
Correction de la distorsion radiale : une approche non-linéaire

Ding Dong! “Tiens, qui c’est?” ... je regarde par le juda. C’est quand même fou, toutes les lignes droites du palier de ma porte apparaissent courbées!

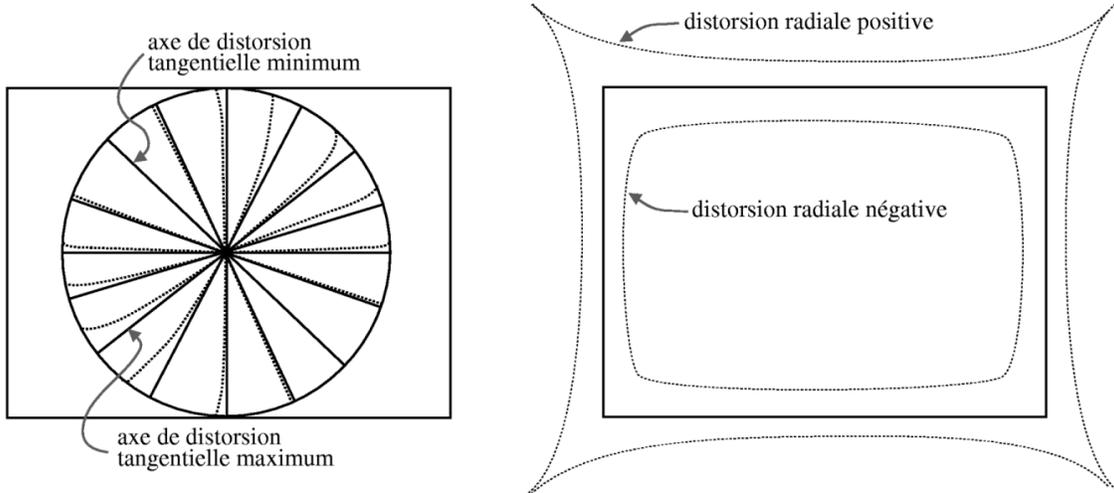
Il s’agit de distorsion radiale, une altération de l’image due à l’optique utilisée et en particulier à la position du diaphragme dans cette optique. Nous proposons dans ce sujet de corriger cette distorsion radiale sur une image prise avec un appareil photo ou une caméra. Ce genre de correction est souvent sollicité en photogrammétrie (reconstruction d’objets 3d partir d’images) ou comme prétraitement avant un trucage sur une séquence vidéo.



A gauche une image distordue prise avec une focale courte et à droite la même image réctifiée. Les droites sont droites.

1 Position du problème

La distorsion radiale est une altération de la prise de vue que l’on rencontre sur des dispositifs d’acquisition d’images tel qu’un appareil photo ou une caméra. Cette distorsion a tendance à arrondir les bords de l’image. L’exemple le plus frappant est le juda de la porte d’entrée ou les fish-eyes. Cette distorsion est souvent plus flagrante sur des appareils à courte focale (grand angle). La distorsion radiale est due à la position du diaphragme dans le système optique : plus le diaphragme est loin du centre optique, plus la distorsion radiale est forte. Il existe une autre forme de distorsion dite tangentielle. Celle-ci se caractérise par une sorte de rotation autour du centre de l’image. L’amplitude de cette rotation varie selon la position des points sur l’image. Cette distorsion est toutefois négligeable devant la distorsion radiale et nous ne la traiterons pas dans ce sujet.



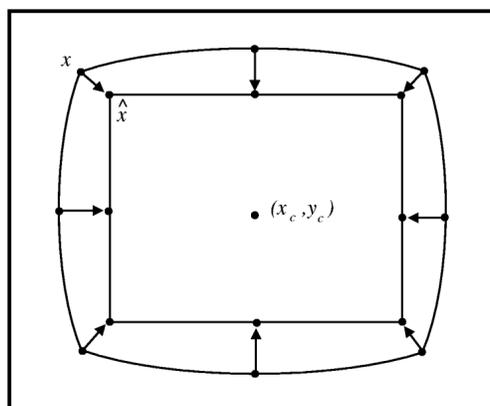
Distorsion tangentielle et distorsion radiale.

