4) Correction du trou de photons.

Depuis les années 70 et le développement des caméras à comptage de photons, plusieurs méthodes ont été proposées pour compenser le trou de photon: calibration de la caméra dans des conditions proches des conditions d'observation (afin d'obtenir un trou de photon de référence à soustraire aux données), division optique du flux de photon afin de le 'répartir' sur deux caméras à comptage de photon puis corrélation croisées des images (Thiébaut 1994), calcul analytique d'une estimation du trou de photon (Perrin 2002), méthode de redressement du signal par ajustement d'une fonction sur le bruit de fond du spectre (P. Berio 1997).

Dans le cas du prototype Carlina, nous cherchons à extraire des modules carrés de visibilités à partir des images biaisés en travaillant sur les spectres de ces images. On ne s'intéresse donc qu'aux conséquences du 'trou de photons' à l'endroit des pics du spectre (On ne cherche pas à modéliser et à combler l'effet du biais de photon sur tout le spectre).

Effet du trou de photon sur le spectre:
On rappelle que le module carré de la transformée de Fourier de l'image (ce qu'on appelle le spectre de puissance) est égal à la transformée de Fourier de l'autocorrélation de l'image (thérorème de Wiener-Khintchine. On peut donc visualiser l'effet du trou de photon sur le spectre en calculant sa transformée de Fourier. Pour simuler le trou de photon, je choisis une fonction gaussienne négative dont le sommet est tronqué à 0 (le centre de l'autocorrélation n'est pas affecté par le biais de photon et vaut n= le nombre d'impacts) et appliquée uniquement sur une couronne de rayon quelques pixels (Fig 19). on calcule ensuite sa transformée de Fourier (Fig 20)

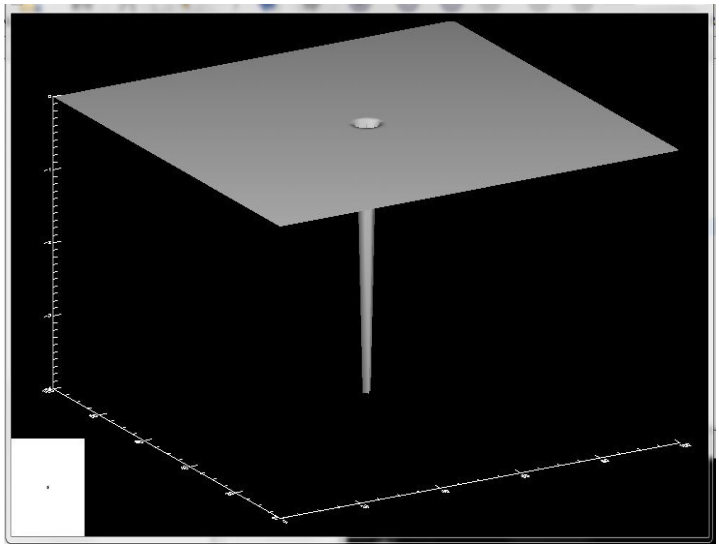
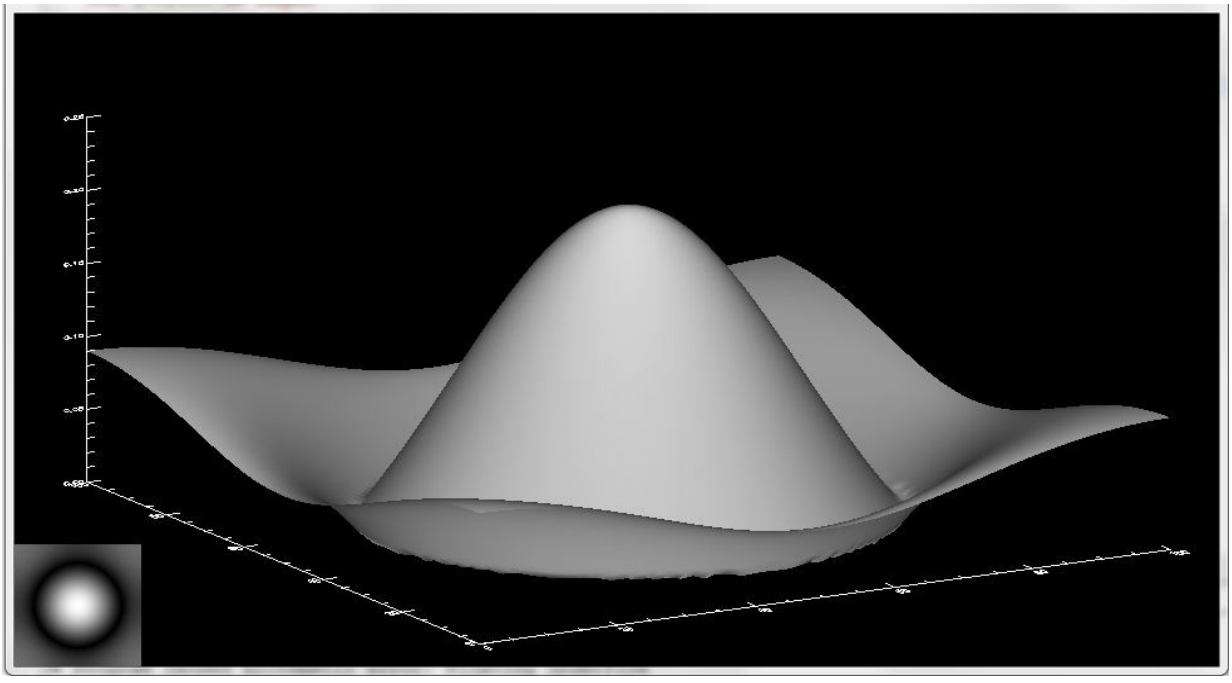


Fig 19 (ci contre): modélisation du trou de photon par une fonction gaussienne négative

Fig 20 (ci dessous): transformée de Fourier du trou de photon. On obtient une enveloppe gaussienne. En bas à gauche on visualise l'image obtenue.



→ le trou de photon dans l'autocorrélation rajoute une forme gaussienne dans le spectre de l'image qu'il faut soustraire aux valeur de nos pics! (Fig 21)

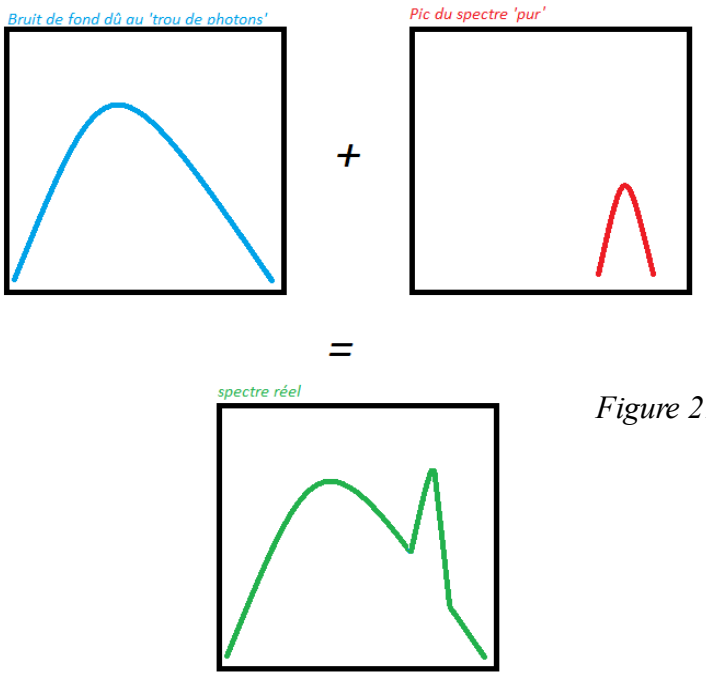


Figure 21

Méthode de correction du trou de photons:
Nous utilisons une variante de la méthode préconisé par P. Berio (A&A 1997). Pour chaque pic du spectre de l'image nous soustrayons le fond du biais de photon en interpolant un polynôme à deux dimensions appuyé sur une couronne entourant le pic de visibilité (Fig 22):

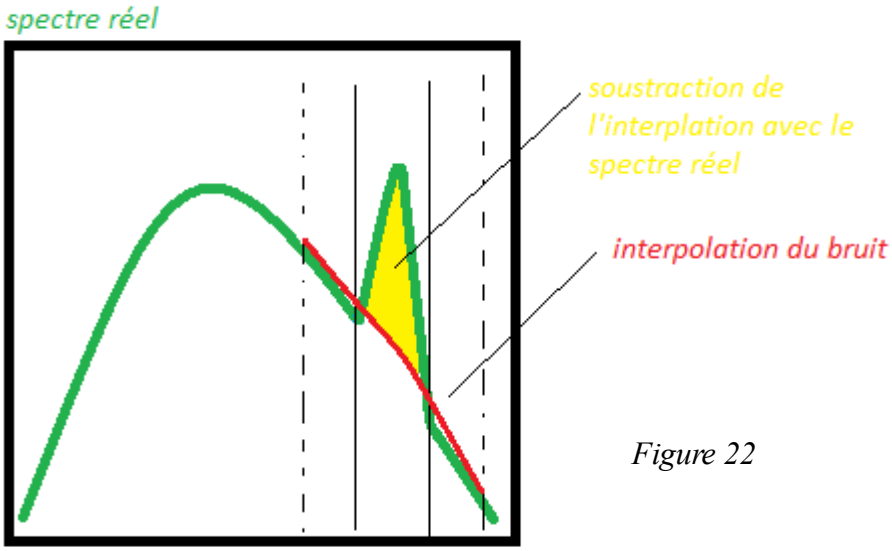
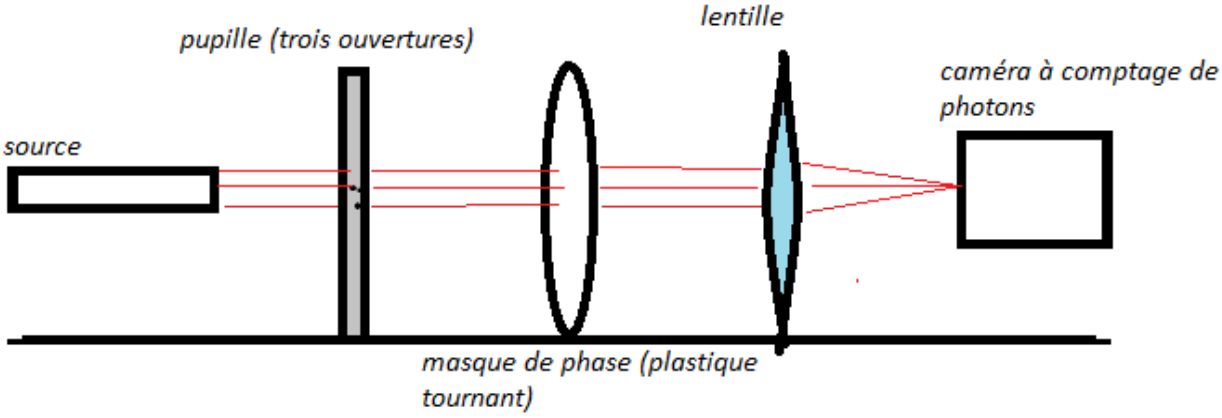


