

# Vision par ordinateur: Calibration de Caméra et géométrie épipolaire

**Sébastien Roy**  
**Jean-Philippe Tardif**

Département d'Informatique et de recherche opérationnelle  
Université de Montréal

Université   
de Montréal

Hiver 2007

# Au programme

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Définition

## Calibration

Retrouver les paramètres internes de la caméra. Parfois, ce terme inclus aussi les paramètres externes.

## Calcul de pose

Calculer les paramètres externes. On suppose souvent que les paramètres internes sont connus.

## Plus généralement

On cherche à associer à chaque pixel un rayon d'échantillonnage de l'espace

- point de départ
- direction

Ceci nous est donnée par les paramètres complets de la caméra

# Calibration

Type de calibration :

- Avec un objet 3D connu
- Avec d'images de plans connus (points, lignes, angle entre les lignes)
- À partir de mouvements de la caméra connus

Grandes étapes

- Identifier la projection des éléments (points, lignes, etc) dans les images
- Minimiser une fonction des paramètres et des entités

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D**
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Idée

## Objet connu

- Correspondances : points/lignes 3D  $\leftrightarrow$  leur image 2D
- Points 3D doivent être en position générale (non-planaire)

Exemple :

- Deux plans avec des damiers
- Dodécaèdre (nous en avons un au laboratoire)