2 Modélisation de la distorsion radiale

Comme son nom l'indique, la distorsion radiale affecte différemment les points selon leur distance avec le centre de distorsion. La distorsion radiale révèle un caractère non-linéaire ce qui complique légèrement sa modélisation. Elle peut être modélisée par un polynôme vérifiant certaines contraintes.

2.1 Modèle mathématique

Soit (x_c, y_c) le centre de distorsion, il s'agit en général du point principal (de l'anglais *principal point*) qui correspond à l'intersection entre l'axe optique et le plan image. Il arrive que le centre de la distorsion radiale soit décentré par rapport à l'image dans le cas d'assemblages d'optiques de mauvaise qualité. Ce centre de distorsion peut toutefois être assimilé au centre de l'image sans trop d'effets néfastes sur le modèle. Soit (x_i, y_i) un ensemble de points sur l'image distordue et (\hat{x}_i, \hat{y}_i) les points correspondants sur l'image rectifiée.



La relation entre ces deux ensembles de points peut s'écrire de la façon suivante :

$$\begin{cases} \hat{x}_i = x_i + \Delta x \\ \hat{y}_i = y_i + \Delta y \end{cases}$$

avec

$$\begin{cases} \Delta x = (x_i - x_c)(k_1 r^2 + k_2 r^4) \\ \Delta y = (y_i - y_c)(k_1 r^2 + k_2 r^4) \end{cases}$$

et

$$r^2 = (x_i - x_c)^2 + (y_i - y_c)^2$$

Nous utilisons ici un polynôme de degré 4 pour r en ne gardant que les puissances paires, ce qui est réputé pour être suffisant, il est toutefois possible d'utiliser un polynôme de degré 2 (un seul coefficient k_1) ou de degré 6 ou plus, toujours en ne gardant que les puissances paires. La correction de la distorsion radiale est donc directement liée à la recherche des coefficients k_1, k_2, \dots, k_n .

2.2 Corriger une image

Si les coefficients sont k_1, k_2, \dots, k_n connus, on peut corriger la distorsion radiale d'une image en appliquant les formules du paragraphe précédent. Il se peut que vous obteniez une image légèrement bruitée. A vous de voir pourquoi et de **chercher des solutions** pour y remédier.



Des courbes parasites peuvent apparaître.

Validation : avant de passer au code du calcul des paramètres k_1, k_2, \dots, k_n , vérifiez que votre fonction de correction d'image fonctionne avec une image test. Sur une image 800×600 , essayez les paramètres $k_1 = 10^{-10}$ et $k_2 = 10^{-12}$ ou bien $k_1 = 2, 5 \cdot 10^{-7}$.

3 Calcul des paramètres de correction : mode opératoire

D'une certaine manière, corriger la distorsion radiale d'une image revient à redresser les droites qui ne le sont pas, mais qui devraient l'être. C'est l'approche que nous avons choisie pour traiter ce problème. Nous allons donc redresser les droites qui ne sont pas droites en les transformant avec notre modèle de correction de distorsion radiale. En choisissant les coefficients k_1 , k_2 , ... k_n du polynôme de correction de façon adéquate, nous pouvons minimiser la courbure de ces segments de droite.

En bref, la méthode proposée se décompose en 3 étapes qui sont :

1. La sélection manuelle de points supposés alignés sur l'image originale.
2. Le calcul non-linéaire des paramètres de correction de distorsion radiale permettant d'aligner au mieux les points supposés alignés.
3. La correction de la distorsion sur l'image distordue.



(1)



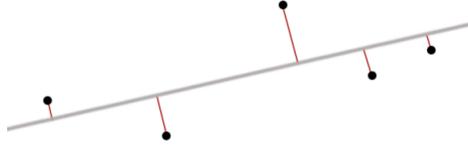
(2)



(3)

4 Des droites pas droites

La première étape consiste à sélectionner un ensemble de points supposés alignés. Afin de définir à quel point ils ne sont pas tout à fait alignés, on calcule d'une part la droite moyenne qui passe au mieux par ces points (au sens des moindres carrés) puis on calcule un résidu correspondant à la somme des distances entre chaque point et cette droite.