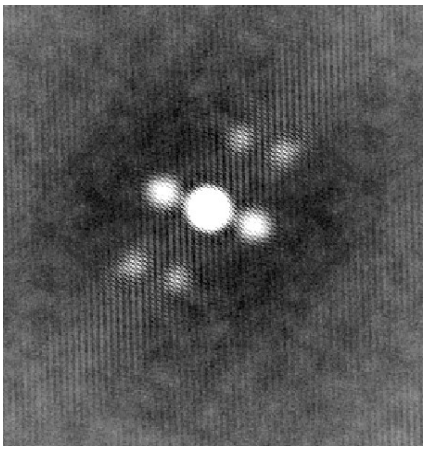
Figure 22

5) Traitement complet des images.

Figure 23



Je présente ici le traitement complet d'une série d'images obtenues en laboratoire avec un montage (Fig 23) simulant les images susceptibles d'être obtenues avec Carlina.
Un masque de phase tournant (un papier plastique) simule les turbulences atmosphériques. La caméra à comptage de photon enregistre les images par série de 1000 (1000 images = environ 10 secondes) avec des temps de pose d'une milliseconde.
Avec un premier programme (code en annexe), je calcule pour chaque image le module carré de la transformée de Fourier puis je somme sur les 1000 images. On obtient un spectre pour chaque série de 1000 images.



Ci dessus à gauche: capture d'écran du programme de calcul du spectre. La liste de traitement contient dans l'exemple 3 répertoires de 1000 images ('test_03', 'test_04' et 'test_05).

Ci dessus à droite: image du spectre obtenu pour 1000 images. On remarque les 6 pics secondaires correspondant aux 3 ouvertures de la pupille. Le calcul du module des visibilités au carré s'effectue en faisant le rapport des pics secondaires sur le pic principal multiplié par 3^2=9

Une fois les spectres obtenus, nous devons en extraire le module des visibilités complexes au carré en effectuant les rapports 9*pic secondaire / pic principal comme il a été démontré dans la partie II. Il faut également s'affranchir du biais de photon en interpolant un polynôme à deux dimensions 'sous' chaque pic (cf Fig 22). Pour cette opération, j'ai développé un logiciel (code en annexe)

interactif de traitement des spectres.

Sélection des pics:
La première étape consiste à sélectionner et à délimiter les 4 pics (le pic central et les trois secondaires) ainsi que les points qui serviront à ajuster les polynômes. Avec la souris, on définie deux zones pour chaque pic: un cercle plein délimitant le pic, et un cercle pointillé délimitant la zone sur laquelle 's'appuiera' le polynôme (Fig 24)

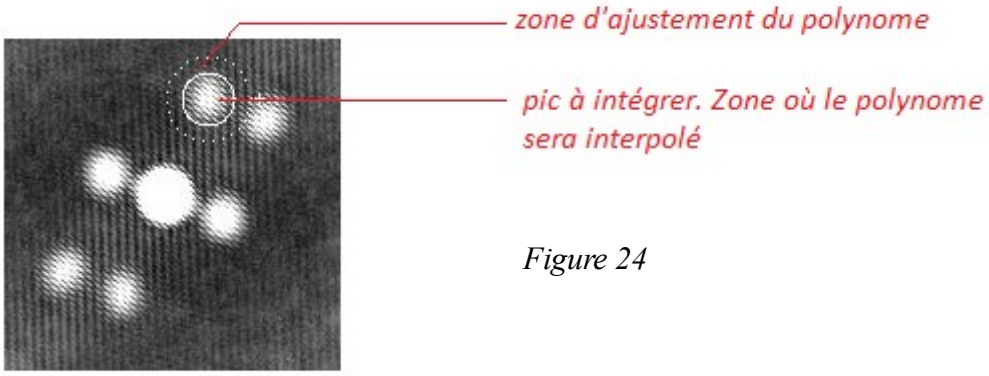


Figure 24

Les pics du spectre étant parfois très proches, il faut rejeter les points contenus dans le cercle en pointillé mais appartenant à un pic voisin (il ne doivent pas être pris en compte dans l'ajustement du polynôme). Tout point contenu à la fois dans un cercle pointillé et dans un cercle plein ne sera pas pris en compte au moment de l'ajustage des polynômes (Fig 25)

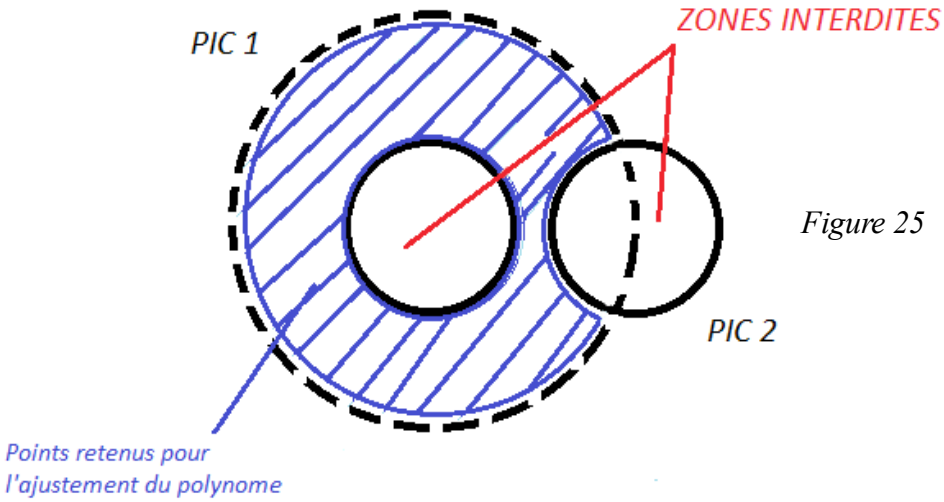
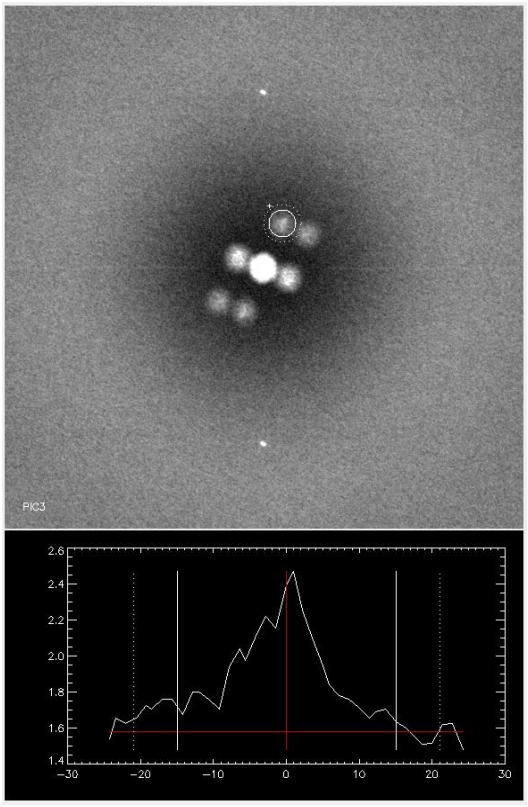


Figure 25

L'interpolation:
Pour chaque pic, on effectue les opérations suivantes:

- Ajustement d'un polynôme à deux dimensions dans la zone pointillée (moins les points appartenant aux zones interdites).
- Interpolation du polynôme obtenue 'sous' le pic (= à l'intérieur du cercle plein)
- Soustraction du polynôme au pic et intégration de la différence.

La procédure d'ajustement des polynômes utilisent des fonctions de 'fitage' d'IDL basées sur une méthode des moindres carrés. On utilise au choix la fonction 'sfit' ou 'mp2dfitfun'. À l'aide d'un curseur, on peut choisir le degré du polynôme et un écran permet de visualiser l'ajustement effectué. Expérimentalement, on remarque que les méthodes d'ajustement ont tendance à diverger pour des polynômes de degré supérieur à 3. De même, le calcul de la visibilité est affecté par le degré du polynôme choisi. Le résultat peut être sensiblement différent selon que l'on choisisse des polynômes de degré 0 où 1 où des polynômes de degré 2 où 3. Plus le degré est élevé, moins il est contraint par l'ajustement et plus il est libre de prendre une forme irrégulière 'sous' le pic. C'est à l'utilisateur de faire le choix qui lui semble le plus correct.
Ci dessous, 4 captures d'écran présentant la résolution d'un même pic avec des polynômes de degré différents. Le profil permet de visualiser l'ajustement de la fonction à l'image réelle.