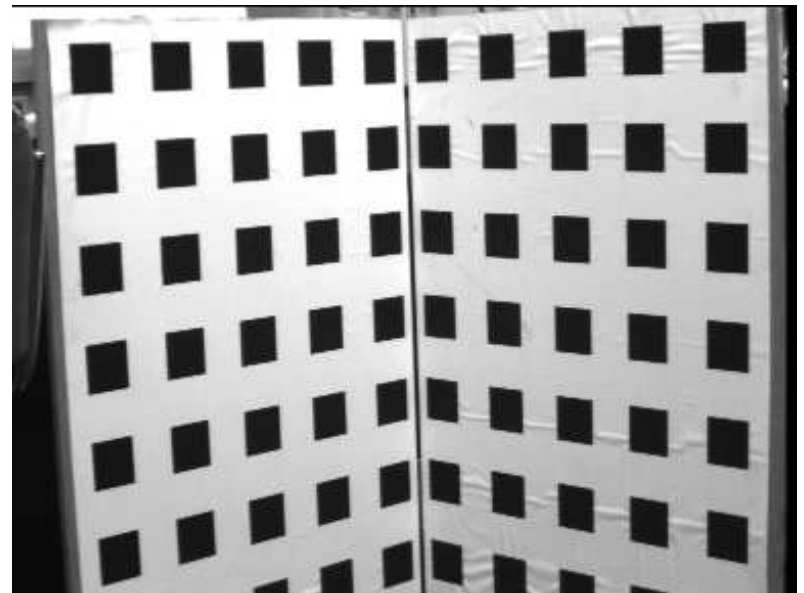




# Première étape

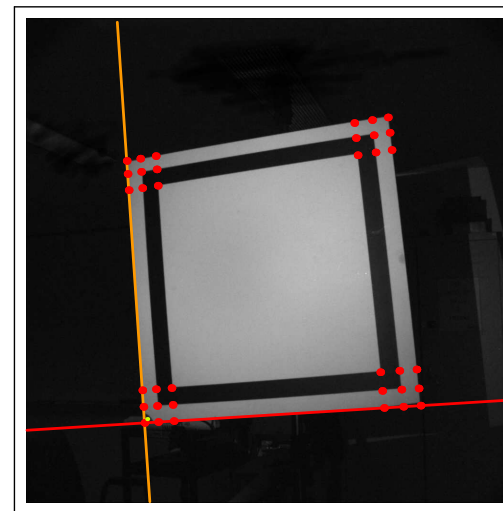
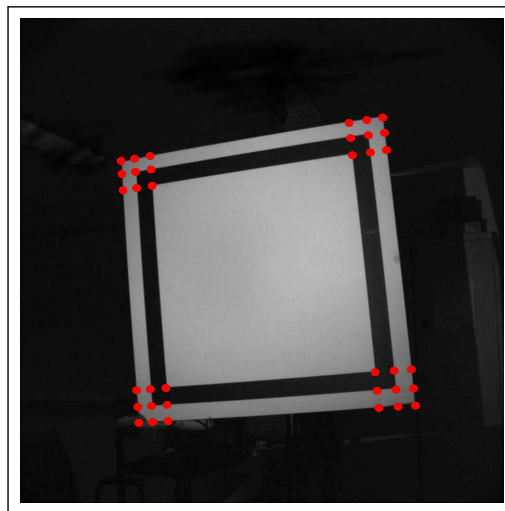
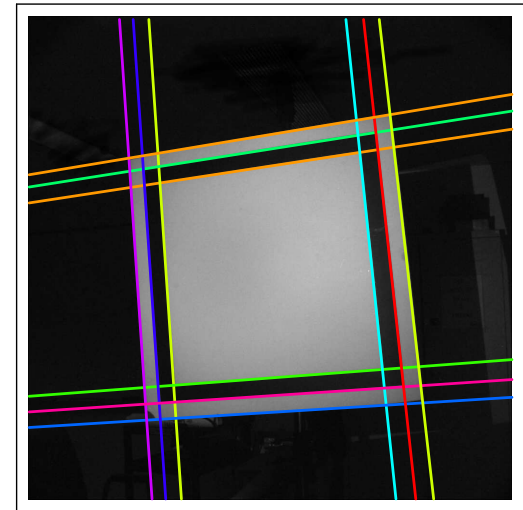
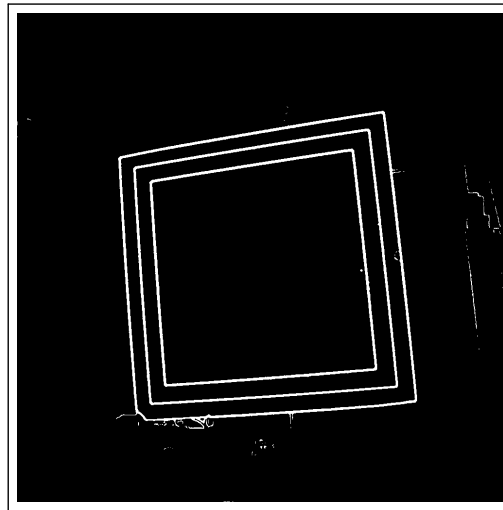
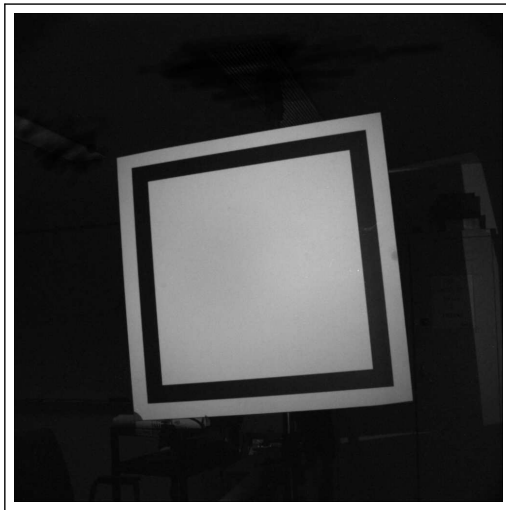
Avec une grille de calibration :

- Trouver des points saillants
- Trouver les bordures des carrés et trouver les lignes
- Intersection de ces lignes pour avoir des coordonnées sous-pixels
- Retrouver automatiquement la configuration des points (i.e. leur coordonnée)



# Première étape : un exemple

- Processus idéalement automatique
- Très souvent semi-automatique (comme OpenCV)



# Modèle perspective : Formulation linéaire

*Resectioning* Deux relations utiles :

- Pour un point 3D :

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \times \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{pmatrix} = \mathbf{0}$$

avec une eq. redondante

- Pour une ligne 3D formée de deux points  $Q_i, I$ , sa projection dans l'image

$$(l_1 \quad l_2 \quad l_3) \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{pmatrix} Q_{i1} \\ Q_{i2} \\ Q_{i3} \\ Q_{i4} \end{pmatrix} = 0$$

- Système d'équations en  $\mathbf{m}$ , le vecteur des  $m_{ij}$
- Résolution avec la SVD, donc la contrainte  $\|\mathbf{m}\| = 1$
- Minimisation de l'erreur algébrique

# Décomposition de $\mathcal{M}$

**Méthode indirecte :**

$$\begin{aligned}\mathcal{M} &= [\mathcal{H} \quad \mathbf{m}_4] \\ &= \mathcal{H} [\mathbf{I} \quad \mathcal{H}^{-1}\mathbf{m}_4] \\ &= \mathcal{H} [\mathbf{I} \quad -\hat{\mathbf{C}}]\end{aligned}$$

- donc  $\mathcal{H} = \mathcal{K}\mathcal{R}$
- Décomposition RQ de  $\mathcal{H}$

# Décomposition de $\mathcal{M}$

**Méthode directe :**

$$\rho \mathcal{H} = \rho \begin{bmatrix} \mathbf{h}_1^\top \\ \mathbf{h}_2^\top \\ \mathbf{h}_3^\top \end{bmatrix} = \begin{bmatrix} \alpha \mathbf{r}_1^\top + s \mathbf{r}_2^\top + \mathbf{r}_3^\top u_0 \\ \beta \mathbf{r}_2^\top + \mathbf{r}_3^\top v_0 \\ \mathbf{r}_3^\top \end{bmatrix} = \mathcal{K} \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix}$$

donc :

- $\rho = \pm 1 / \|\mathbf{h}_3\|$
- $\mathbf{r}_3 = \rho \mathbf{h}_3$
- $\rho \mathbf{h}_2^\top \rho \mathbf{h}_3 = v_0$
- ...