Un point $\mathbf{x} = (x, y, w)^\top$ de \mathbb{P}^2 appartient à la droite $\mathbf{l} = (a, b, c)^\top$ ssi $\mathbf{x}^\top \mathbf{l} = \mathbf{l}^\top \mathbf{x} = 0$.

Considérons un ensemble de points \mathbf{x}_i supposés alignés, la droite \mathbf{l} passant au mieux par ces points au sens des moindres carrés doit essayer de satisfaire au mieux l'équation $\mathbf{x}_i^\top \mathbf{l} = 0$ pour chaque point \mathbf{x}_i , ce qui revient à résoudre le système suivant :

$$\begin{bmatrix} x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & w_n \end{bmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Ce genre de système peut se résoudre en utilisant une décomposition en valeurs singulières (*SVD* en anglais). Vous trouverez ce genre d'outils dans la bibliothèque **OpenKraken**.

Le résidu de cet ensemble de points correspond à la somme des distances entre chaque point \mathbf{x}_i et cette droite \mathbf{l} . Nous rappelons que la distance entre un point $\mathbf{x} = (x, y, 1)^\top$ et la droite $\mathbf{l} = (a, b, c)^\top$ se calcule par :

$$d(\mathbf{x}, \mathbf{l}) = \frac{|\mathbf{x}^\top \mathbf{l}|}{\sqrt{a^2 + b^2}}$$

Pour une image donnée, le résidu de l'image R est la somme des résidus des droites de l'image sélectionnées par l'utilisateur. Pour des questions de stabilité numérique, il est préférable de pondérer la contribution de chaque droite par le nombre de points qui la représente. La contribution R_i d'une droite \mathbf{l}_i est donc :

$$R_i = \frac{\sum_{\mathbf{x}_j \in \mathbf{l}_i} d(\mathbf{x}_j, \mathbf{l}_i)}{\text{Card}(\mathbf{x}_j)_{\mathbf{x}_j \in \mathbf{l}_i}}$$

Le résidu R de l'image s'écrit alors :

$$R = \sum_i R_i$$

4.1 Redressement des droites

Redresser les droites revient à minimiser le résidu de l'image. Pour ce faire, on va chercher des coefficients k_1, k_2, \dots, k_n de correction de distorsion radiale et les appliquer aux points \mathbf{x}_i . On obtient alors un nouvel ensemble de points $\hat{\mathbf{x}}_i$. Le but est de trouver des coefficients tels que le calcul du résidu R de l'image avec ces points $\hat{\mathbf{x}}_i$ soit minimal (ou au moins inférieur au résidu de l'image sans correction).

Pour résumer, on cherche les coefficients k_1, k_2, \dots, k_n qui, une fois appliqués sur les points \mathbf{x}_i minimisent le résidu de l'image. Il s'agit d'un problème non linéaire dont le mode opératoire est expliqué dans la partie 5

5 Systèmes non-linéaires

5.1 Les paramètres

Un phénomène est dit non linéaire lorsque des grandeurs caractéristiques du phénomène reliées entre elles ne varient pas proportionnellement l'une à l'autre. Il existe plusieurs sortes de problèmes non-linéaire et plusieurs façons de les traiter. La plupart des méthodes partent d'une solution initiale que l'on espère pas trop éloignée de la solution recherchée, puis affinent itérativement cette solution de départ pour se rapprocher de la solution optimale. Le problème non-linéaire que nous allons traiter peut s'écrire sous la forme $X = f(\mathbf{a}, \mathbf{b})$ où :

- f est la fonction que l'on souhaite traiter.
- On veut minimiser $\epsilon = |X - f(\mathbf{a}, \mathbf{b})|$, autrement dit, on veut faire tendre f vers X autant que possible.
- \mathbf{a} est un vecteur contenant toutes les variables que nous pouvons modifier pour faire tendre f vers X .
- \mathbf{b} est un vecteur contenant toutes les constantes représentant le système modéliser. Ainsi, l'évaluation de f nécessite les variables \mathbf{a} et les constantes du système \mathbf{b} .