Polynome de degré 0:

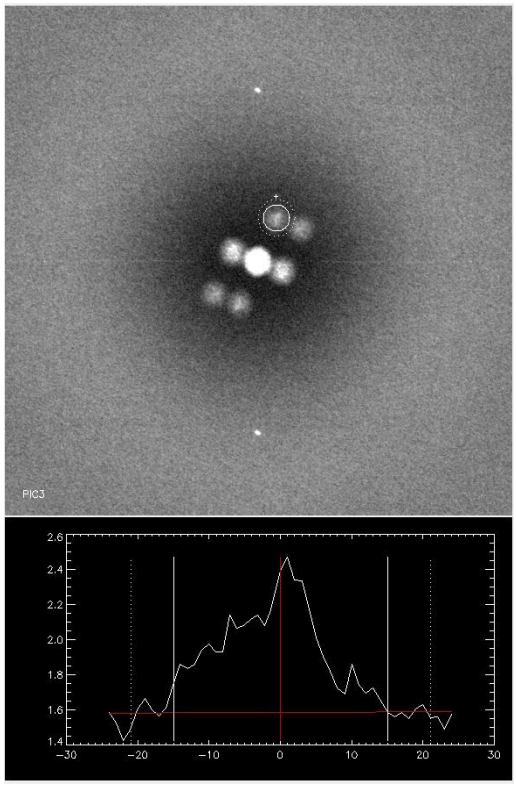
Khi carré = 0,0074

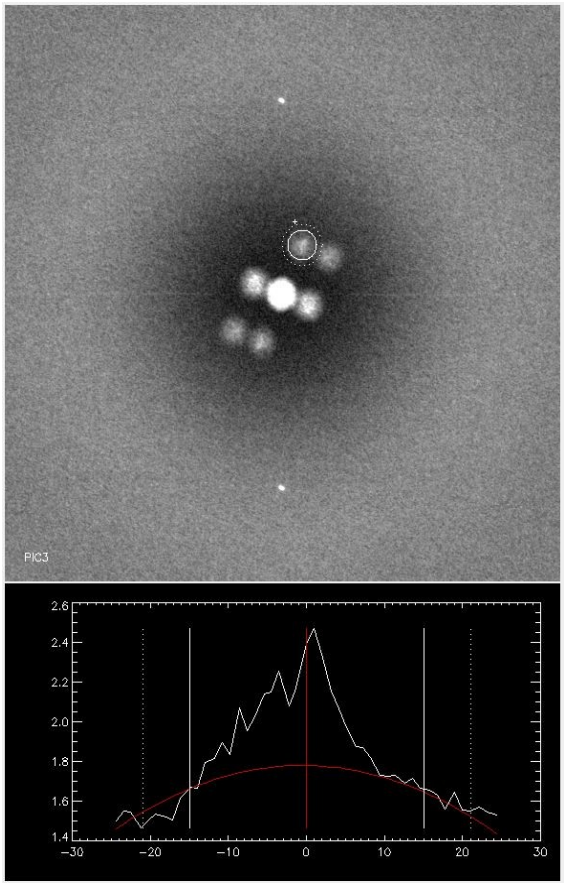
Module de la visibilité au carré = 0,152

Polynôme de degré 1:

Khi carré = 0,0062

Module de la visibilité au carré = 0,149





Polynôme de degré 2:

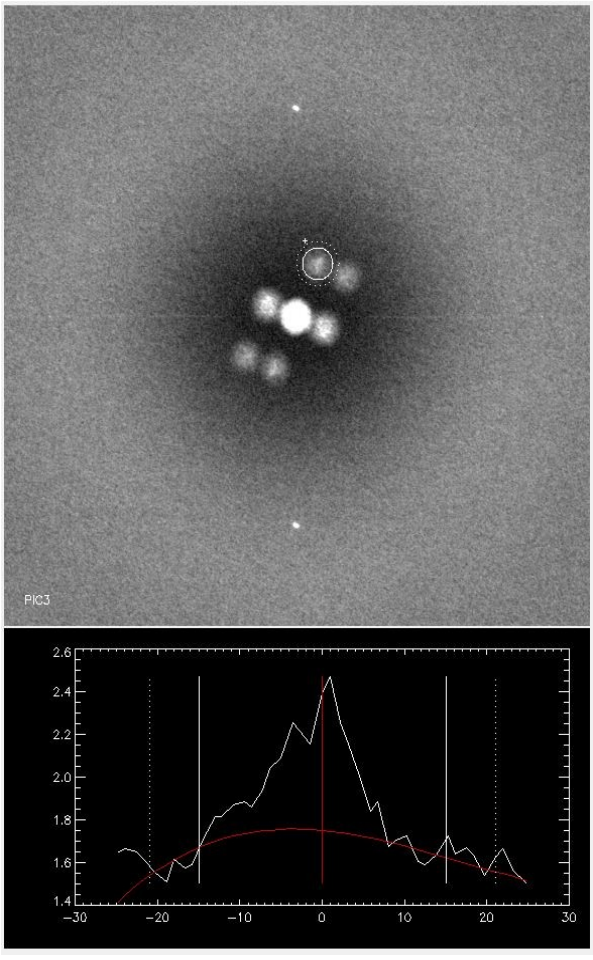
Khi carré = 0,0062

Module de la visibilité au carré = 0,083

Polynôme de degré 3:

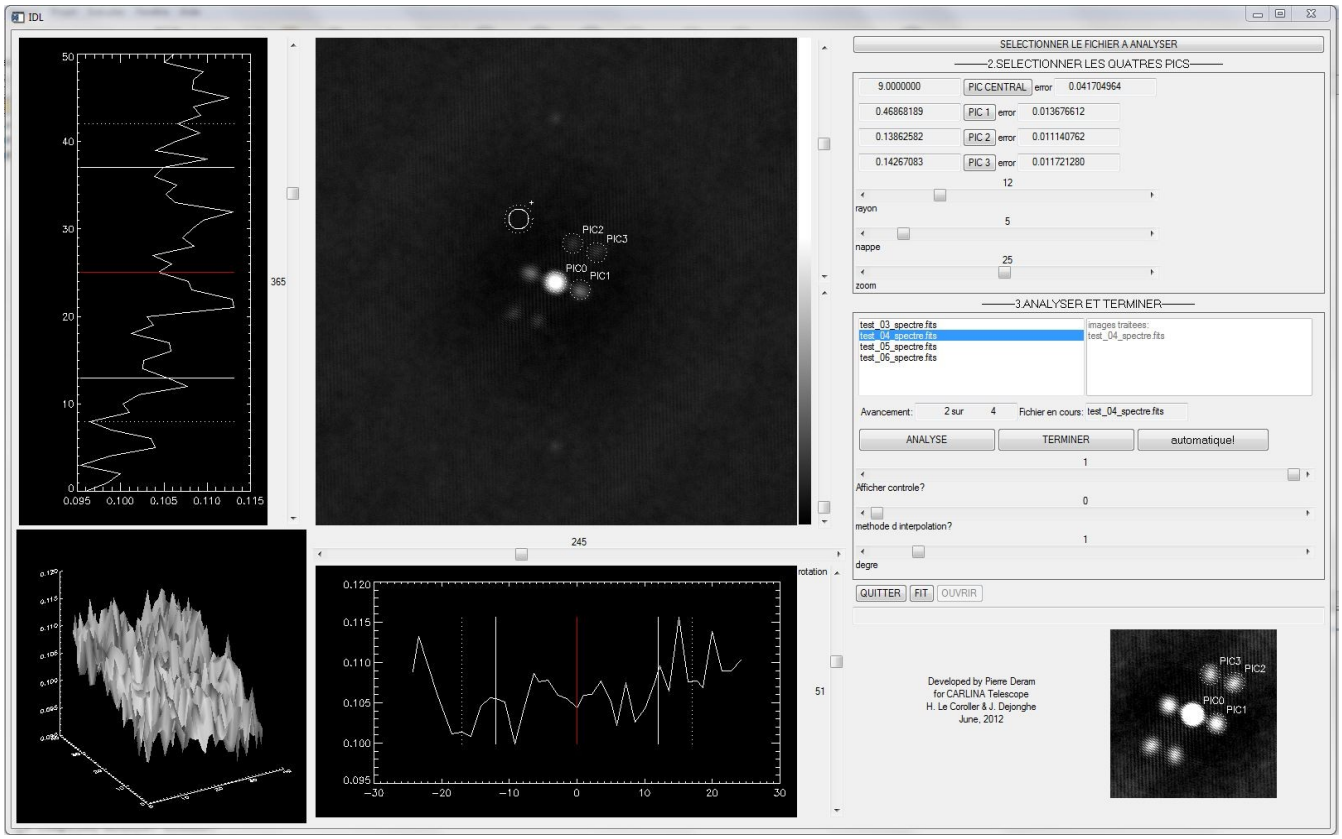
Khi carré = 0,0054

Module de la visibilité au carré: 0,091



Calcul de la visibilité:
Le logiciel interpole un polynôme en dessous de chaque pic puis calcule le module au carré des visibilités associées en effectuant le rapport 9*pic secondaire/pic principal comme il a été montré dans la partie II. Le tout est enregistré dans un document texte. Le logiciel est conçu de telle sorte que l'on puisse traiter rapidement les spectres les uns après les autres.

Ci dessous: capture d'écran de l'interface graphique du programme d'extraction des visibilités.



Bibliographie:

Le Coroller et al. 2012,SPIE, The first diluted telescope ever built in the world

Coroller, H. L., Dejonghe, J., Arpesella, C., Vernet, D., and Labeyrie, A., "Tests with a carlina-type hypertelescope prototype," A&A 426, 721 (2004 Paper I).

Coroller, H. L., Dejonghe, J., Regal, X., Sottile, R., Mourard, D., Ricci, D., Lardiere, O., Vansuu, A. L.,Boer, M., Becker, M. D., Clausse, J.-M., Guillaume, C., and Meunier, J.-P., "Tests with a carlina-type diluted telescope. primary coherencing," A&A 539, 59 (2012 Paper II).

Berio P., Vakili F., Mourard D., Bonneau D., "Removing the photon-centroiding hole in optical stellar interferometry", A&A 129,609-615 (1998)

Labeyrie, A., "Resolved imaging of extra-solar planets with future 10-100 km optical interferometric arrays." A&A 118, 517 (1996)

Tallon, M., Tallon-Bosc, I., "the object-image relationship in Michelson stellar interferometry", A&A 253, 641-645 (1992)

Thiébaut, E., "Speckle interferometry with a photon-counting detector", A&A 284, 340-348 (1994)

Perrin, G., "Subtracting the photon noise bias from single-mode optical interferometer visibilities", A&A 398, 385-390 (2003)

ANNEXES:

Annexe I: Comment combiner la lumière en interférométrie? Calcul de différences de marche pour un réseau co-sphérique

La présentation précédente développe le principe du traitement des images en interférométrie. Elle ne répond pas à la question technique de savoir comment faire interférer la lumière derrière une pupille géante! Dans le cas d'une pupille plane (comme il a été présenté dans les pages et les simulations précédentes), on ne peut imaginer utiliser un écran à l'infini (la condition f>> B imposerait une distance immense) ni même une lentille convergente qui devrait avoir un diamètre recouvrant toute la surface de la pupille! Actuellement, les interféromètres utilisent des lignes à retards pour guider la lumière derrière les sous ouvertures et combiner les rayons de manière à obtenir des franges (Fig 12 en début de partie II). Nous ne reviendrons pas sur les difficultés techniques que représente un tel système. Carlina utilise le stigmatisme des miroirs sphériques pour égaliser les chemins optiques. Je propose ci-dessous un calcul direct de la figure d'interférence obtenue le long de la demi sphère focale par calcul des différences de marche.

