INSA ROUEN
À taille humaine,
à l'échelle du monde

Projet de Physique P6-3
STPI/P6-3/2009 – 8

RECONSTRUCTION D'IMAGES PANORAMIQUES AVEC L'OBJECTIF FISH-EYE

Etudiants :

Sarah BOUHAYAT **Anne POLLET**

Xlang FANG **Gabriel ZEITOUNI MARTINS**

Yi HU

Enseignant-responsable du projet :

David HONORÉ

Date de remise du rapport : ***20/06/09***

Référence du projet : ***STPI/P6-3/2009 –8***

Intitulé du projet : Reconstruction d'images panoramiques avec l'objectif fish-eye

Type de projet : ***expérimental***

Objectifs du projet *(10 lignes maxi)* :

- Etude des déformations dues à l'objectif fish-eye.
- Correction des déformations.
- Reconstruction d'images panoramiques.

Si existant, n° cahier de laboratoire associé : ***A 30227***

INSTITUT NATIONAL DES SCIENCES APPLIQUEES DE ROUEN
Département Sciences et Techniques Pour l'Ingénieur
BP 8 – place Emile Blondel - 76131 Mont-Saint-Aignan - tél : 33 2 35 52 83 00 - fax : 33 2 35 52 83 69

TABLE DES MATIERES

NOTATIONS, ACRONYMES

1. INTRODUCTION

1.1. Description de l'appareil fish-eye

Pour ce projet, les photos ont été prises avec un appareil à objectif fish-eye. C'est un objectif qui possède un champ angulaire très grand, c'est à dire 180° et donc une distance focale très courte.

L'appareil utilisé est un NIKON D40X.

Le tableau ci-dessous représente les caractéristiques de cet appareil :

Type d'objectif	Nikkor DX AF de type G avec processeur et monture baïonnette Nikkor
Focale	10,5 mm
Ouverture maximale	f/2,8
Construction optique	Une lentille ED
Champ angulaire	180°
Système de projection	Projection à angle équisolide
Mise au point	Manuelle via une bague de mise au point séparée
Distance de mise au point minimale	0,14 m
Diaphragme	Automatique
Dimensions	Diamètre : 63 mm*rallonge 62,5 mm de la bride de montage d'objectif de l 'appareil
Poids	305 g

Le diaphragme permet de réguler la netteté de la photo. Le diaphragme est composé de lamelles métalliques qui reconstituent un « iris ». Son ouverture adaptable permet de fixer la profondeur de champ et d'agir sur la quantité de lumière qui pénètre dans l'appareil. Bien régler l'ouverture du diaphragme permet d'éviter certaines aberrations chromatiques.

Pour un objectif fish-eye, l'image projetée suit une loi mathématique, il y existe plusieurs types de projections, c'est selon le fabricant. La projection équisolide est la plus courante.

1.2. Avantages et Inconvénients du fish-eye

Contrairement aux autres objectifs, l'objectif fish-eye permet de prendre des photos de très grands champs visuels grâce à une distance focale très courte. Des fois, l'angle de champ atteint 180° dans la diagonale voire dans toute la photo.

L'objectif fish-eye est donc très prisé des amateurs de photos de paysages ou de photos d'architecture. Cependant, cet objectif présente quelques petits inconvénients. En effet, la capacité à prendre des photos de très grand angle s'accompagne de nombreux défauts optiques : distorsions géométriques en barillet assez importantes, des aberrations chromatiques ainsi que d'autres défauts optiques. Toutes ces déformations ne sont pas négligeable surtout lorsqu'on recherche à obtenir des photos de grande qualité.

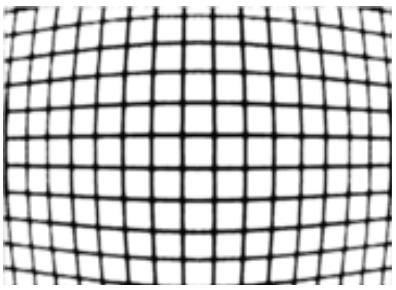
Exemple de photo :

Figure 1 : photo d'un bâtiment de l'INSA prise avec objectif usuel

Figure 2 : photo d'un bâtiment de l'INSA prise avec objectif fish-eye

La photo représentée en Figure 1 est la photo d'un bâtiment de l'INSA prise avec un objectif standard. Aucune déformation ni aberration optique n'est visible sur cette image.

La photo de la Figure 2 est une photo du même bâtiment que ci-dessus prise au même endroit (la place du pied de l'appareil est inchangée) prise avec un objectif fish-eye. On remarque que le champ de vision est plus grand mais que l'image présente une déformation en barillet. Cette distorsion est due au fait que l'objectif ne respecte pas les proportions de l'objet photographié entre le centre de l'image et les périphériques.



La Figure 3 représente la déformation en barillets (comme dans la Figure 2). L'objet photographié apparaît comme bombé. Cela est du au fait que l'objectif produit une image plus grande de la partie centrale. Les lignes droites des périphériques sont donc incurvées vers l'extérieur.

