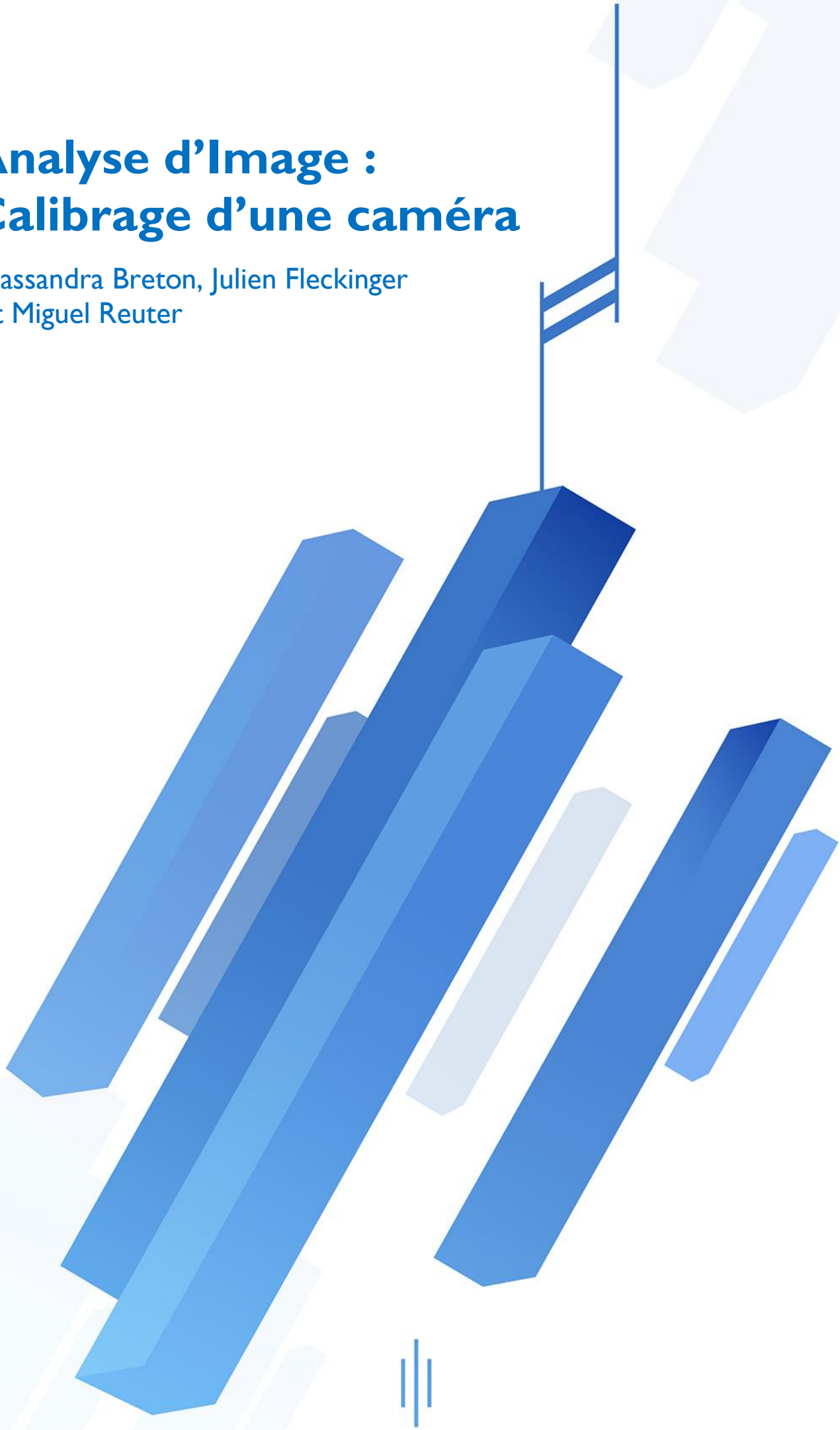
Analyse d'Image : Calibrage d'une caméra

Cassandra Breton, Julien Fleckinger
et Miguel Reuter



Projet : calibrage d’une caméra

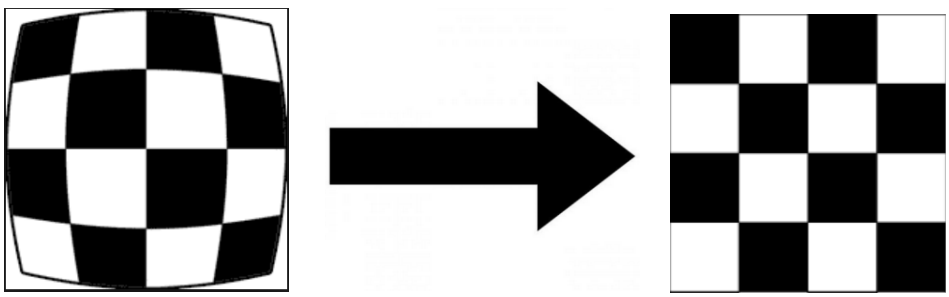
Sommaire

Introduction

L'objectif du TP est de faire de calibration de caméra. Cette opération consiste à modéliser le processus de formation des images, c’est-à-dire, de déterminer la relation entre un point dans l'espace et un son point associé dans l'image de la caméra. Le TP a été réalisé en C++ à l'aide de la bibliothèque OpenCV. Dans le cadre de ce TP, nous nous sommes arrêtés à la correction de l’image obtenue.