Dans le cas de la distorsion radiale, le vecteur \mathbf{a} correspond aux coefficients k_1, k_2, \dots, k_n et, de façon optionnelle, au centre de la distorsion (x_c, y_c) . Le vecteur \mathbf{b} contient toutes les informations constantes du système, c'est à dire le nombre de droites sélectionnées dans l'image et pour chacune d'entre elles, le nombre et la liste des points qui la composent. Le vecteur \mathbf{b} contient aussi le degré maximum des coefficients k_1, k_2, \dots, k_n ainsi que le centre de la distorsion si l'on ne désire pas le faire varier. La fonction $f(\mathbf{a}, \mathbf{b})$ extrait les paramètres du vecteur \mathbf{b} et utilise les variables du vecteur \mathbf{a} pour calculer le résidu R de l'image. Comme on désire minimiser R , on peut poser $X = 0$.

5.2 La matrice jacobienne

La matrice jacobienne est la matrice des dérivées partielles du premier ordre d'une fonction vectorielle. Dans le système non-linéaire $f(\mathbf{a}, \mathbf{b})$, c'est la matrice J dont chaque élément est définie par :

$$J_{ij} = \frac{\partial f_i}{\partial a_j}$$

Dans notre cas, la fonction f renvoie un scalaire, la matrice jacobienne est donc une matrice à une seule ligne dont le j^{eme} élément s'écrit :

$$J_j = \frac{\partial f}{\partial a_j}$$

La matrice jacobienne permet d'estimer la variation du résidu de l'image R si l'on fait varier légèrement l'un des coefficients k_1, k_2, \dots, k_n .

En pratique, une dérivée partielle se calcule soit à partir de l'expression formelle de la dérivée de la fonction traitée si elle est connue, soit en l'estimant numériquement en faisant varier de δa_j le j^{eme} élément de \mathbf{a} :

$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_j \\ \vdots \\ a_n \end{pmatrix} \quad \hat{\mathbf{a}}_j = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_j + \delta a_j \\ \vdots \\ a_n \end{pmatrix}$$

L'élément J_j se calcule alors comme suit :

$$J_j = \frac{\partial f}{\partial a_j} \simeq \frac{f(\hat{\mathbf{a}}_j, \mathbf{b}) - f(\mathbf{a}, \mathbf{b})}{\delta a_j}$$

Dans la littérature, il est conseillé de choisir $\delta a_j = \max(\min(a_j \cdot 10^{-4}, 10^{-6}), 10^{-15})$. Ceci permet d'essayer de prendre une valeur de $\delta a_j = a_j \cdot 10^{-4}$ tout en imposant que δa_j soit compris entre 10^{-6} et 10^{-15} .

5.3 Le pas $\Delta \mathbf{a}$

Le but de l'étape suivante est d'utiliser la matrice jacobienne calculée précédemment afin de trouver la meilleur variation $\Delta \mathbf{a}$ de \mathbf{a} permettant de minimiser $\epsilon = |X - f(\mathbf{a}, \mathbf{b})|$. Les méthodes de résolution de systèmes non linéaires sont généralement basées sur l'hypothèse que la fonction f a localement un comportement linéaire. Elle peut donc être approximée en \mathbf{a} par :

$$f(\mathbf{a} + \Delta \mathbf{a}, \mathbf{b}) \simeq f(\mathbf{a}, \mathbf{b}) + J \Delta \mathbf{a}$$

Dans notre cas où $X = 0$, nous cherchons un $\Delta \mathbf{a}$ tel que $f(\mathbf{a} + \Delta \mathbf{a}, \mathbf{b}) = 0$. L'équation précédente nous donne alors :

$$f(\mathbf{a}, \mathbf{b}) + J\Delta \mathbf{a} = 0$$

soit

$$J\Delta \mathbf{a} = -f(\mathbf{a}, \mathbf{b})$$

Ce qui peut se résoudre par :

$$\Delta \mathbf{a} = -J^+ f(\mathbf{a}, \mathbf{b})$$

où $J^+ = (J^\top J)^{-1} J^\top$ est la matrice pseudo-inverse de J .