Figure 3 :distorsion en barillet

Cependant, ces déformations ne sont pas irréversibles. Il existe de nombreux logiciels tels que Photoshop ou même des logiciels donnés par les constructeurs d'appareils photos qui permettent de corriger les déformations dues aux objectifs fish-eyes.

Le principal objectif de notre projet est d'arriver à corriger ces déformations en écrivant un programme à l'aide du logiciel Scilab. La programmation repose sur la formule théorique de correction qui est :

$$X = x - k.x.(x^2+y^2)$$

$$Y = y - k.y.(x^2+y^2)$$

X et Y désignent les positions des croix en cm sur la mire.

x et y désignent les positions des croix en pixel.

k désigne le coefficient de correction à déterminer.

2. METHODOLOGIE / ORGANISATION DU TRAVAIL

2.1. Objectifs et moyens

Nous avons pu voir dans la partie précédente qu'avec un objectif fish-eye, on obtenait des déformations en barillet et donc que la photo prise n'est pas droite.

Figure 4 : image originale

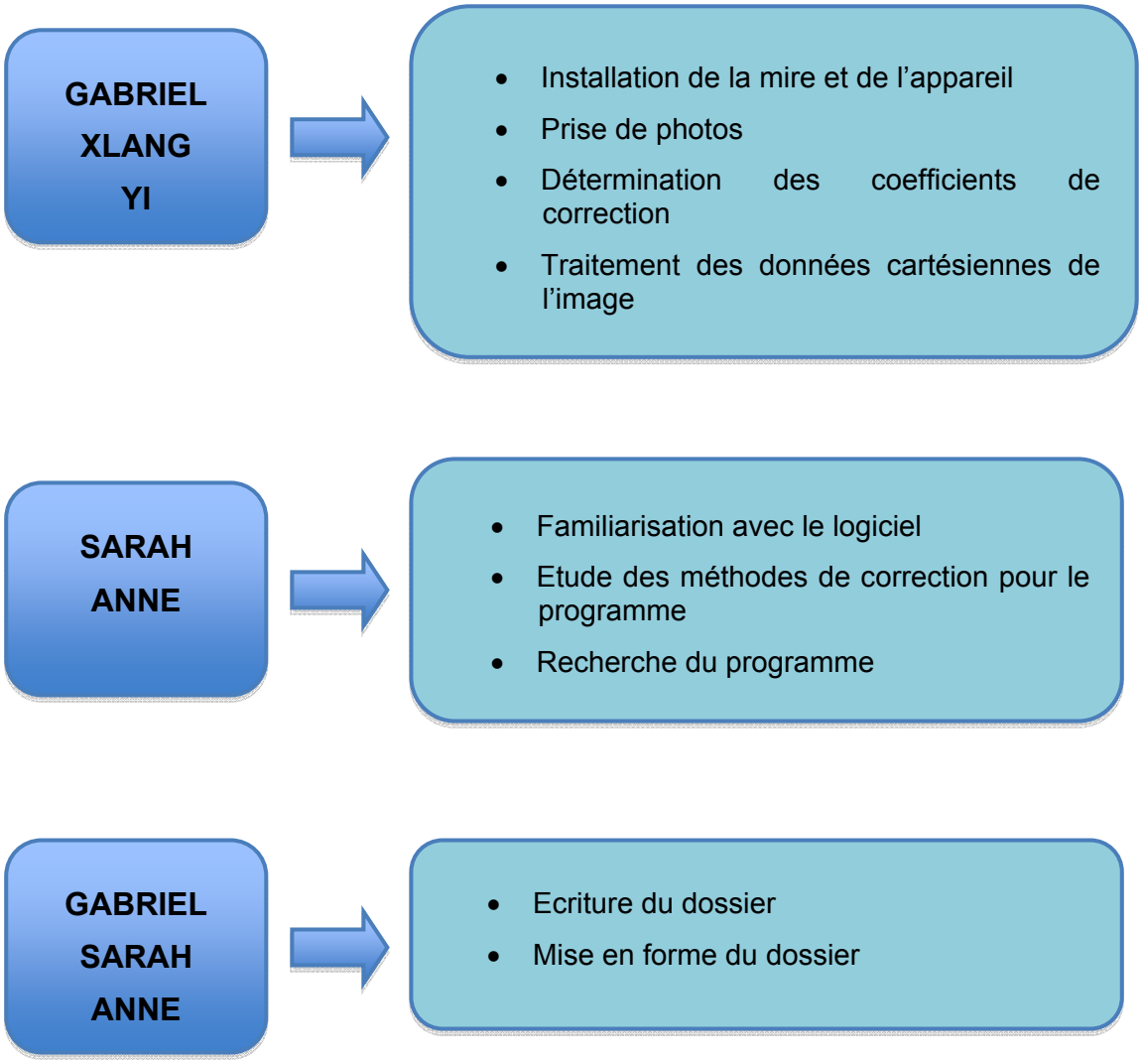
Figure 5 : image corrigée avec DxO OPtics Pro [1]

Le principal objectif du projet est donc de chercher à corriger cette déformation. Pour cela, nous avons utilisé une mire sur laquelle des croix ont été imprimées, et un appareil photo NIKKON D40X. Cet appareil photo équipé d'un objectif fish-eye a permis d'obtenir une image déformée. Ensuite grâce à un logiciel de programmation, nous avons tenté de trouver un programme qui permettrait de transformer l'image déformée obtenue à partir de l'appareil

photo en une image droite, c'est à dire, comme si cette image avait été prise avec un appareil photo qui n'engendre pas de déformation de la photo.

