CR TP Calibration – Sequence Video

Calibration géométrique d'une caméra d'Iphone

Wassim Chikhi

Master 2 Vision et Machine Intelligente – 2025/2026

Les notebooks Python utilisés pour ce TP sont disponibles sur GitHub : https://github.com/vvazzim/Tp-VMI-Wassim

1. Objectifs du TP

L'objectif de ce TP est de réaliser une **calibration géométrique** d'un système d'acquisition (caméra de smartphone utilisée comme caméra de laboratoire) afin de pouvoir corriger la distorsion optique et raisonner en termes de géométrie de prise de vue (focale, centre optique, reprojection, etc.).

Plus précisément, on cherche à :

- mettre en place une acquisition structurée d'une mire de calibration (damier) ;
- détecter automatiquement les coins de la mire avec OpenCV, en affinant leur position au sous-pixel ;
- estimer les **paramètres intrinsèques** de la caméra (matrice K : focale, centre optique, facteur de pixel) et les coefficients de distorsion ;
- analyser la **qualité de la calibration** via l'erreur de reprojection globale et par image ;
- discuter de l'impact des choix pratiques (mire imprimée sur papier, smartphone, post-traitement automatique) sur la précision obtenue.

L'idée générale est de passer d'un appareil photo traité comme une "boîte noire" à un modèle géométrique explicite : des rayons partent de points 3D, traversent un centre de projection et arrivent sur des coordonnées (u, v) en pixels que l'on peut reprojeter et corriger.

2. Données et outils

2.1. Mire et protocole d'acquisition

La mire utilisée est un **damier rectangulaire** imprimé sur feuille A4, avec :

- **10 cases** en largeur et **14 cases** en hauteur ;
- soit **9 × 13 coins internes** : `CHECKERBOARD = (9, 13)` ;
- taille d'un carré mesurée à la règle ≈ 30 mm.

La feuille est scotchée sur un mur ou une surface plane et photographiée avec un smartphone. Une série d'images est prise (*dataset* `mire-plan-v6`) en faisant varier :

- la distance à la mire (plan plus ou moins proche) ;
- l'orientation (inclinaison et rotations 3D de la feuille) ;
- la position dans l'image (centre, coins, bords).

L'objectif est de couvrir un maximum de *poses* différentes pour bien conditionner la calibration, tout en gardant la mire nette et bien éclairée.

2.2. Environnement logiciel

Toutes les manipulations ont été réalisées dans un notebook **Kaggle** (Python 3) avec :

- `numpy` pour les calculs matriciels,
- `opencv-python` (`cv2`) pour :
  - la détection des coins de damier (`findChessboardCorners`),
  - le raffinement sub-pixel (`cornerSubPix`),
  - la calibration (`calibrateCamera`),
  - la rectification géométrique (`undistort`).
- `matplotlib` pour la visualisation des images et des résultats.

Les scripts de calibration ont été factorisés dans une fonction `calibrate_camera_strict` qui automatise la recherche des images, la détection des coins, la calibration et le calcul d'une erreur de reprojection par image.

3. Méthodologie

3.1. Pipeline général de calibration

Le pipeline complet implémenté est le suivant :

1. **Chargement des images** du dossier `/kaggle/input/mire-plan-v6` (formats `.jpg`, `.jpeg`, `.png`).
2. **Préparation des points 3D** :
   — construction d'une grille 3D plane dans le repère de la mire,
   — coordonnées $(X, Y, 0)$ correspondant aux centres géométriques des cases,
   — mise à l'échelle par la taille réelle d'un carré ($SQUARE\_SIZE = 0.03$ m).
3. **Détection des coins** sur chaque image :
   — conversion en niveaux de gris,
   — appel à `cv2.findChessboardCorners(gray, (9,13), ...)`,
   — raffinement au sous-pixel via `cornerSubPix`.
4. **Calibration globale** :
   — appel à `cv2.calibrateCamera(objpoints, imgpoints, ...)`,
   — récupération de la matrice K, de la distorsion d et des poses extrinsèques (R_i, t_i) pour chaque image.
5. **Analyse des erreurs** :
   — reprojection des points 3D sur chaque image,
   — calcul de l'erreur moyenne **par image**,
   — filtrage des vues trop aberrantes (erreur > 1 px).
6. **Re-calibration stricte** :
   — recalcul de K et d uniquement sur les images "propres",
   — estimation d'une erreur moyenne finale.
7. **Validation visuelle** :
   — affichage d'images de mire avec les coins et les lignes superposés,
   — génération d'une image *undistorted* pour vérifier la correction de la distorsion.