L'équation de la pupille à N ouvertures est:

$$P(\theta,\phi)=\delta(\theta-\theta_k;\phi-\phi_k).e^{i\phi_k}$$

où thêta et phi désigne les coordonnées sphériques calculées depuis le centre C du miroir primaire. Il est sous entendu que l'on se déplace le long de la demi sphère de rayon R, rayon du miroir primaire.

Un point M de la demi sphère focale sera lui aussi repéré par ses coordonnées thêta et phi. Il est sous entendu que l'on se déplace le long de la demi sphère focale de rayon F=R/2.

On cherche à calculer l'équation des franges en un point M de la demi-sphère focale:
Les rayons provenant d'une direction E fixé sont réfléchis sur les miroir M1, M2,..., Mk, puis interfèrent en M. L'amplitude complexe en M est donc:

$$E(M)=\sum_{k=1}^{N} e^{i\phi_k+i\frac{2\pi}{\lambda}.|EM_kM|}$$

L'équation des franges est donnée par le module carré de cette amplitude:

$$I(M)=E\,\overline{E}=N+\sum_{(k,k')}^{N(N-1)/2} 2\cos\left(\frac{2\pi}{\lambda}\left(|EM_kM|-|EM_{k'}M|\right)+\phi_k-\phi_{k'}\right)$$

Il nous faut donc calculer les différences de marche entre les rayons issus de miroirs deux à deux différents.
L'astuce consiste à utiliser le stigmatisme considéré comme parfait (corrigé en réalité par le correcteur de Mertz) du miroir sphérique décrit par le réseau de pupille: pour l'étoile source E à l'infini, il existe un point de la demi sphère focale où les rayons réfléchis convergent à égalité des chemins optiques. C'est le point où se forme en pupille pleine l'image de l'étoile. On le note O.
Inversement, si on s'intéresse à un point M quelconque de la sphère focale, il existe une direction E'

du ciel conjuguée. Pour tout point M, on peut donc créer une 'étoile fictive' pointée par le télescope. On la note E'.
Finalement, nous avons donc les deux relations suivantes pour tout miroir k et k' (Fig 14):

$$|EM_kO|=|EM_{k'}O| \quad et \quad |E'M_kM|=|E'M_{k'}M|$$

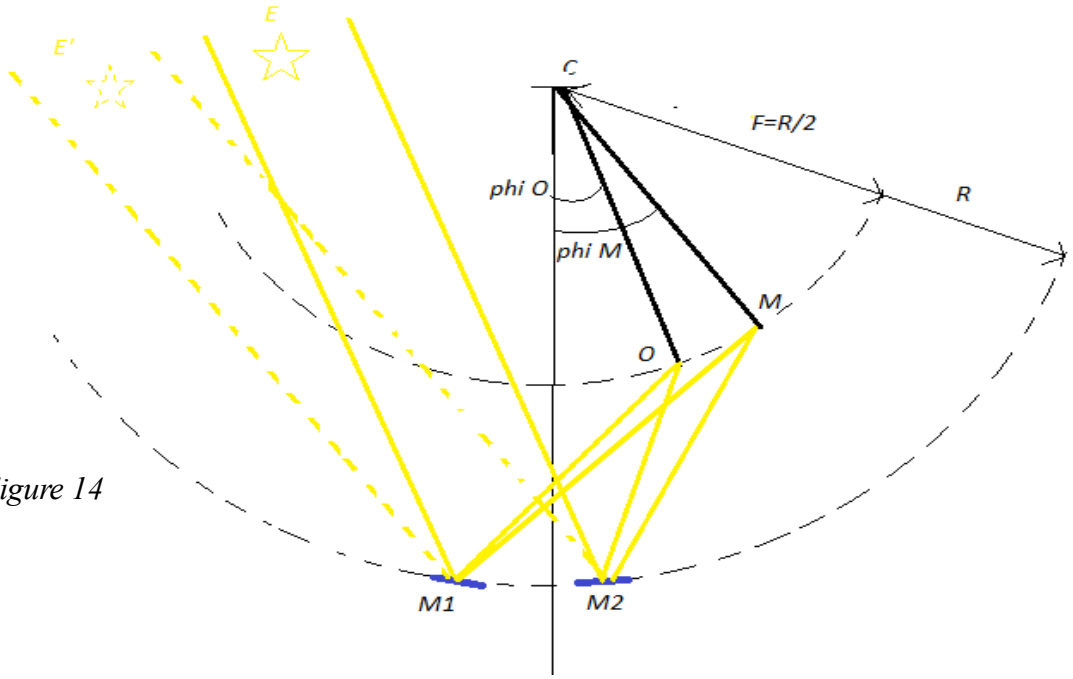


Figure 14

En décomposant les trajets optiques, les deux relations se transforment en:

$$|EM_k|+|M_kO|=|EM_{k'}|+|M_{k'}O| \quad et \quad |E'M_k|+|M_kM|=|E'M_{k'}|+|M_{k'}M|$$

En sommant les deux relations il vient directement:

$$|EM_kM|-|EM_{k'}M|=(|E'M_{k'}|-|E'M_k|)+(|M_{k'}O|-|M_kO|)$$

ou l'on peut remplacer la dernière parenthèse en utilisant la première relation; on a alors:

$$|EM_kM|-|EM_{k'}M|=(|E'M_{k'}|-|E'M_k|)+(|EM_k|-|EM_{k'}|)$$

Le calcul de la différence de marche est considérablement simplifié! Les termes entre parenthèse étant des différences de marche usuelles. Il suffit ensuite de remarquer que la construction des trajets optiques implique que la direction CM pointe vers E' et que la direction CO pointe vers E. Il vient alors directement:

$$|E'M_{k'}|-|E'M_k|=\frac{M_k\vec{M}_{k'}.C\vec{M}}{|C\vec{M}|} \quad et \quad |EM_k|-|EM_{k'}|=\frac{-M_k\vec{M}_{k'}.C\vec{O}}{|C\vec{O}|}$$

d'où le résultat final en réinjectant ces relations dans l'expression (*) :

$$|EM_k M| - |EM_{k'} M| = \frac{M_k \vec{M}_{k'} . \vec{OM}}{F}$$

où F=R/2 est le rayon focal du miroir sphérique mais aussi la distance séparant la pupille au sol de la surface focale (='l'écran') . On a donc **l'expression des franges pour M appartenant à la demi-sphère focale**:

$$I(M) = E\,\overline{E} = N + \sum_{(k,k')}^{N(N-1)/2} 2\cos\left(\frac{2\pi}{\lambda F}(M_k \vec{M}_{k'} . \vec{OM}) + \phi_{k-k'}\right)$$

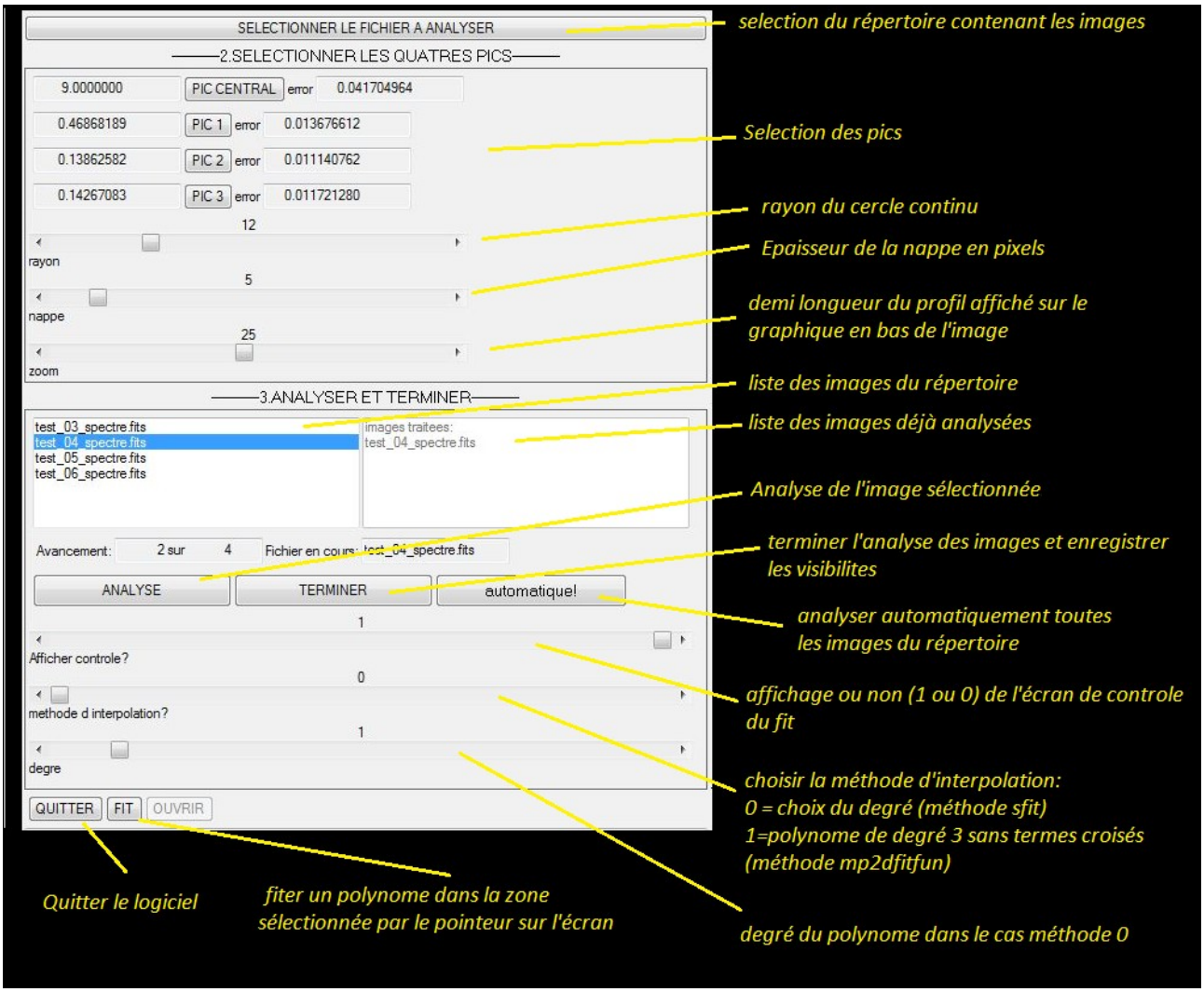
remarque: Si F devient très grand, la sphéricité s'atténue et la pupille et le plan focal deviennent localement 'plats'. On converge exactement vers l'expression trouvée dans la partie II dans le cas d'une pupille plane et d'un écran placé à une distance F très grande!