2.2. Répartition des tâches :

Voilà ci-dessous un organigramme représentant les différentes tâches accomplies par les membres du groupe :



Nous avons dès le début du projet, décidé de séparer notre groupe en deux. Une partie du groupe s'est occupé de la mire et de la prise de photo et l'autre partie, de tout ce qui est programmation. Une division des tâches nous a permis d'avancer plus rapidement : pendant qu'une partie s'est occupée de prendre les photos l'autre partie s'est intéressée à la programmation et notamment à la compréhension du fonctionnement du logiciel Scilab.

Cette organisation a rendu le travail plus efficace.

3. ETUDE NUMERIQUE DE L'IMAGE

Ayant comme but la correction de la distorsion causée par l'objectif fish-eye, une étude numérique détaillée de l'image a été faite afin de quantifier et qualifier les aberrations.

3.1 Prise de photo

3.1.1. Choix de la mire

Parmi les trois mires disponibles, deux étaient composées de croix et se ne différaient que par la taille de celles-ci. La troisième était composée de carrés.

Afin de faciliter la cartographie, la mire composée de carrés a été exclue. Parmi les deux autres mires, nous avons choisi la plus grande. En effet, celle-ci permettait de prendre une photo à plus grande distance de la mire et par conséquent avec un plus grand nombre de croix, ce qui augmentait la précision de notre travail.

Ensuite, nous avons attaché la mire sur la fenêtre, obtenant de cette façon, une bonne luminosité et un bon contraste.

3.1.2. Problématique et méthodologie

L'objectif étant d'obtenir une photo parfaitement droite, nous avons utilisé le logiciel Paint, afin de comparer les coordonnées des croix de la colonne centrale des photos prises, zone où la distorsion est nulle. A l'échelle des pixels nous nous sommes rendus compte de la difficulté à obtenir une image droite

Après plusieurs échecs, nous avons décidé d'utiliser les pieds de la caméra, ce qui nous permettait d'avoir une image bien centrée sur la croix du centre de la mire puis on a fait légèrement varier le positionnement verticale de la caméra en prenant des séquences des photos. De cette manière, nous avons finalement obtenu une photo avec une incertitude acceptable.

3.2. Notre image

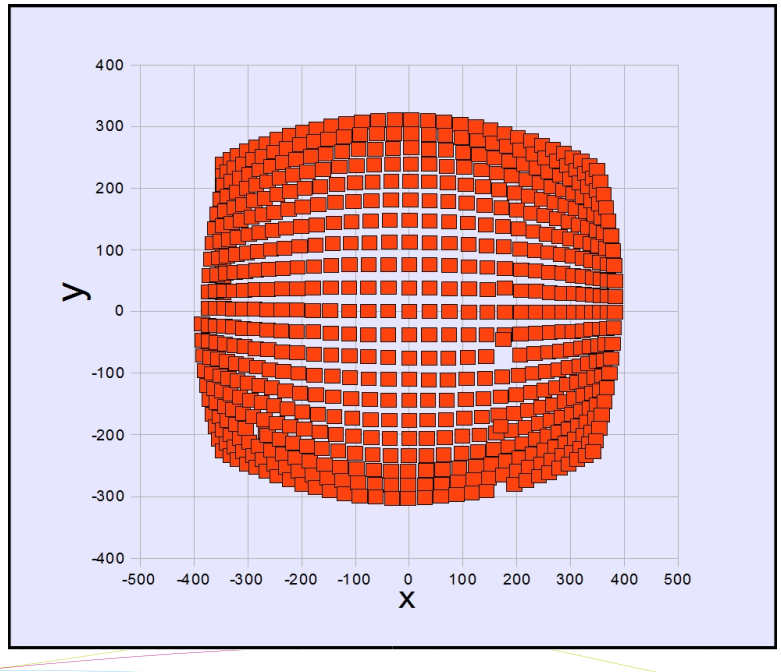


Figure 6 : Représentation de la photo déformée

Nous avons repéré les coordonnées de chaque croix en pixel à l'aide du logiciel Paint, que nous avons par la suite placées dans un tableau Excel. Nous avons donc utilisé les données du tableau pour reproduire l'image déformée que nous avions.

La figure 6 représente donc l'image déformée que nous avons obtenu après traitement des coordonnées du tableau.