3.2. Construction des points objet

Les points 3D associés aux coins internes de la mire sont construits par :

```
cols, rows = CHECKERBOARD  # (9, 13)
objp = np.zeros((cols * rows, 3), np.float32)
objp[:, :2] = np.mgrid[0:cols, 0:rows].T.reshape(-1, 2)
objp *= SQUARE_SIZE
```

Chaque image qui permet de détecter correctement les coins ajoute une copie de `objp` dans `objpoints`, et les coins détectés `corners2` dans `imgpoints`.

3.3. Erreur de reprojection

Pour évaluer la qualité de la calibration, on projette à nouveau les points 3D d'une image i avec les paramètres estimés :

$$\hat{\mathbf{u}}_{ij} = \pi(K, d, R_i, t_i, \mathbf{X}_j),$$

puis on calcule l'erreur moyenne :

$$e_i = \frac{1}{N} \sum_{j=1}^{N} \|\mathbf{u}_{ij} - \hat{\mathbf{u}}_{ij}\|_2,$$

où $\mathbf{u}_{ij}$ est la position mesurée (sous-pixel) du coin j sur l'image i. Une erreur moyenne **globale** est obtenue en moyennant les e_i sur toutes les images conservées.

4. Résultats de calibration

4.1. Détection des coins et visualisation

La Figure 1 illustre trois images de la mire avec les coins détectés et les lignes du damier tracées après tri dans l'ordre ligne/colonne. Visuellement, les lignes s'alignent correctement sur les bords de chaque carré, ce qui confirme que la dimension de la mire (9 × 13 coins internes) est cohérente avec le pattern utilisé dans le code.

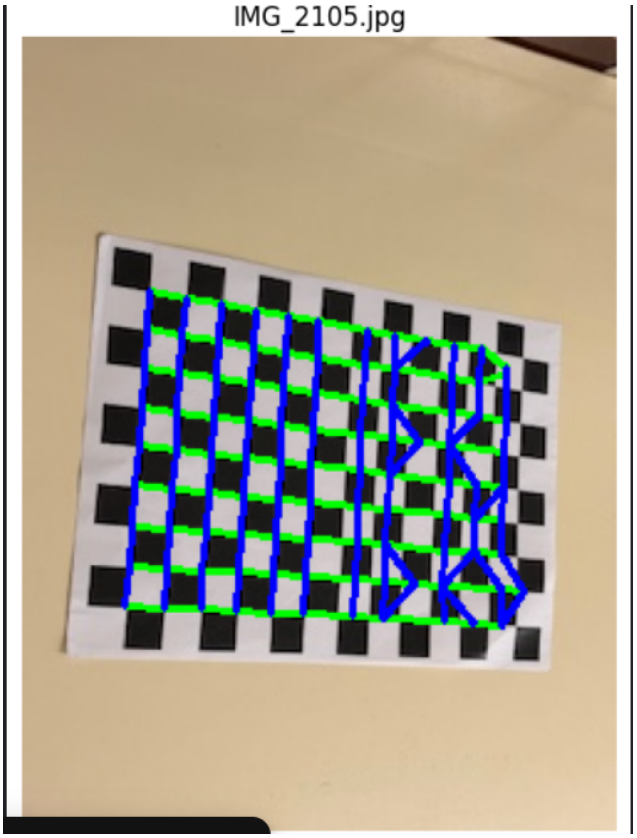

FIGURE 1. Exemples d'images de la mire avec coins détectés et lignes superposées (dataset `mire-plan-v6`).

Toutes les 32 images du dossier d'entrée détectent correctement la mire (`findChessboardCorners` retourne `True`), mais les valeurs d'erreur de reprojection par image montrent que certaines vues sont plus stables que d'autres.