Un pas $\Delta \mathbf{a}$ est acceptable s'il satisfait effectivement $|f(\mathbf{a} + \Delta \mathbf{a}, \mathbf{b})| < |f(\mathbf{a}, \mathbf{b})|$, ce qui n'est pas nécessairement le cas selon f et \mathbf{a} . Une bonne valeur de $\Delta \mathbf{a}$ doit d'une part ne pas être trop petite sans quoi la résolution du système est trop longue et le risque de tomber dans un minimum local augmente. Cette valeur ne doit pas non plus être trop grande car le risque serait alors de dépasser le minimum global et de diverger. La solution à ce problème est de pondérer le pas $\Delta \mathbf{a}$ par une valeur λ tel qu'il soit ni trop petit, ni trop grand. On choisit d'abord une valeur initiale de λ . Dans la littérature, il est conseillé de choisir λ comme 10^{-3} fois la valeur moyenne des éléments de la diagonale de $N = J^\top J$. Le calcul de $\Delta \mathbf{a}$ se fait alors en résolvant à l'aide de la *SVD* le système suivant :

$$(J^\top J + \lambda Id)\Delta \mathbf{a} = -J^\top f(\mathbf{a}, \mathbf{b})$$

Si le $\Delta \mathbf{a}$ obtenu fait converger f , alors on met à jour \mathbf{a} et on divise λ par 10 pour que le pas de la prochaine itération soit plus grand. Si le $\Delta \mathbf{a}$ obtenu ne fait pas converger f , alors le pas était trop grand et on ré-écrit le système précédent avec une valeur de λ multipliée par 10. On répète ce processus jusqu'à ce qu'on obtienne un λ acceptable.

5.4 En pratique

La stabilité numérique des méthodes de résolution de systèmes non linéaires dépend très souvent de la qualité des conditions initiales. Dans notre cas, la résolution du système peut partir des conditions initiales suivantes :

- (x_c, y_c) est le centre de l'image.
- les coefficients k_1, k_2, \dots, k_n sont tous nuls.

La résolution d'un système linéaire est un processus itératif décrit dans l'algorithme 1.

6 Travail demandé

Vous devez réaliser par groupe de deux étudiants un programme en langage C++ à l'aide de la bibliothèque **OpenKraken**. Vous séparerez vos programmes dans des fichiers `.cpp` et `.hpp` qui seront compilés à l'aide d'un `makefile`. Votre programme devra compiler sans warning (mais avec l'option `-Wall`). Vous rendrez votre projet sous forme d'archive nommée `nom1_nom2_radial.tgz`. Cette archive doit générer un repertoire nommé `nom1_nom2_radial/` contenant votre projet.

Algorithme 1 : Résolution de systèmes non linéaires

Data : Un vecteur \mathbf{a} des variables à calculer, un vecteur \mathbf{b} des constantes du système et une fonction $f(\mathbf{a}, \mathbf{b})$ à minimiser.

Result : une valeur de \mathbf{a} telle que f soit minimal.

```
1  $J = [\partial f / \partial \mathbf{a}]$ 
2  $\lambda = (J^\top J) \cdot 10^{-3}$ 
3 foreach iteration do
4    $J = [\partial f / \partial \mathbf{a}]$ 
5   compteur = 0
6   accepté = false
7   repeat
8     résoudre :  $(J^\top J + \lambda Id)\Delta \mathbf{a} = -J^\top f(\mathbf{a}, \mathbf{b})$ 
9     if  $|f(\mathbf{a} + \Delta \mathbf{a}, \mathbf{b})| < |f(\mathbf{a}, \mathbf{b})|$  then
10       $\mathbf{a} = \mathbf{a} + \Delta \mathbf{a}$ 
11       $\lambda = \lambda / 10$ 
12      accepté = true
13    else
14       $\lambda = \lambda \times 10$ 
15    compteur = compteur + 1
16    if compteur > 100 then
17      return  $\mathbf{a}$ 
18  until accepté = true ;
19 return  $\mathbf{a}$ 
```

6.1 Validation de votre projet

Votre programme devra pouvoir charger une image et une liste de segments, eux-mêmes définis par une liste de points. Votre programme devra ensuite calculer les coefficients de correction de distorsion radiale, puis corriger l'image chargée. Votre programme devra au moins fonctionner pour une correction à deux coefficients k_1 et k_2 .