3.2.1. Calcul d'incertitude

On appelle y' un axe imaginaire parfaitement droit. On cherche l'angle formé entre cet axe y' et celui de de notre photo.

Pour cela, on mesure la distance entre le centre de l'image, c'est-à-dire où les deux axes se croisent, et deux extrémités.

On obtient donc : $\theta \approx 0{,}735$ °

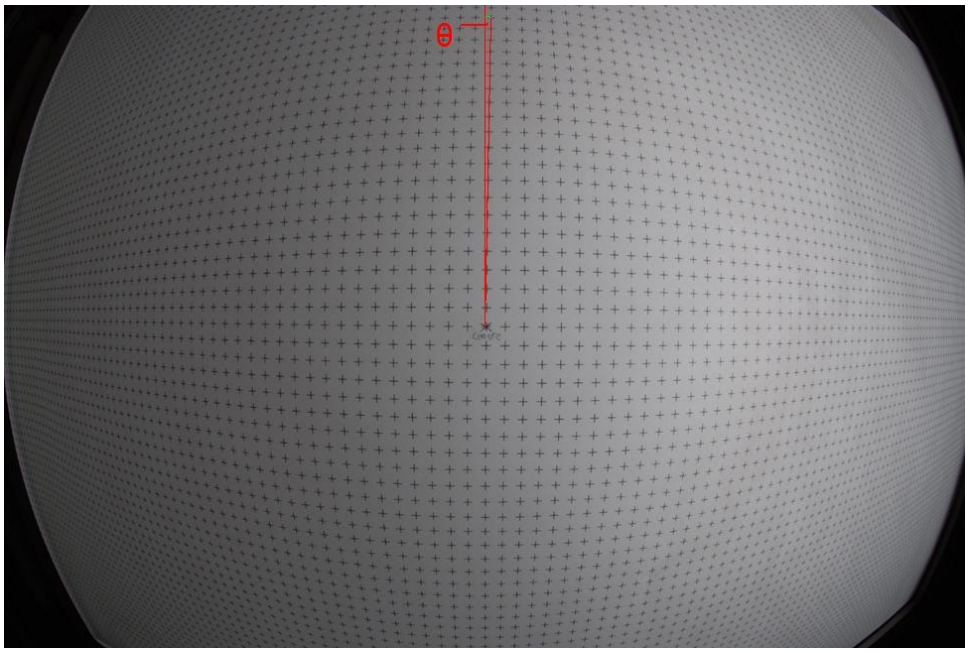


Figure 7 : Mire avec angle d'incertitude

La Figure 7 montre la photo finale de la mire avec l'angle d'incertitude présent.

3.2.2. Cartographie

Nous avons utiliser le logiciel Excel pour construire un tableau composé de deux colonnes X et Y, relatant les coordonnées des croix.

Dans ce tableau, le point (0,0) représente la croix qui se trouve a l'extrémité, en haut à gauche de la photo. La valeur de X augmente quand on se déplace vers la droite, et la valeur de Y augmente lorsque l'on descendait. Il a fallu donc déplacer notre repère vers le centre de l'image.

Afin réaliser ce changement de repère, on a utilisé les fonctions :

$$x = X - f \text{ et } y = g - Y.$$

g et f sont des constantes qui représentent la moitié de la distance de deux croix opposés que se trouvent respectivement sur les extrémités de l'axe vertical et horizontal.

x et y représentent les nouvelles coordonnées des croix respectivement sur l'axe horizontal et vertical.

3.2.3. Calcul de la distorsion

Les distorsions sur l'axe horizontal et vertical sont appelés respectivement δx et δy. Pour les calculer il nous faut Xc et Yc.

$$Xc = mPx \text{ et } Yc = nPy$$

m représente le nombre de la ligne et *n* de la colonne.