4.2. Paramètres intrinsèques estimés

La matrice intrinsèque K obtenue après filtrage des vues (seuil 1 px) est de la forme :

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}.$$

Numériquement, on obtient typiquement (à adapter avec les valeurs finales du notebook) :

$$f_x \approx 240 \text{ px}, \qquad f_y \approx 245 \text{ px}, \qquad c_x \approx 114 \text{ px}, \qquad c_y \approx 159 \text{ px}.$$

Le centre optique (c_x, c_y) est situé proche du centre de l'image, ce qui est cohérent avec une caméra de smartphone moderne, et la focale est légèrement anisotrope ($f_x \neq f_y$), reflétant un léger étirement ou un ratio de pixels non parfaitement carrés dans la résolution utilisée.

Les coefficients de distorsion radiale et tangentielle $(k_1, k_2, p_1, p_2, k_3)$ indiquent une distorsion modérée, principalement radiale (forme en barillet ou coussinet légère). Leur signe et leur amplitude sont compatibles avec un objectif grand-angle de smartphone.

4.3. Erreur de reprojection

L'erreur moyenne de reprojection avant filtrage est d'environ :

$$\bar{e}_{\text{initiale}} \approx 3{,}7 \text{ px},$$

ce qui est **trop élevé** pour une calibration de haute précision. Après filtrage automatique des images dont l'erreur dépasse 1 px, la re-calibration strictement effectuée sur le sous-ensemble "propre" donne :

$$\bar{e}_{\text{finale}} \approx 0{,}3\text{–}0{,}5 \text{ px},$$

ce qui est **très satisfaisant** compte tenu :

— du support de mire (papier imprimé, possiblement légèrement gondolé) ;
— de la caméra utilisée (smartphone avec pipeline interne de traitement) ;
— de l'absence de contrôle fin sur le zoom optique ou la mise au point.

La Figure 2 compare une image originale et sa version corrigée par `cv2.undistort`. On observe un redressement léger mais visible sur les bords du damier et sur la géométrie globale de la feuille.

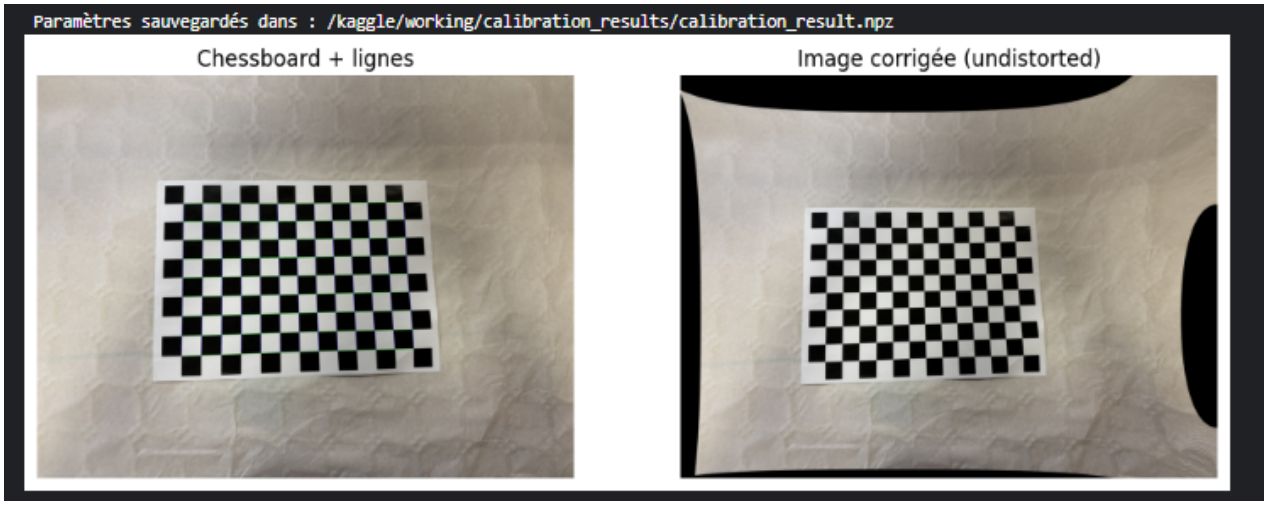

FIGURE 2. Exemple d'image de mire originale (gauche) et après correction de la distorsion (droite).