Annexe II: Code du programme de création du spectre.

```
pro widget1_event, ev
widget_control, ev.id, get_uvalue=uval
widget_control,ev.top, get_uvalue=u
case uval of
'u-avancement':
'u-list':
'u-etat':
'u-button':begin
result=dialog_pickfile(/directory)
cd, result
result1=file_search('*', /test_directory)
widget_control,u.liste, set_value=result1
a=strarr(50)+'a'
a[0]=result1
u.list=a
widget_control, ev.top, set_uvalue=u
end
'u-lancer':begin
widget_control, u.charge2,get_value=c2
b=u.list
a=where(b ne 'a',Count)
print, Count
result=u.list[0:Count-1]
n=size(result)
print, n
print, result
n=n[1]
u1=1/float(n)
for i=0, n-1 do begin
cd, result[i]
lc2=fltarr(501,10)+1
widget_control, u.etat, set_value=result[i]
widget_control, u.avancement, set_value=string(i)+' sur '+string(n)
a=file_search('*.fits')
dim=size(a)
nbre=dim[1]
u2=1/float(nbre)
c=size(readfits(a(1)))
c=c[1]
totale=fltarr(c,c)
t=fltarr(c,c)
for j=0, nbre-1 do begin
image=readfits(a(j))
four=c*fft(image, -1)
four=abs(four)^2
t=t+four
lc2[long(500*float((j+1)*u2)),0:9]=0
tvscl, lc2
endfor
cd,'..'
totale=fft(t,+1)
t=retourne(t)
totale=retourne(totale)
t=abs(t)
totale=abs(totale)
writefits, result[i]+'_spectre.fits', t
totale=rotate(totale,1)
totale=rotate(totale,5)
writefits, result[i]+'_xxx-autocorrel.fits', totale
```

```
totale(where(totale eq max(totale)))=0
writefits,result[i]+'_autocorrel.fits',totale
endfor
end
endcase
end
pro widget1
u={liste:'',charge2:'',list:strarr(50),etat:'', avancement:''}
base=widget_base(uvalue=u,/column)
bouton=widget_button(base,value='Selectionner le repertoire parent', uvalue='u-
button')
liste=widget_list(base, value='', uvalue='u-list', xsize=50,ysize=10,units=0)
lancer=widget_button(base, value='>>>lancez>>>', uvalue='u-lancer')
base1=widget_base(base, /row)
text1=widget_label(base1, value=' Avancement:')
avancement=widget_text(base1, value='', uvalue='u-avancement')
text0=widget_label(base1, value='Fichier en cours:')
etat=widget_text(base1, value='', uvalue='u-etat')
charge2=widget_draw(base, xsize=500,ysize=10)
u.liste=liste
u.etat=etat
u.avancement=avancement
u.charge2=charge2
u.list=strarr(50)+'a'
widget_control, base, set_uvalue=u
widget_control, base, /realize
xmanager, 'widget1', base
end
```

Annexe III: capture d'écran de l'écran de contrôle du logiciel d'extraction des visibilités:



Annexe IV: Code du programme d'extraction des visibilités.

```
pro widget3_event, ev
widget_control, ev.id, get_uvalue=uval
widget_control,ev.top, get_uvalue=u
case uval of
'u-terminer':begin
openw,lun, 'test.txt', /get_lun,/append
printf, lun, '+FIN+'
close,lun
free_lun,lun
liretxt,'test.txt'
widget_control, u.liste,set_value=''
widget_control, u.liste1,set_value='images traitees'
widget_control, u.analyse,sensitive=0
widget_control, u.lancer,sensitive=0
widget_control, u.terminer, sensitive=0
end
'u-controle':begin
end
'u-echelle':begin
widget_control, u.bas,get_value=bas
widget_control, u.haut,get_value=haut
widget_control, u.couleur, get_value=index
vec=findgen(580)
vec=vec<(haut*580/100.)
vec=vec>(bas*580/100.)
vec=rebin(transpose(vec),15,580)
wset, index
tvscl, vec
a=u.image
b=alog(a)
d=max(b)-min(b)
min=min(b)+bas*d/100.
max=min(b)+haut*d/100.
b=b>min
b=b<max
u.log=b
widget_control,ev.top, set_uvalue=u
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
widget_control, u.n,get_value=n
widget_control, u.slide1,get_value=x
widget_control, u.slide2,get_value=y
widget_control, u.graph0, get_value=index
widget_control,u.zoom,get_value=zoom
wset, index
tvscl, u.log
if u.pic0.et eq 1 then begin
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
endif
if u.pic1.et eq 1 then begin
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device     ;
endif
if u.pic2.et eq 1 then begin
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
endif
if u.pic3.et eq 1 then begin
```

```
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
endif
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
end
'u-fitage':begin
widget_control, u.fitage, get_value=fitage
if fitage eq 1 then widget_control, u.d,sensitive=0
if fitage eq 0 then widget_control, u.d,sensitive=1
end
'u-slide':begin
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
widget_control, u.n,get_value=n
widget_control, u.slide1,get_value=x
widget_control, u.slide2,get_value=y
widget_control, u.graph0, get_value=index
widget_control, u.zoom, get_value=zoom
wset, index
tvscl, u.log
if u.pic0.et eq 1 then begin
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
endif
if u.pic1.et eq 1 then begin
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device
endif
if u.pic2.et eq 1 then begin
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
endif
if u.pic3.et eq 1 then begin
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
endif
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
x1=2*x-x2
y1=2*y-y2
dy=y2-y1
dx=x2-x1
d=abs(dx)>abs(dy)
a=fltarr(d+1)
b=fltarr(d+1)
dy=dy/float(d)
dx=dx/float(d)
for i=0,d do begin
a[i]=x1+long(i*dx)
b[i]=y1+long(i*dy)
endfor
c=u.image(a,b)
widget_control, u.graph1, get_value=index
wset, index
```

```
sig=fltarr(d+1)
sig=1*(abs(a-x2) le abs(a-x1))*(abs(b-y2) le abs(b-y1))
sig(where(sig eq 0))=-1
coord=sig*sqrt((a-x)^2+(b-y)^2)
plot,coord,c,/ynozero,xrange=[-zoom,zoom]
oplot,[0,0],[min(c),max(c)],color=200
oplot,[-r,-r],[min(c),max(c)]
oplot,[r,r],[min(c),max(c)]
oplot,[-r-n,-r-n],[min(c),max(c)],/linestyle
oplot,[+r+n,+r+n],[min(c),max(c)],/linestyle
widget_control, u.graph2, get_value=index
wset, index
c=u.image[x,y-25:y+25]
plot, c, findgen(51)
m=max(c)
mi=min(c)
oplot,[m,mi],[25-r,25-r]
oplot, [m,mi],[25-r-n,25-r-n], linestyle=1
oplot,[m,mi],[25,25], color=200
oplot, [m,mi],[25+r+n, 25+r+n], linestyle=1
oplot,[m,mi],[25+r, 25+r]
c=u.image[x-r-n:x+r+n,y-r-n:y+r+n]
widget_control, u.pic1.draw1, get_value=index
wset,index
shade_surf,c
end
'u-lancer':begin
l=u.list
for i=0, u.taille-1 do begin
adresse=u.list[i]
widget_control, u.etat, set_value=u.list[i]
widget_control, u.avancement, set_value=string(i+1)+' sur'+string(u.taille)
im=readfits(adresse,header)
dim=size(header)
dim=dim[1]
date=''
for j=0,dim-1 do begin
if strpos(header[j],'DATE') ne -1 then date=header[j]
endfor
dim=size(im)
dim=dim[1]
n=(dim-581)/2
image=fltarr(581,581)
image[0,0]=im[n+1:n+581,n+1:n+581]
widget_control, u.bas,get_value=bas
widget_control, u.haut,get_value=haut
widget_control, u.couleur, get_value=index
vec=findgen(580)
vec=vec<(haut*580/100.)
vec=vec>(bas*580/100.)
vec=rebin(transpose(vec),15,580)
wset, index
tvscl, vec
b=alog(image)
d=max(b)-min(b)
min=min(b)+bas*d/100.
max=min(b)+haut*d/100.
b=b>min
b=b<max
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
```

```
widget_control, u.n,get_value=n
widget_control, u.slide1,get_value=x
widget_control, u.slide2,get_value=y
widget_control, u.graph0, get_value=index
widget_control, u.zoom,get_value=zoom
wset, index
tvscl, b
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
widget_control, u.fitage,get_value=fitage
tt=[u.pic0,u.pic1,u.pic2,u.pic3]
for o=0,3 do begin
x=tt[o].x
y=tt[o].y
r=tt[o].ray
n=tt[o].nap
widget_control, u.graph0, get_value=index
widget_control, u.d, get_value=g
wset, index
tab0=circle(x,y,r)
tab1=circle(x,y,r+n)
result0=polyfillv(tab0[0,*],tab0[1,*],581,581)
result1=polyfillv(tab1[0,*],tab1[1,*],581,581)
x=result1 mod 581
y=long(result1/581)
dim1=size(result1)
n=dim1[1]
a=[]
for j=0,n-1 do begin
r0=(x(j)-u.pic0.x)^2+(y(j)-u.pic0.y)^2
r0=sqrt(r0)
r1=(x(j)-u.pic1.x)^2+(y(j)-u.pic1.y)^2
r1=sqrt(r1)
r2=(x(j)-u.pic2.x)^2+(y(j)-u.pic2.y)^2
r2=sqrt(r2)
r3=(x(j)-u.pic3.x)^2+(y(j)-u.pic3.y)^2
r3=sqrt(r3)
test=(r0 gt u.pic0.ray)*(r1 gt u.pic1.ray)*(r2 gt u.pic2.ray)*(r3 gt u.pic3.ray)
if test eq 1 then a=[a,result1(j)]
endfor
d=a mod 581
e=long(a/581.)
f=image[d,e]
if fitage eq 1 then begin
n=size(f)
n=n[1]
start=fltarr(7)
err=fltarr(n)+1
p=mpfit2dfun('myfunct',d,e,f,err,start)
image(result1)=image(result1)-myfunct(result1 mod 581,long(result1/581),p)
```

```
endif
if fitage eq 0 then begin
tab=transpose([[d],[e],[f]])
extra=sfit(tab,g,/irregular,/max_degree,kx=p)
image(result1)=image(result1)-horner2dbis(result1 mod 581,long(result1/581),p)
endif
if o eq 1 then p1=p
if o eq 2 then p2=p
if o eq 3 then p3=p
a=image(result0)
b=image(result1)
m=size(result0)
m=m[1]
n=size(result1)
n=n[1]
kicarre=(b)^2
kicarre=total(kicarre)
d=(a)^2
kicarre=kicarre-total(d)
tt[o].error=kicarre/float(n-m)
tt[o].flux=total(a)
tt[o].vis=9*tt[o].flux/float(tt[0].flux)
endfor
widget_control, u.text0, set_value=string(tt[0].vis)
widget_control, u.text1, set_value=string(tt[1].vis)
widget_control, u.text2, set_value=string(tt[2].vis)
widget_control, u.text3, set_value=string(tt[3].vis)
widget_control, u.error0, set_value=string(tt[0].error)
widget_control, u.error1, set_value=string(tt[1].error)
widget_control, u.error2, set_value=string(tt[2].error)
widget_control, u.error3, set_value=string(tt[3].error)
openw,lun, 'test-auto.txt', /get_lun,/append
printf, lun, adresse
printf,lun,date
printf,lun,'polynome de degre: ', g
printf, lun,'visibilite1', tt[1].vis
printf,lun,'visibilite2', tt[2].vis
printf,lun,'visibilite3', tt[3].vis
printf,lun,'---'
close, lun
free_lun,lun
endfor
openw,lun,'test-auto.txt',/get_lun,/append
printf,lun,'+FIN+'
close,lun
free_lun,lun
liretxt,'test-auto.txt'
widget_control,u.liste,set_value=''
widget_control, u.liste1,set_value='images traitees'
widget_control,u.lancer,sensitive=0
widget_control,u.terminer,sensitive=0
widget_control,u.analyse,sensitive=0
end
'u-repertoire':begin
widget_control, u.memo,get_value=index
wset,index
tvscl, readfits('C:\Users\DERAM\Documents\ARTIFICIEL\image.fits')
widget_control, u.liste1,set_value='images traitees'
result=dialog_pickfile(/directory)
cd, result
result1=file_search('*spectre.fits')
```

```
widget_control,u.liste, set_value=result1
dim=size(result1)
dim=dim[1]
u.taille=dim
u.i=0
a=strarr(50,2)
a[*,0]='a'
a[*,1]='n'
a[0,0]=result1
u.list=a
widget_control, ev.top, set_uvalue=u
end
'u-list':begin
c=ev.index
u.i=c+1
u.adresse=u.list[c,0]
widget_control, u.etat, set_value=u.list[c,0]
widget_control, u.avancement, set_value=string(c+1)+'sur'+string(u.taille)
im=readfits(u.list[c,0],header)
dim=size(header)
dim=dim[1]
for i=0,dim-1 do begin
if strpos(header[i],'DATE') ne -1 then u.date=header[i]
endfor
dim=size(im)
dim=dim[1]
n=(dim-581)/2
b=fltarr(581,581)
b[0,0]=im[n+1:n+581,n+1:n+581]
u.image=b
widget_control, u.bas,get_value=bas
widget_control, u.haut,get_value=haut
widget_control, u.couleur, get_value=index
vec=findgen(580)
vec=vec<(haut*580/100.)
vec=vec>(bas*580/100.)
vec=rebin(transpose(vec),15,580)
wset, index
tvscl, vec
a=u.image
b=alog(a)
d=max(b)-min(b)
min=min(b)+bas*d/100.
max=min(b)+haut*d/100.
b=b>min
b=b<max
u.log=b
widget_control,ev.top, set_uvalue=u
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
widget_control, u.n,get_value=n
widget_control, u.slide1,get_value=x
widget_control, u.slide2,get_value=y
widget_control, ev.top,set_uvalue=u
widget_control, u.graph0, get_value=index
widget_control, u.zoom,get_value=zoom
wset, index
tvscl, u.log
if u.pic0.et eq 1 then begin
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
```

```
endif
if u.pic1.et eq 1 then begin
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device
endif
if u.pic2.et eq 1 then begin
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
endif
if u.pic3.et eq 1 then begin
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
endif
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
x1=2*x-x2
y1=2*y-y2
dy=y2-y1
dx=x2-x1