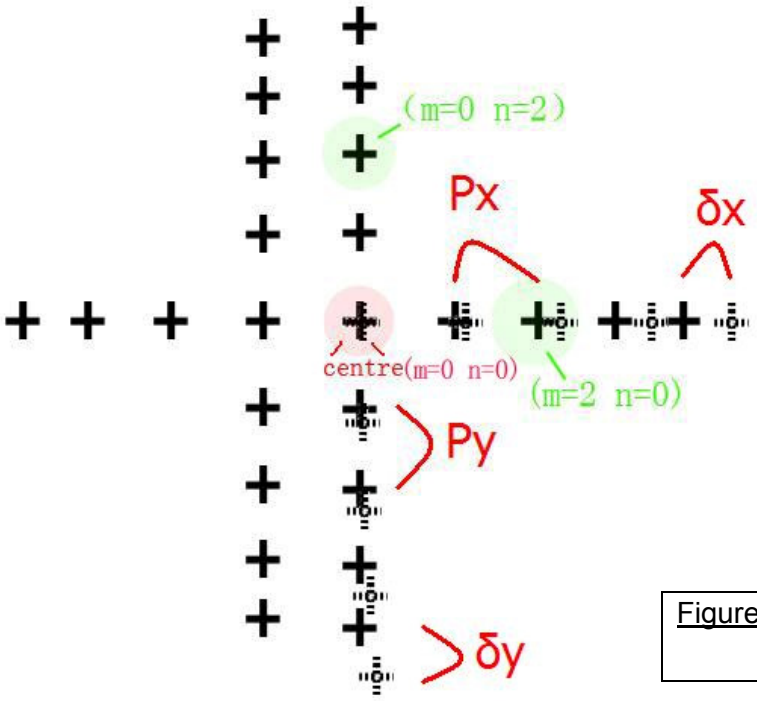


Figure 8 : Illustration des variables

La Figure 8 présente les différentes variables utilisées pour les calculs de distorsion.