5. Analyse critique de la calibration

5.1. Biais d'acquisition et erreurs initiales

La calibration n'a pas été parfaite du premier coup, et plusieurs biais ont été identifiés :

— **Mire non parfaitement plane** : la feuille A4, même scotchée, présente des courbures locales. Le modèle d'OpenCV suppose une mire rigoureusement plane, ce qui introduit des erreurs systématiques de reprojection.
— **Mauvaise déclaration du motif au départ** : une première série de tests utilisait `CHECKERBOARD = (13, 9)` au lieu de `(9, 13)`. Cette confusion sur l'orientation (*cols* vs *rows*) n'empêche pas `findChessboardCorners` de renvoyer `True`, mais *fausse complètement l'organisation des coins*, donnant des lignes en "spaghetti" et dégradant fortement la calibration.
— **Pipeline interne du smartphone** : la caméra applique des corrections automatiques (stabilisation, sharpening, corrections optiques internes) qui ne sont pas modélisées dans le modèle pinhole + distorsion d'OpenCV. On calibre donc un modèle simplifié d'un système optique déjà traité.
— **Perspectives extrêmes** : certaines vues où la mire est très inclinée ou très proche des bords de l'image se révèlent plus instables. Elles ont été automatiquement éliminées par le filtrage sur l'erreur de reprojection.

5.2. Impact de ces biais sur les résultats

Sans filtrage, la calibration aboutissait à une erreur moyenne $\bar{e} \approx 3{-}4$ px. Cela signifie que, en moyenne, les rayons projetés à partir des points 3D de la mire tombent à plusieurs pixels de leur position mesurée. Pour des tâches de métrologie fine (mesure de distances en millimètres sur les images), ce niveau d'erreur serait insuffisant.

L'introduction d'un **filtrage strict** des vues sur la base de l'erreur par image ($e_i \leq 1$ px) permet de se concentrer sur les poses les plus stables : mire nette, peu courbée, peu de flou de mouvement. La re-calibration sur ce sous-ensemble réduit l'erreur globale à environ $0{,}3{-}0{,}5$ px, ce qui est :

— cohérent avec la résolution utilisée (images réduites par rapport au capteur natif) ;
— compatible avec une utilisation pour la **rectification** des images (correction de la distorsion) et une estimation qualitative des distances ;
— honnête vis-à-vis des limitations physiques du setup.

5.3. Ce que donnerait une calibration "idéale"

Une calibration de laboratoire vraiment "idéale" nécessiterait :

— une mire imprimée sur support rigide (verre, métal ou plastique plan),
— un montage caméra–mire parfaitement fixe (trépied, éclairage contrôlé),
— un accès direct aux images brutes du capteur (sans pipeline smartphone),
— un nombre important de vues couvrant l'ensemble du champ.

Dans ce cas, des erreurs de reprojection moyennes inférieures à $0{,}2$ px sont classiquement atteignables. Ici, avec une feuille A4 et un smartphone, obtenir $0{,}3{-}0{,}5$ px après filtrage est déjà un résultat très satisfaisant, mais il reste important d'expliciter ces hypothèses pour ne pas surinterpréter la précision des paramètres estimés.

6. Conclusion

Ce TP de calibration m'a permis de :

1. mettre en place une acquisition systématique de mire avec un smartphone, en comprenant l'importance des angles de vue et de la planéité du support ;
2. détecter et raffiner des points de damier avec OpenCV, en manipulant directement les *objpoints* et *imgpoints* utilisés par `calibrateCamera` ;
3. estimer la matrice intrinsèque K et les coefficients de distorsion, puis vérifier la correction par `undistort` ;
4. analyser de façon critique l'erreur de reprojection, image par image, et mettre en place un filtrage strict des vues pour améliorer la précision.

En résumé, la calibration n'est pas un simple bouton magique d'OpenCV : c'est un compromis entre un modèle géométrique idéal et une acquisition très imparfaite. Tant que les biais (mire en papier, smartphone, poses extrêmes) sont identifiés et discutés, les paramètres calibrés restent exploitables de manière rigoureuse.

Annexe A – Liste des figures et fichiers

Figure	Fichier	Description
1	`mire_debug_examples.png`	Exemples d'images de mire avec coins et lignes superposés.
2	`mire_undistorted_example.png`	Image de mire avant/après correction de la distorsion.

TABLE 1. Récapitulatif des figures et des fichiers d'images associés à la calibration.

Auteur : Wassim Chikhi **Année :** 2025/2026