d=abs(dx)>abs(dy)
a=fltarr(d+1)
b=fltarr(d+1)
dy=dy/float(d)
dx=dx/float(d)
for i=0,d do begin
a[i]=x1+long(i*dx)
b[i]=y1+long(i*dy)
endfor
c=u.image(a,b)
widget_control, u.graph1, get_value=index
wset, index
sig=fltarr(d+1)
sig=1*(abs(a-x2) le abs(a-x1))*(abs(b-y2) le abs(b-y1))
sig(where(sig eq 0))=-1
coord=sig*sqrt((a-x)^2+(b-y)^2)
plot,coord,c,/ynozero,xrange=[-zoom,zoom]
oplot,[0,0],[min(c),max(c)],color=200
oplot,[-r,-r],[min(c),max(c)]
oplot,[r,r],[min(c),max(c)]
oplot,[-r-n,-r-n],[min(c),max(c)],/linestyle
oplot,[+r+n,+r+n],[min(c),max(c)],/linestyle
widget_control, u.graph2, get_value=index
wset, index
c=u.image[x,y-25:y+25]
plot, c, findgen(51)
m=max(c)
mi=min(c)
oplot,[m,mi],[25-r,25-r]
oplot, [m,mi],[25-r-n,25-r-n], linestyle=1
oplot,[m,mi],[25,25], color=200
oplot, [m,mi],[25+r+n, 25+r+n], linestyle=1
oplot,[m,mi],[25+r, 25+r]
c=u.image[x-r-n:x+r+n,y-r-n:y+r+n]
widget_control, u.pic1.draw1, get_value=index
wset,index
shade_surf,c
end
'u-quitter':begin
```

```
widget_control,ev.top,/destroy
end
'u-image':begin
t=u.t
if ev.type eq 1 then t=-t
u.t=t
widget_control, ev.top, set_uvalue=u
x=ev.x
y=ev.y
test=(t eq 1)*(x lt 530)*(x gt 49)*(y lt 530)*(y gt 49)
if test eq 1 then begin
widget_control, u.slide1,set_value=x
widget_control, u.slide2,set_value=y
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
widget_control, u.n,get_value=n
widget_control, ev.top,set_uvalue=u
widget_control, u.graph0, get_value=index
widget_control, u.zoom,get_value=zoom
wset, index
tvscl, u.log
if u.pic0.et eq 1 then begin
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
endif
if u.pic1.et eq 1 then begin
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device
endif
if u.pic2.et eq 1 then begin
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
endif
if u.pic3.et eq 1 then begin
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
endif
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
x1=2*x-x2
y1=2*y-y2
dy=y2-y1
dx=x2-x1
d=abs(dx)>abs(dy)
a=fltarr(d+1)
b=fltarr(d+1)
dy=dy/float(d)
dx=dx/float(d)
for i=0,d do begin
a[i]=x1+long(i*dx)
b[i]=y1+long(i*dy)
endfor
c=u.image(a,b)
widget_control, u.graph1, get_value=index
wset, index
sig=fltarr(d+1)
sig=1*(abs(a-x2) le abs(a-x1))*(abs(b-y2) le abs(b-y1))
sig(where(sig eq 0))=-1
```

```
coord=sig*sqrt((a-x)^2+(b-y)^2)
plot,coord,c,/ynozero,xrange=[-zoom,zoom]
oplot,[0,0],[min(c),max(c)],color=200
oplot,[-r,-r],[min(c),max(c)]
oplot,[r,r],[min(c),max(c)]
oplot,[-r-n,-r-n],[min(c),max(c)],/linestyle
oplot,[+r+n,+r+n],[min(c),max(c)],/linestyle
widget_control, u.graph2, get_value=index
wset, index
c=u.image[x,y-25:y+25]
plot, c, findgen(51)
m=max(c)
mi=min(c)
oplot,[m,mi],[25-r,25-r]
oplot, [m,mi],[25-r-n,25-r-n], linestyle=1
oplot,[m,mi],[25,25], color=200
oplot, [m,mi],[25+r+n, 25+r+n], linestyle=1
oplot,[m,mi],[25+r, 25+r]
c=u.image[x-r-n:x+r+n,y-r-n:y+r+n]
widget_control, u.pic1.draw1, get_value=index
wset,index
shade_surf,c
endif
end
'u-degre':
'u-pic0':begin
widget_control, u.slide1, get_value=x
widget_control, u.slide2, get_value=y
widget_control, u.r, get_value=r
widget_control, u.n, get_value=n
u.pic0.x=x
u.pic0.y=y
u.pic0.ray=r
u.pic0.nap=n
u.pic0.et=1
widget_control, u.text0,set_value='ok...'
widget_control, ev.top, set_uvalue=u
if (u.pic0.et)*(u.pic1.et)*(u.pic2.et)*(u.pic3.et) eq 1 then begin
widget_control, u.analyse,/sensitive
widget_control, u.lancer,/sensitive
endif
end
'u-pic1':begin
widget_control, u.slide1, get_value=x
widget_control, u.slide2, get_value=y
widget_control, u.r, get_value=r
widget_control, u.n, get_value=n
u.pic1.x=x
u.pic1.y=y
u.pic1.ray=r
u.pic1.nap=n
u.pic1.et=1
widget_control, u.text1,set_value='ok...'
widget_control, ev.top, set_uvalue=u
if (u.pic0.et)*(u.pic1.et)*(u.pic2.et)*(u.pic3.et) eq 1 then begin
widget_control, u.analyse,/sensitive
widget_control, u.lancer,/sensitive
endif
end
'u-pic2':begin
widget_control, u.slide1, get_value=x
```

```
widget_control, u.slide2, get_value=y
widget_control, u.r, get_value=r
widget_control, u.n, get_value=n
u.pic2.x=x
u.pic2.y=y
u.pic2.ray=r
u.pic2.nap=n
u.pic2.et=1
widget_control, u.text2,set_value='ok...'
widget_control, ev.top, set_uvalue=u
if (u.pic0.et)*(u.pic1.et)*(u.pic2.et)*(u.pic3.et) eq 1 then begin
widget_control, u.analyse,/sensitive
widget_control, u.lancer,/sensitive
endif
end
'u-pic3':begin
widget_control, u.slide1, get_value=x
widget_control, u.slide2, get_value=y
widget_control, u.r, get_value=r
widget_control, u.n, get_value=n
u.pic3.x=x
u.pic3.y=y
u.pic3.ray=r
u.pic3.nap=n
u.pic3.et=1
widget_control, u.text3,set_value='ok...'
widget_control, ev.top, set_uvalue=u
if (u.pic0.et)*(u.pic1.et)*(u.pic2.et)*(u.pic3.et) eq 1 then begin
widget_control, u.analyse,/sensitive
widget_control, u.lancer,/sensitive
endif
end
'u-analyse':begin
widget_control, u.terminer, /sensitive
widget_control, u.fitage,get_value=fitage
tt=[u.pic0,u.pic1,u.pic2,u.pic3]
for o=0,3 do begin
x=tt[o].x
y=tt[o].y
r=tt[o].ray
n=tt[o].nap
image=u.image
widget_control, u.graph0, get_value=index
widget_control, u.d, get_value=g
wset, index
tab0=circle(x,y,r)
tab1=circle(x,y,r+n)
result0=polyfillv(tab0[0,*],tab0[1,*],581,581)
result1=polyfillv(tab1[0,*],tab1[1,*],581,581)
x=result1 mod 581
y=long(result1/581)
dim1=size(result1)
n=dim1[1]
a=[]
for i=0,n-1 do begin
r0=(x(i)-u.pic0.x)^2+(y(i)-u.pic0.y)^2
r0=sqrt(r0)
r1=(x(i)-u.pic1.x)^2+(y(i)-u.pic1.y)^2
r1=sqrt(r1)
r2=(x(i)-u.pic2.x)^2+(y(i)-u.pic2.y)^2
r2=sqrt(r2)
```

```
r3=(x(i)-u.pic3.x)^2+(y(i)-u.pic3.y)^2
r3=sqrt(r3)
test=(r0 gt u.pic0.ray)*(r1 gt u.pic1.ray)*(r2 gt u.pic2.ray)*(r3 gt u.pic3.ray)
if test eq 1 then a=[a,result1(i)]
endfor
d=a mod 581
e=long(a/581.)
f=u.image[d,e]
if fitage eq 1 then begin
n=size(f)
n=n[1]
start=fltarr(7)
err=fltarr(n)+1
p=mpfit2dfun('myfunct',d,e,f,err,start)
image(result1)=image(result1)-myfunct(result1 mod 581,long(result1/581),p)
endif
if fitage eq 0 then begin
tab=transpose([[d],[e],[f]])
extra=sfit(tab,g,/irregular,/max_degree,kx=p)
image(result1)=image(result1)-horner2dbis(result1 mod 581,long(result1/581),p)
endif
if o eq 1 then p1=p
if o eq 2 then p2=p
if o eq 3 then p3=p
a=image(result0)
b=image(result1)
m=size(result0)
m=m[1]
n=size(result1)
n=n[1]
kicarre=(b)^2
kicarre=total(kicarre)
d=(a)^2
kicarre=kicarre-total(d)
tt[o].error=kicarre/float(n-m)
tt[o].flux=total(a)
tt[o].vis=9*tt[o].flux/float(tt[0].flux)
endfor
widget_control, u.controle, get_value=controle
if controle eq 1 then widget4,tt,p1,p2,p3,u.image,u.log,fitage
widget_control, u.text0, set_value=string(tt[0].vis)
widget_control, u.text1, set_value=string(tt[1].vis)
widget_control, u.text2, set_value=string(tt[2].vis)
widget_control, u.text3, set_value=string(tt[3].vis)
widget_control, u.error0, set_value=string(tt[0].error)
widget_control, u.error1, set_value=string(tt[1].error)
widget_control, u.error2, set_value=string(tt[2].error)
widget_control, u.error3, set_value=string(tt[3].error)
c=u.i-1
test=0
if u.list[c,1] eq 'y' then test=1
u.list[c,1]='y'
b=u.list[where(u.list[*,1] eq 'y'),0]
widget_control, u.liste1,set_value=['images traitees:',b]
widget_control, ev.top, set_uvalue=u
if test eq 0 then begin
openw,lun, 'test.txt', /get_lun,/append
printf, lun, u.adresse
printf,lun,u.date
printf,lun,'polynome de degre: ', g
printf, lun,'visibilite1', tt[1].vis
```

```
printf,lun, 'visibilite2', tt[2].vis
printf,lun, 'visibilite3', tt[3].vis
printf,lun, '---'
close, lun
free_lun,lun
endif
if test eq 1 then begin
ligne=''
openu, lun, 'test.txt',/get_lun
readf, lun, ligne
while strpos(ligne,u.adresse) eq -1 do begin
readf,lun,ligne
endwhile
readf,lun,ligne
readf,lun,ligne
printf,lun,'visibilite1', tt[1].vis
printf,lun, 'visibilite2', tt[2].vis
printf,lun, 'visibilite3', tt[3].vis
printf,lun, '---'
close,lun
free_lun,lun
endif
end
'u-ouvrir':begin
fn=dialog_pickfile()
u.adresse=fn
im=readfits(fn,header)
dim=size(header)
dim=dim[1]
for i=0,dim-1 do begin
if strpos(header[i],'DATE') ne -1 then u.date=header[i]
endfor
dim=size(im)
dim=dim[1]
n=(dim-581)/2
b=fltarr(581,581)
b[0,0]=im[n+1:n+581,n+1:n+581]
u.image=b
widget_control, u.bas,get_value=bas
widget_control, u.haut,get_value=haut
widget_control, u.couleur, get_value=index
vec=findgen(580)
vec=vec<(haut*580/100.)
vec=vec>(bas*580/100.)
vec=rebin(transpose(vec),15,580)
wset, index
tvscl, vec
a=u.image
b=alog(a)
d=max(b)-min(b)
min=min(b)+bas*d/100.
max=min(b)+haut*d/100.
b=b>min
b=b<max
u.log=b
widget_control,ev.top, set_uvalue=u
widget_control, u.rotation, get_value=angle
widget_control, u.r,get_value=r
widget_control, u.n,get_value=n
widget_control, u.slide1,get_value=x
widget_control, u.slide2,get_value=y
```

```
widget_control, ev.top,set_uvalue=u
widget_control, u.graph0, get_value=index
widget_control, u.zoom,get_value=zoom
wset, index
tvscl, u.log
if u.pic0.et eq 1 then begin
plots, circle(u.pic0.x, u.pic0.y,u.pic0.ray), /device,/linestyle
xyouts,u.pic0.x+u.pic0.ray,u.pic0.y+u.pic0.ray, 'PIC0',/device
endif
if u.pic1.et eq 1 then begin
plots, circle(u.pic1.x, u.pic1.y,u.pic1.ray), /device,/linestyle
xyouts,u.pic1.x+u.pic1.ray,u.pic1.y+u.pic1.ray, 'PIC1',/device
endif
if u.pic2.et eq 1 then begin
plots, circle(u.pic2.x, u.pic2.y,u.pic2.ray), /device,/linestyle