Donc on peut finalement calculer les distorsions d'après les formules

$$\boldsymbol{\delta x = x - Xc} \text{ et } \boldsymbol{\delta y = y - Yc.}$$

3.2.4. Analyse et interprétations

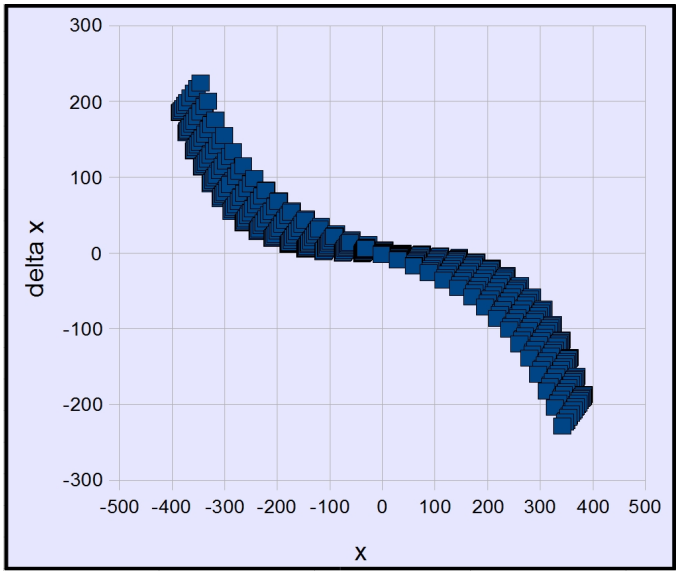


Figure 9 : Ecart des pixels horizontaux en fonction de x

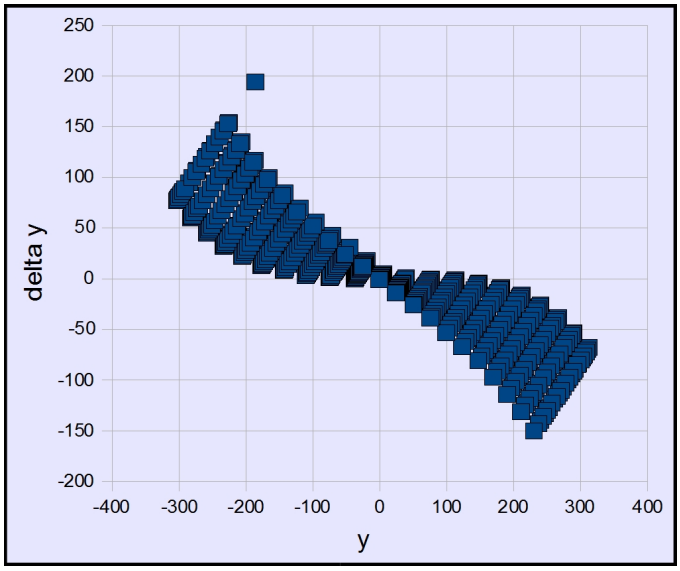


Figure 10 : Ecart des pixels verticaux en fonction de y

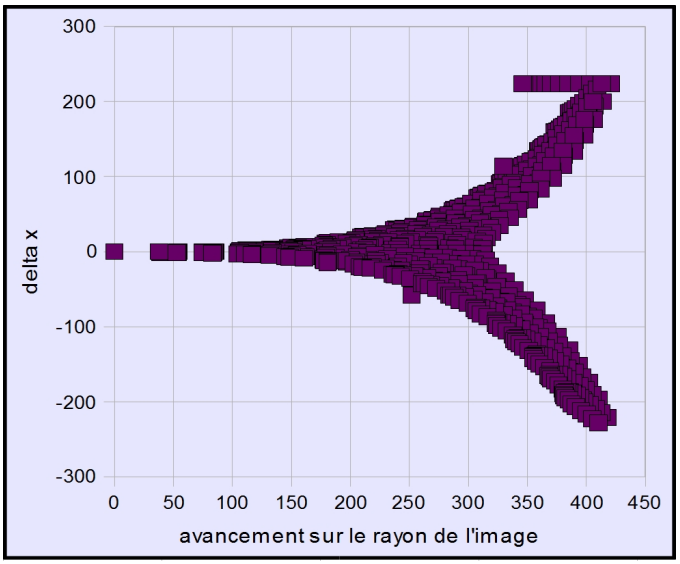


Figure 11: Ecart des pixels horizontaux en fonction du rayon

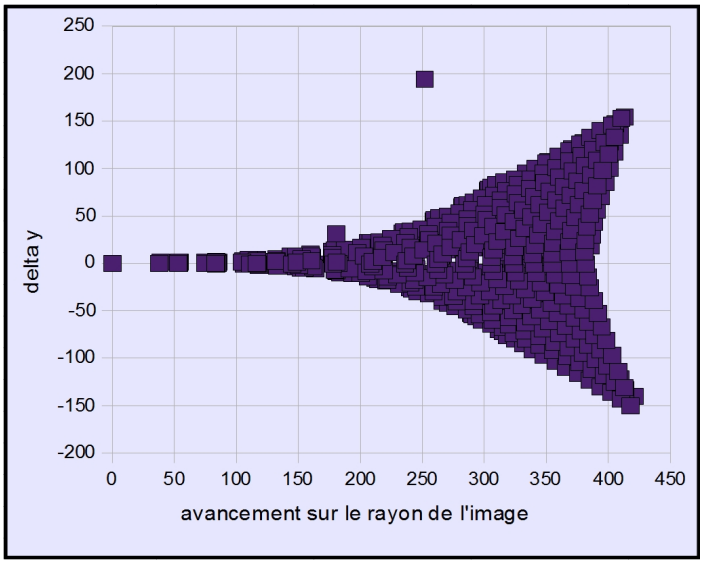


Figure 12 : Ecart des pixels verticaux en fonction du rayon

D'après les diagrammes ci dessus, c'est a dire de la Figure 9 et la Figure 10, on peut constater que les répartitions en fonction de x et y sont symétriques. On en déduit que les déformations sont équivalentes le long de cercles concentriques par rapport au centre de l'image.

De plus, d'après les diagrammes 11 et 12 on peut voir que la distorsion augmente plus le rayon de l'image s'agrandit ; l'évolution de la distorsion n'est donc bien pas linéaire, comme le prédit la théorie.

Sur deux diagrammes, Figure 10 et 12, on peut remarquer des quelques points singuliers qui sont dus à des erreurs pendant l'extraction des coordonnées des croix de la photo.

3.3. Description du logiciel utilisé

Pour la partie informatique, nous avons utilisé un logiciel qui permettait de faire des études graphiques : Scilab.

Scilab est un logiciel gratuit qui a été développé par l'INRA, Institut National de Recherche en Informatique et automatique. C'est un logiciel de calcul numérique qui possède des fonctions préprogrammées, des boîtes à outils, appelées Toolbox. Il dispose aussi d'un éditeur et d'un terminal. Le langage de programmation ressemble à celui de Matlab, nous avons donc dû nous familiariser avec ce langage.

Nous n'avons jamais utilisé le logiciel Scilab auparavant. Il a donc fallu se familiariser avec le logiciel et son mode de fonctionnement.

Le choix de ce logiciel est avant tout stratégique : il permet de traiter les images grâce à des applications qui y sont incluses, il dispose donc déjà de méthodes et des fonctions spécifique à l'étude des images telles que la lecture ou l'écriture d'images ou encore le calcul sur les positions des pixels.

Pour pourvoir travailler sur les photos que nous avions prises, il faut télécharger la Toolbox spécifique au traitement d'images appelée « siptoolbox ». Au début, la Toolbox présentait une erreur et ne se charger pas complètement, avec l'aide fournie sur le site officiel de Scilab, on a pu remédier très facilement à cette erreur en téléchargeant un complément à la boîte à outils.

A chaque démarrage du logiciel, il faut télécharger la ToolBox pour pouvoir travailler sur des photos en effet, c'est comme si l'on devait créer un environnement propre à l'étude numérique d'images.

3.4. Description du principe de programmation

Le langage de programmation utilisé sous Scilab est très proche du langage C ou C++ que nous avons déjà utilisé auparavant en cours d'informatique.

Le principe de programmation est simple. En effet, il faut d'abord télécharger l'image originale, qui est déformée, concrètement cela correspond à la lecture d'une image par le logiciel Scilab qui ensuite stocke l'image sous forme d'une matrice dont le nombre de lignes et de colonne correspondent à la taille de l'image en pixels. Par exemple pour une image de taille 968×648, la matrice correspondante est une matrice de 968 colonnes et 648 lignes.

Ensuite il faut effectuer un calcul sur la position des pixels et corriger cette dernière pour redresser l'image : pour cela un calcul est réalisé pour chaque pixel de l'image et c'est la position du pixel qui est changée pour pouvoir corriger l'image.

Pour créer le programme permettant de corriger la distorsion nous avons utilisé la méthode suivante :

Une image est constituée de nombreux pixels. Nous nous sommes basés sur les matrices des images en pixel pour constituer notre programme.