Afin de valider facilement votre code, vous créerez un répertoire `output/` dans lequel vous stockerez les images suivantes, en respectant les noms proposés :

- l'image non corrigée : `inputImage.jpg`.
- l'image non corrigée avec les segments de droite sélectionnés : `inputImageLines.jpg`.
- l'image corrigée : `correctedImage.jpg`.
- l'image corrigée avec les segments de droites corrigés : `correctedImageLines.jpg`.
- (optionnel) l'image corrigée sans "trous" : `correctedImageOptimal.jpg`.

La validation de votre méthode de résolution de systèmes non linéaires se fera grâce à un graphe de convergence de la fonction f et une étude de la variation de la variable λ .

6.2 Options de commandes et autres contraintes

Il est important de respecter les notations spécifiées dans ce sujet, notamment le nom des options détaillées ci-dessous. Pour commencer, votre exécutable s'appellera : `./radial`. Ensuite, les options suivantes sont obligatoires :

- `-h` pour afficher l'aide
- `-i [image].jpg` pour spécifier le nom de l'image à corriger.
- `-l [image]*.list` pour spécifier les noms des listes contenant les points.

Vous pouvez tout à fait rajouter d'autres options dont vous détaillerez la fonction dans l'aide.

Attention, votre programme ne doit pas contenir de `std::cin` ou de `scanf`.

Enfin, votre fonction de résolution de systèmes non linéaires prendra en paramètre la fonction à minimiser sous forme de pointeur de fonction. Son prototype sera :

```
void nonLinearSystemSolver(  
    kn::Vector<double> &a,  
    const kn::Vector<double> &b,  
    double (*costFunctionPtr)(const kn::Vector<double>&,const kn::Vector<double>&),  
    const unsigned int nbMaxIteration);
```

A noter que pour des questions de généricité, toutes les données constantes quelque soit leur type doivent être stockées sous la forme d'un vecteur `b` de `double`.

6.3 Formats de fichier

Afin de faciliter l'édition des segments de droites, vous pouvez utiliser un programme indépendant de sélection de points. Chaque segment sera stocké sous forme d'une liste de vecteurs (la bibliothèque **OpenKraken** propose cette fonctionnalité). Chaque segment de droite sera ainsi stocké dans un fichier et tous les fichiers porteront le même nom auquel sera ajouté en suffixe un numéro (ex : `myImageLine05.list`).

6.4 Rapport

Vous fournirez en version papier et en version électronique un rapport de 10 pages maximum. **Consacrez suffisamment de temps au rapport car il représente une bonne partie de la note finale.** Essayez de respecter au mieux les directives suivantes :

- Ne perdez pas de temps à réexpliquer le sujet du projet, l'enseignant le connaît déjà, faites seulement un bref résumé de quelques lignes. De manière plus générale, ne détaillez pas des méthodes déjà expliquées dans l'énoncé à moins que vous les ayez modifiées.
- Un rapport sert surtout à montrer comment vous avez fait face aux problèmes (d'ordre algorithmique). Certains problèmes sont connus (on en parle dans l'énoncé), d'autres sont imprévus. Montrez que vous les avez remarqués et compris. Donnez la liste des solutions à ce problème et indiquez votre choix. Justifiez votre choix (vous avez le droit de dire que c'est la méthode la plus facile à coder).
- Il ne doit figurer aucune ligne de code dans votre rapport. Un rapport n'est pas un listing de votre programme où vous détaillez chaque fonction. Vous devez par contre détailler vos structures de données et mettre du pseudocode pour expliquer vos choix algorithmiques.
Il est autorisé d'utiliser des "raccourcis" tels que "initialiser le tableau `tab` à 0" plutôt que de détailler la boucle faisant la même chose.
- n'hésitez pas à mettre des images dans votre rapport pour illustrer vos propos et vos résultats.
- Enfin, il est **très important** de faire la liste de ce que vous avez fait, de ce qui fonctionne correctement et la liste des dysfonctionnements. Précisez quand il s'agit d'options que vous avez rajoutées en plus de ce qui était demandé.

7 Pour finir

Vous pouvez laisser libre cours à votre imagination, toute amélioration sera la bienvenue. Vous trouverez quelques informations complémentaires à l'adresse :

<http://www-igm.univ-mlv.fr/~vnozick/>

Bon courage.