But

Nous cherchons à effectuer l'opération de calibration d'une caméra, pour ce faire, nous avons pris des captures d’un damier afin de corriger les déformations observables sur les images de ce damier, obtenues avec une caméra.



Outre les paramètres intrinsèques de la caméra comme la focale, les coefficients de distorsion sont à trouver pour la calibration. Dans notre cas, 5 coefficients de distorsion sont à déterminer.

Les paramètres nécessaires aux calculs de la distorsion des images :

- Le nombre d'images obtenues,
- Le nombre de cases en hauteur et largeur du damier,
- La taille réelle des cases,
- Le nombres de points d'intérêt.

Résultats obtenus

Figure 1 : Image initiale 1

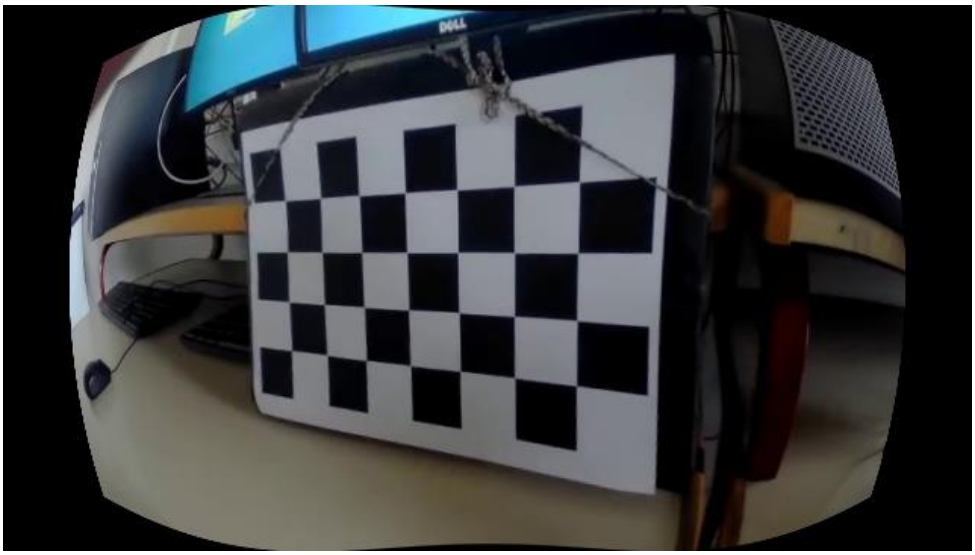


Figure 2 : Image résultat de l'image 1

Figure 3 : Image initiale 2

Figure 4 : Image résultat de l'image 2

Notre cas

Pour commencer, nous avons créé un panel de 25 images d'un damier de 8x5 cases, dont chaque case fait 35.25x35.25mm. L'appareil utilisé pour les prises de vue crée un effet de « fisheye », ainsi les corrections que nous obtenons se voient beaucoup mieux.

Nous obtenons 7x4 points d'intérêts par image (correspondants aux intersections du damier). Pour chaque prise, nous avons fait varier la position du damier par rapport à la caméra.

L'intégralité du code se trouve dans « Reconstruction3D\src\main.cpp ».

Initialisation des paramètres du programme

Cette partie va détailler les paramètres nécessaires au paramétrage de la caméra, demandés à l'utilisateur. Ils se trouvent dans le fichier « Reconstruction3D\conf\in_VID5.xml » et « FISHEYE.xml ».

BoardSize_Width
Le nombre de points d'intérêts que contient le damier présent sur les images, en largeur.
Valeur = 7

BoardSize_Height
Le nombre de points d'intérêts que contient le damier présent sur les images, en hauteur.
Valeur = 4

Square_Size
La taille des cases du damier en millimètres. **Valeur =** 35.25

Input
Contient le chemin vers le fichier de configuration où se trouve la liste des images à utiliser.
Valeur = «Reconstruction3D\ conf\FISHEYE.xml »

Input_Delay
Délai entre chaque changement d'image en millisecondes. **Valeur =** 100

Calibrate_FixAspectRatio
Permet à l'aspect ratio d'être le même pour fx et fy que l'input cameraMatrix. **Valeur =** 1

Calibrate_AssumeZeroTangentialDistortion
Oblige les coefficients de distorsion tangentielle à être égale à 0. **Valeur =** 0

Calibrate_FixPrincipalPointAtTheCenter
Permet de corriger le point d'intérêt principale. **Valeur =** 1

Write_outputFileName
Fichier où sera enregistré les paramètres de calibration calculés. **Valeur =**
« out_camera_data.xml »

Fonctionnement du code

Cette partie traite du fonctionnement et de la structure du code. Celui-ci se trouve dans « Reconstruction3D\src\main.cpp ». Le programme nécessite la librairie sdl2 et glew pour fonctionner.