Nous nous sommes donc intéressés à la position de chaque pixels.

En effet à chaque image correspond une matrice. Avec le logiciel nous pouvions extraire la matrice de pixels de l'image déformé.

Ci-dessous voici un schéma explicitant la démarche que l'on a eu au niveau des matrices de pixels.

Figure 13 : Matrice de l'image déformée

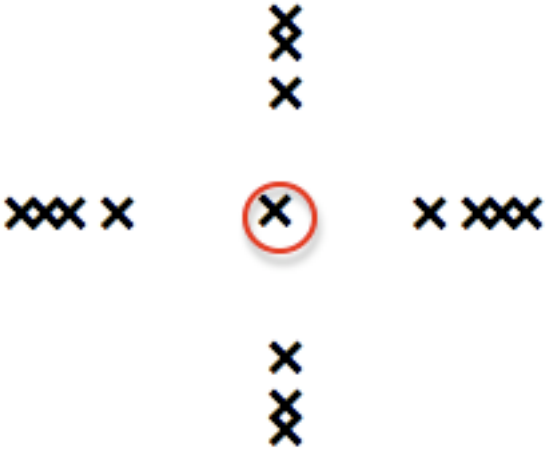


Figure 14 : Matrice de l'image déformée

La Figure 13 représente la matrice de pixel de l'image non déformée. A partir de l'image déformée, nous obtenions une matrice de pixels représentée par l'image 14. Sur cette dernière toutes les croix ne sont pas représentées mais chacune subit la même modification. A partir de cette matrice nous cherchions a retrouver pour un pixel donné, le pixel correspondant au pixel de l'image non déformée. Pour cela nous utilisions la formule donnée au début du dossier :

$$X = x - k.x.(x^2+y^2)$$

$$Y = y - k.y.(x^2+y^2)$$

Le calcul consiste à affecter à chaque pixel une nouvelle position qui permet de redresser l'image.

Le programme final figure en annexe

3.5. Résultats

Pour tester le programme et voir la correction réalisée par ce dernier, nous avons fait le choix de travailler sur le quart de la photo. Le calcul à effectué est assez laborieux et prend donc du temps. En réduisant la taille de l'image ou en choisissant de ne travailler que sur une partie de l'image on gagne du temps : on vérifie d'abord que le programme marche sur une petite partie de l'image et ensuite si celui-ci marche l'appliquer à toute l'image.

Figure 15 : image résultat retournée par le programme.

L'image ci-dessus, Figure 15, est l'image obtenue après que le programme ait réalisé le calcul. Le fond noir correspond à l'image, les « points verts » sont en fait les pixels : à chaque fois que le programme trouvait un pixel et en corrigeait la position un point vert apparait sur la photo que le programme retourne.

On remarque que plus on s'approche du centre de l'image plus les pixels en vert sont alignés sur des lignes droites. Par contre, plus on s'éloigne vers les bords plus les pixels verts sont en désordre.

Cependant, les positions des pixels verts se rapprochent plus des positions des pixels sur la photo corrigée. Pour avoir une meilleure correction, il faudrait sur un échantillon de pixels plus petit.

4. CONCLUSION ET PERSPECTIVES

4.1. Conclusion sur le travail réalisé

A la fin de la durée du projet, nous n'avons pas pu aboutir à une bonne correction. L'image finale obtenue n'est pas satisfaisante car il reste encore des pixels qui ne sont pas corrigés. De plus nous n'avons pas eu l'occasion de tester le programme pour l'ensemble de la mire, nous n'avons travaillé que sur le quart de la photo de la mire que nous avons prise. Pour avoir une idée plus générale de l'efficacité du programme, il aurait aussi fallu le tester sur la photo d'un objet réel comme par exemple sur la photo du bâtiment de l'INSA.

Pour la suite et pour les futurs projets en rapport avec l'objectif fish-eye, des améliorations sont à faire. Il faudrait travailler avec des images de plus petite dimension pour faciliter le chargement sous Scilab. Ou encore, diviser la photo en plusieurs petites zones pour avoir de meilleurs résultats. Cela implique le calcul de plusieurs coefficients de correction spécifiques à chaque zone qui seront alors en quelque sorte proportionnels au niveau de la distorsion. De plus il faudrait sûrement ajouter des interpolations.

Le projet était peu être trop ambitieux à la vue de la durée du projet. Etant donné qu'aucun de nous n'avait de connaissance en photographie ou encore en programmation sur Scilab, on a perdu du temps au départ le temps que l'on cerne bien le projet, que l'on se répartisse les tâches et que l'on se familiarise avec le logiciel Scilab.

4.2. Conclusions sur l'apport personnel de cette U.V. projet

Ce projet de physique nous a permis d'approfondir nos connaissances en photographie d'autant qu'aucun de nous n'était familier avec la photographie.

En effet pour nous tous, nous avons découvert ce domaine et les nombreuses explorations qu'il offre. Nous avons aussi pu nous rendre compte que la photographie pouvait se rapporter au domaine de l'ingénierie notamment à la branche de l'optique ou la branche de l'informatique pour le traitement numérique des images.