xyouts,u.pic2.x+u.pic2.ray,u.pic2.y+u.pic2.ray, 'PIC2',/device
endif
if u.pic3.et eq 1 then begin
plots, circle(u.pic3.x, u.pic3.y,u.pic3.ray), /device,/linestyle
xyouts,u.pic3.x+u.pic3.ray,u.pic3.y+u.pic3.ray, 'PIC3',/device
endif
x2=long(x+zoom*cos(angle*!pi/180))
y2=long(y+zoom*sin(angle*!pi/180))
plots, circle(x,y,r),/device
plots, circle(x,y,r+n), /device,linestyle=1
plots,x2,y2,/device,psym=1,symsize=0.5
x1=2*x-x2
y1=2*y-y2
dy=y2-y1
dx=x2-x1
d=abs(dx)>abs(dy)
a=fltarr(d+1)
b=fltarr(d+1)
dy=dy/float(d)
dx=dx/float(d)
for i=0,d do begin
a[i]=x1+long(i*dx)
b[i]=y1+long(i*dy)
endfor
c=u.image(a,b)
widget_control, u.graph1, get_value=index
wset, index
sig=fltarr(d+1)
sig=1*(abs(a-x2) le abs(a-x1))*(abs(b-y2) le abs(b-y1))
sig(where(sig eq 0))=-1
coord=sig*sqrt((a-x)^2+(b-y)^2)
plot,coord,c,/ynozero,xrange=[-zoom,zoom]
oplot,[0,0],[min(c),max(c)],color=200
oplot,[-r,-r],[min(c),max(c)]
oplot,[r,r],[min(c),max(c)]
oplot,[-r-n,-r-n],[min(c),max(c)],/linestyle
oplot,[+r+n,+r+n],[min(c),max(c)],/linestyle
widget_control, u.graph2, get_value=index
wset, index
c=u.image[x,y-25:y+25]
plot, c, findgen(51)
m=max(c)                                                    mi=min(c)
oplot,[m,mi],[25-r,25-r]
oplot, [m,mi],[25-r-n,25-r-n], linestyle=1
oplot,[m,mi],[25,25], color=200
oplot, [m,mi],[25+r+n, 25+r+n], linestyle=1
```

```
oplot,[m,mi],[25+r, 25+r]
c=u.image[x-r-n:x+r+n,y-r-n:y+r+n]
widget_control, u.pic1.draw1, get_value=index
wset,index
shade_surf,c
end
'u-fit':begin
widget_control, u.zoom, get_value=zoom
widget_control, u.slide1, get_value=x
widget_control, u.slide2, get_value=y
widget_control, u.r, get_value=r
widget_control, u.n, get_value=n
widget_control, u.graph0, get_value=index
widget_control, u.d, get_value=g
widget_control,u.fitage, get_value=fitage
wset, index
tab0=circle(x,y,r)
tab1=circle(x,y,r+n)
result0=polyfillv(tab0[0,*],tab0[1,*],581,581)
result1=polyfillv(tab1[0,*],tab1[1,*],581,581)
dim1=size(result1)
n=dim1[1]
a=[]
for i=0,n-1 do begin
index=where(result0 eq result1[i], count)
if count eq 0 then a=[a,result1[i]]
endfor
d=a mod 581
e=long(a/581.)
f=u.image[d,e]
clone=u.image
if fitage eq 1 then begin
siz=size(f)
siz=siz[1]
start=fltarr(7)
err=fltarr(siz)+1
p=mpfit2dfun('myfunct',d,e,f,err,start)
clone(result1)=clone(result1)-myfunct(result1 mod 581,long(result1/581),p)
endif
if fitage eq 0 then begin
tab=transpose([[d],[e],[f]])
extra=sfit(tab,g,/irregular,/max_degree,kx=p)
clone(result1)=clone(result1)-horner2dbis(result1 mod 581,long(result1/581),p)
endif
widget_control, u.n, get_value=n
widget2, u.image,u.log,p,x,y,r,n,fitage,'PIC'
a=clone(result0)
n=size(result0)
n=n[1]
m=size(result1)
m=m[1]
b=clone(result1)
kicarre=(b)^2
kicarre=total(kicarre)
d=(a)^2
kicarre=kicarre-total(d)
result=total(a)
error=kicarre/float(m-n)
print, 'total couronne'
print, error
print, 'total pic'
```

```
print,result
print, 'kicarre'
print, kicarre/float(m-n)
widget_control, u.text, set_value=string(result)
end
endcase
end
pro widget3
pic={x:0l,y:0l,ray:0l,nap:0l,et:0l,flux:0d,vis:0d,error:0d,draw1:'',draw2:''}
struct={memo:'',terminer:'',controle:'',lancer:'',liste1:'',analyse:'',bas:'',ha
ut:'',couleur:'',date:'',fitage:'',error0:'',error1:'',error2:'',error3:'',rotat
ion:'',i:0l,taille:0l,etat:'',avancement:'',liste:'',list:strarr(50,2),adresse:'
',pic0:pic,
pic1:pic,pic2:pic,pic3:pic,t:0l,d:'',n:'',zoom:'',text:'',text0:'',text1:'',text
2:'',text3:'',slide1:'', slide2:'',
image:fltarr(581,581),log:fltarr(581,581),list1:fltarr(51),list2:fltarr(51),grap
h1:'', graph2:'',graph0:'', r:''}
base=widget_base(/row,uvalue=struct)
base2=widget_base(base, /column)
base1=widget_base(base, /column)
base5=widget_base(base,/column)
bouton=widget_button(base5,value='SELECTIONNER LE FICHIER A ANALYSER',
uvalue='u-repertoire')
esp=widget_label(base5,value='')
deux=widget_label(base5,font='BOLD', value='----------2.SELECTIONNER LES QUATRES
PICS----------')
basedeux=widget_base(base5,/column,frame=1)
base50=widget_base(basedeux, /row)
text0=widget_text(base50,value='', uvalue='text')
button0=widget_button(base50,value='PIC CENTRAL', uvalue='u-pic0')
tex0=widget_label(base50,value='error')
error0=widget_text(base50,value='',uvalue='u-error0')
base51=widget_base(basedeux, /row)
text1=widget_text(base51,value='', uvalue='text')
button1=widget_button(base51,value='PIC 1', uvalue='u-pic1')
tex1=widget_label(base51,value='error')
error1=widget_text(base51,value='',uvalue='u-error1')
base52=widget_base(basedeux, /row)
text2=widget_text(base52,value='', uvalue='text')
button2=widget_button(base52,value='PIC 2', uvalue='u-pic2')
tex2=widget_label(base52,value='error')
error2=widget_text(base52,value='',uvalue='u-error2')
base53=widget_base(basedeux, /row)
text3=widget_text(base53,value='', uvalue='text')
button3=widget_button(base53,value='PIC 3', uvalue='u-pic3')
tex3=widget_label(base53,value='error')
error3=widget_text(base53, value='', uvalue='u-error3')
rayon=widget_slider(basedeux,uvalue='u-slide',/drag,value=12,
title='rayon',minimum=0, maximum=50)
nappe=widget_slider(basedeux,uvalue='u-slide',/drag,value=5,title='nappe',
minimum=1, maximum=40)
zoom=widget_slider(basedeux,uvalue='u-slide',/drag, minimum=1,
maximum=50,value=25,title='zoom')
esp=widget_label(base5,value='')
trois=widget_label(base5,font='bold',value='----------3.ANALYSER ET
TERMINER----------')
basetrois=widget_base(base5,/column,frame=5)
baseliste=widget_base(basetrois,/row)
liste=widget_list(baseliste, value='', uvalue='u-list',xsize=50,ysize=7)
liste1=widget_list(baseliste, value='images traitees:', uvalue='u-
list1',xsize=50,ysize=7,sensitive=0)
```

```
base6=widget_base(basetrois, /row)
te=widget_label(base6, value=' Avancement:')
avancement=widget_text(base6, value='', uvalue='u-avancement')
te=widget_label(base6, value='Fichier en cours:')
etat=widget_text(base6, value='', uvalue='u-etat')
baseanalyse=widget_base(basetrois,/row)
analyse=widget_button(baseanalyse,value='ANALYSER', uvalue='u-
analyse',sensitive=0,xsize=166)
terminer=widget_button(baseanalyse,value='TERMINER',uvalue='u-
terminer',sensitive=0,xsize=166)
lancer=widget_button(baseanalyse,font='italic', value='automatique!',uvalue='u-
lancer',sensitive=0,xsize=160)
controle=widget_slider(basetrois,value=1,uvalue='u-
controle',minimum=0,maximum=1,xsize=50,title='Afficher controle?')
fitage=widget_slider(basetrois,value=0,uvalue='u-
fitage',minimum=0,maximum=1,title='methode d interpolation?')
degre=widget_slider(basetrois,uvalue='u-degre',minimum=0,
maximum=10,value=1,title='degre')
base3=widget_base(base5, /row)
quitter=widget_button(base3, value='QUITTER', uvalue='u-quitter')
fit=widget_button(base3,value='FIT',uvalue='u-fit')
ouvrir=widget_button(base3, value='OUVRIR', uvalue='u-ouvrir',sensitive=0)
text=widget_text(base5,value='', uvalue='text')
baseremer=widget_base(base5,/row)
basetext=widget_base(baseremer,/column)
t=widget_label(basetext,value='',ysize=50)
t=widget_label(basetext, value='Developed by Pierre
Deram',xsize=300,/align_center)
t=widget_label(basetext, value='for CARLINA Telescope',xsize=300,/align_center)
t=widget_label(basetext,value='H. Le Coroller & J.
Dejonghe',xsize=300,/align_center)
t=widget_label(basetext,value='June, 2012',xsize=300,/align_center)
memo=widget_draw(baseremer,xsize=201,ysize=201)
baseimage=widget_base(base1,/row)
image=widget_draw(baseimage, xsize=581,
ysize=581,/button_events,/motion_events,uvalue='u-image')
couleur=widget_draw(baseimage, xsize=15,ysize=580)
basecouleur=widget_base(baseimage, /column)
haut=widget_slider(basecouleur, minimum=0,maximum=100,uvalue='u-
echelle',value=100,ysize=290,/drag,/vertical,/suppress_value)
bas=widget_slider(basecouleur, minimum=0,maximum=100,uvalue='u-
echelle',value=0,ysize=290,/drag,/vertical,/suppress_value)
slide1=widget_slider(base1, xsize=581, maximum=480, minimum=100, uvalue='u-
slide',/drag)
base11=widget_base(base1,/row)
graph1=widget_draw(base11, xsize=581, ysize=300)
rotation=widget_slider(base11,uvalue='u-slide',minimum=-
180,maximum=180,value=0,title='rotation',ysize=300, /vertical,/drag)
base4=widget_base(base2, /row,/align_left)
graph2=widget_draw(base4, xsize=300, ysize=581)
slide2=widget_slider(base4,ysize=581, maximum=480, minimum=100, /vertical,
uvalue='u-slide',/drag)
graph20=widget_draw(base2, xsize=350,ysize=350)
struct.memo=memo
struct.terminer=terminer
struct.controle=controle
struct.liste1=liste1
struct.t=1
struct.d=degre
struct.zoom=zoom
struct.text=text
```

```
struct.text0=text0
struct.text1=text1
struct.text2=text2
struct.text3=text3
struct.error0=error0
struct.error1=error1
struct.error2=error2
struct.error3=error3
struct.slide1=slide1
struct.slide2=slide2
struct.graph1=graph1
struct.graph2=graph2
struct.graph0=image
struct.r=rayon
struct.n=nappe
struct.pic0.et=0
struct.pic1.et=0
struct.pic1.draw1=graph20
struct.pic2.et=0
struct.pic3.et=0
struct.list=strarr(50)+'a'
struct.liste=liste
struct.etat=etat
struct.avancement=avancement
struct.rotation=rotation
struct.fitage=fitage
struct.bas=bas
struct.haut=haut
struct.couleur=couleur
struct.analyse=analyse
struct.lancer=lancer
widget_control, base, set_uvalue=struct
widget_control, base, /realize
xmanager, 'widget3', base
end
```