Comme dit dans la patrie « But », le programme a besoin du d'images utilisées, du nombre de cases du damier, du nombre de points d'intérêts qui doivent être trouvés et de la taille réelle des cases.
Il lance ensuite une boucle sur toutes les images pour trouver ces points d'intérêts qui sont alors enregistrés.

Pour le calcul des coefficients, on se base sur la position de points d'intérêts du damier. En effet, les positions des coins des cases du damier sur les photos prises (donc déformées par l'effet de distorsion) vont être comparées à celle d'une image idéale, sans distorsion et en se basant uniquement sur la géométrie connue du damier. On cherche à trouver la correction à effectuer sur les images avec distorsion, pour avoir des images de damier "idéales", c'est à dire sans distorsion.
L'application va évaluer les paramètres de distorsion de manière à minimiser l'erreur entre les positions des points d'intérêts des images corrigées et les positions sur la géométrie connue du damier.

Enfin, on corrige les images d'origines en leur appliquant la matrice de distorsion trouvée et on les affiche.

Après lancement du programme, les indices des images résultats « corrected » sont les mêmes que ceux des images desquels les corrections ont été calculés. Exemple :
« corrected_0 » est le résultat de la correction de l'image « fish0 ».

Descriptions des méthodes utilisées dans le code

Settings::read(const FileNode & f)
Permet de lire les données enregistrées dans le fichier f de paramétrage et utilise interprate() pour initialisé le programme avec.

Settings::interprate()
Fait des tests de validité des paramètres donnés en entrée et initialise le programme.

Settings::Mat nextImage()
Donne la matrice contenant l'image suivante.

bool Settings::readStringList(const string& f, vector<string>& l)
Permet de lire le fichier contenant la liste des images f et met les chemins des images dans la liste l.

bool Settings::isListOfImages(const string& f)
Vérifie que le fichier fichier f (contenant la liste des chemins d'images) est accepté par le programme.
read (const FileNode& f, Settings& s, const Settings& default = Settings())
Initialise la structure Settings s, soit avec des données récupérées dans le FileNode f, soit avec une structure par défaut.

static double computeReprojectionErrors(const vector<vector<Point3f> >& points, const vector<vector<Point2f> >& p_image, const vector<Mat>& rvecs, const vector<Mat>& tvecs, const Mat& m_camera, const Mat& distCoeffs, <float>& perViewErrors)
Calcul l'erreur de reprojection obtenue avec les données des images.

calcBoardCornerPositions(Size s_boardSize, float s_square, vector<Point3f>& corners)
Trouve les points d'intérêts sur l'image.

static bool runCalibration(Settings& s, Size& s_image, Mat& m_camera, Mat& distCoeffs, vector<vector<Point2f> > p_image, vector<Mat>& rvecs, vector<Mat>& tvecs, vector<float>& reprojErrs, double& totalAvgErr)
Calcul les coefficients de distorsion, l'erreur et trouve les paramètres intrinsèque de la caméra.

saveCameraParams(Settings& s, Size& s_image, Mat& m_camera, Mat& distCoeffs, const vector<Mat>& rvecs, const vector<Mat>& tvecs, const vector<float>& reprojErrs, const vector<vector<Point2f> >& imagePoints, double totalAvgErr)
Permet d'enregistrer les paramètres calculés de la caméra dans un fichier afin de pouvoir les réutiliser.

bool runCalibrationAndSave(Settings& s, Size imageSize, Mat& cameraMatrix, Mat& distCoeffs,vector<vector<Point2f> > imagePoints)
Lance runCalibration() et vérifie si les résultats sont bons pour savoir s'il est nécessaire d'enregistrer les paramètres calculés dans un fichier.

Cas limites

On peut voir que les bords de l'image ne sont pas correctement corrigés. Ceci doit être dû au fait que les points d'intérêts se trouve centrés sur l'image et éloignés des bords. La distorsion n'étant pas linéaire en tout point de l'image, la correction n'est donc pas bonne pour les bords.

Le grand nombre d'images (25) fait que l'on obtient une erreur assez élevée (aux alentours de 1.1) mais toutes les images corrigées montrent de bons résultats.
Les prises de vues qui différent fortement sont sûrement à l'origine de cette forte erreur. Cependant, trop d'images similaires risquerait de créer trop d'équations similaires et donc causerais des problèmes de résolution ce qui empêcherais la calibration de se terminer.

Figure 5 : Image initiale 3

Figure 6 : Image résultat de l'image 3 – les bords du damier sont mal corrigés

Conclusion

En conclusion, l'objectif du TP a été de faire de la calibration de caméra. Globalement (visuellement, en tout cas), les images corrigées semblent correctes et les déformations sont nettement diminuées. Le résultat n'est cependant pas parfait, sans doute à cause d'une méthode de détermination des coefficients de distorsion imparfaite ou alors un modèle trop simpliste utilisé.