Il nous a aussi permis d'expérimenter le travail de groupe. C'est la première fois que nous sommes confrontés à travailler en équipe avec des personnes que nous ne connaissons pas forcément ; chose qui n'est facile à réaliser et qu'il faut apprendre à gérer. Il faut se répartir les tâches, échanger ses idées avec ses coéquipiers et les défendre tout en gardant un esprit de groupe et une ambiance détendue mais sérieuse.

On a ainsi pu découvrir que le travail en groupe pouvait être enrichissant, il permet d'explorer de nombreuses pistes de recherche auxquelles chacun de nous individuellement n'y avait pas pensé. Mais il présente aussi quelques inconvénients : il faut une parfaite organisation pour que chaque tâche soit réalisée en temps voulu pour que chaque partie du projet puisse être menée à bien. Il faut donc une synchronisation qui est favorisée par une communication continue et efficace au sein du groupe. Des fois, la barrière de la langue se posait aussi.

Il est vrai que plus tard dans notre vie professionnelle nous serons amenés à avoir des relations au niveau international et il faut savoir gérer cela pour éviter justement que la langue soit une barrière. Dans ce projet on retrouve cette situation et nous avons pu nous organiser de manière à ce que tout le monde puisse accomplir une tâche même si la langue pouvait poser problème.

D'autre part, le projet n'était en relation direct avec aucune matière qui nous est enseignée dans le sens où aucun pré-requis n'était nécessaire et nous n'avons jamais utilisé nos cours. Mais le travail de recherche et le travail de groupe sont des aperçus de ce que sera notre futur travail d'ingénieurs.

Pour conclure, les apports de ce projet sont nombreux : il nous a permis d'avoir une idée un peu plus claire de ce que serait notre travail plus tard et donc de réaliser l'importance des relations humaines pour la bonne conduite d'un projet.

4.3. Perspectives pour la poursuite de ce projet

Pour la poursuite de ce projet il serait intéressant de reconstituer une image panoramique avec un autre logiciel pour pouvoir évaluer la différence entre celui que l'on a utilisé dans notre projet.

De plus, après avoir essayé plusieurs projet avec un appareil photo ayant un objectif fish-eye, pourquoi ne pas regarder pour les caméras. En effet le fish-eye est aussi utilisé pour des vidéos.

5. BIBLIOGRAPHIE

[1] http://www.dxo.com/fr/photo/dxo_optics_pro/optics_geometry_corrections/distortion (valide à la date du 31/05/09).

[2] www.panochrome.fr/articles/Angle_de_champs.pdf (valide à la date du 31/05/2009).

[3] www.ann.jussieu.fr/~postel/scilab/NoticeScilab.pdf (valide à la date du 31/05/2009).

[4] ufrsciencestech.u-bourgogne.fr/licence2/i3c/TraitementImage/TP/TP1.pdf (valide à la date du 31/05/2009).

[5] Diane Ligrand, « Introduction au traitement de l'image 2ème édition », Vuibert, 2008.

6. ANNEXES

6.1. Programme utilisé

```
//              Chargement d'une image Nikon pour traitement
//
//
//il faut tout d'abord changer de répertoire
//il faut ensuite lancer la siptoolbox

clc
stacksize(65000001)

// vérification du chargement de la Toolbox
if ~isdef('SIPDIR') then
  x_message_modeless('SIPToolbox n"est pas chargé');
  abort;
end

// lecture de l'image
chdir("C:\Users\sirsirette\Desktop\P6-3");
NomImage='DSC_0660mini.jpg';
image_in=gray_imread(NomImage);
xbasc();
imshow(image_in);
halt();

// traitement de l'image
image_out=0;
dimensions_image_in=size(image_in);
xmax=dimensions_image_in(1);
ymax=dimensions_image_in(2);

kx=-1.3*10^(-6);
```

```
ky=kx;
disp(kx);

for Xp=1:xmax/4
   for Yp=1:ymax/4
     Xc(Xp,Yp)=Xp+kx*Xp*(Xp*Xp+Yp*Yp);
     Yc(Xp,Yp)=Yp+ky*Yp*(Xp*Xp+Yp*Yp);
   end
end

maximum=max(max(Xc),max(Yc));
disp(maximum);
image_out=0;

for Xp=1:xmax/4
   for Yp=1:ymax/4
      Xc2=round(Xc(Xp,Yp)/maximum*200);
      Yc2=round(Yc(Xp,Yp)/maximum*200);

      image_out(Xc2,Yc2,1)=image_in(Xp,Yp,1);  // couleur R
      image_out(Xc2,Yc2,2)=1;                  // couleur G
      image_out(Xc2,Yc2,3)=0;  // couleur B
   end
end

// affichage et enregistrement de l'image
xbasc();
imshow(image_out);
imwrite(image_out,'ImageResultat.jpg');
```