Voir Forsyth-Ponce pour le reste

# Solution optimale

Minimisation de l'erreur de reprojection

$$\sum_i \text{dist}^2(\mathbf{p}_i, \mathcal{M}\mathbf{P}_i)$$

Si on a des contrainte sur  $\mathcal{K}$  :

$$\sum_i \text{dist}^2(\mathbf{p}_i, \mathcal{K}\mathcal{R} \begin{bmatrix} \mathbf{I} & \mathbf{t} \end{bmatrix} \mathbf{P}_i)$$

- Paramétrisation de  $\mathcal{K}$  et  $\mathcal{R}$   
(Plus loin dans ce cours)

# Caméra affine

$$\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} - \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ 1 \end{pmatrix} = \mathbf{p} - \begin{bmatrix} \mathbf{m}_1^\top \\ \mathbf{m}_2^\top \end{bmatrix} \mathbf{P} = \mathbf{0}$$

et le système est donc

$$\begin{bmatrix} \mathbf{P}^\top & \mathbf{0}^\top \\ \mathbf{0}^\top & \mathbf{P}^\top \end{bmatrix} \begin{pmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \end{pmatrix} = \mathbf{p}$$

- On résout avec la pseudo-inverse
- $\mathcal{M}$  n'est pas définie à un facteur d'échelle
- Ceci minimise **l'erreur de reprojection**

# Avantages/Inconvénients

- Simplicité
- Une seule vue, combiner plusieurs vues pour les paramètres internes ?
- Difficile d'avoir beaucoup de points
- Identifier les points automatiquement



# À réfléchir/essayer

- Forcer certaines contraintes
  - Point principal connu
  - Focal identique en  $X$  et  $Y$  (ratio d'image de 1)
  - Aucun skew
- Solution : formuler des système d'équations spécifiques, si possible

ex :

- Méthode Trucco & Verri, point principal connu
- Calibrage planaire
- Forcer les contraintes par après

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire**
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Idée

- Image d'un plan
- Géométrie euclidienne connue
- Paramètres internes calculés à partir de plusieurs vues
- Pose par rapport au plan pour chaque vue

## Plan euclidien

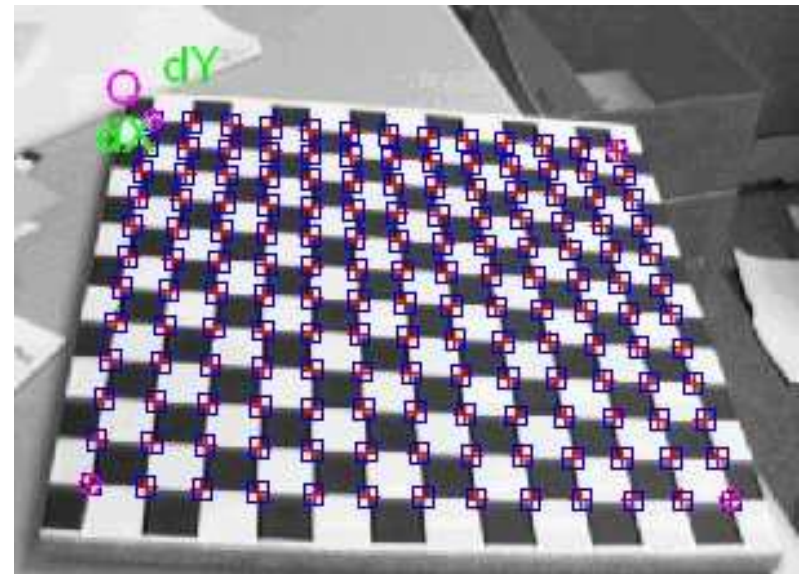
On connaît un système d'axe euclidien dans ce plan. On connaît par exemple :

- l'angle entre les lignes
- la distance entre les points
- le fait que l'échiquier est constitué de "carrés"

# Première étape

Avec une grille de calibration :

- Trouver des points saillants
- Retrouver automatiquement la configuration des points (i.e. leur coordonnée)
  - Pas si difficile



# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire**
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Modèle de projection $\rightarrow$ transformation

Rappel :

- Projection  $3 \times 4$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathcal{M}\mathbf{P} = \mathcal{K}\mathcal{R}_{3 \times 3}[\mathbf{I}_{3 \times 3} | \mathbf{t}] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- Projection d'un plan en  $Z = 0$

$$\mathcal{K}\mathcal{R}[\mathbf{I}_{3 \times 3} | \mathbf{t}] \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = \mathcal{K}\mathcal{R} \begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t} \\ 0 & 0 & \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = \mathcal{H} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

# Conique absolue $\Omega_\infty$

## Définition

Conique de points complexes dans la plan infini  $\pi_\infty = (0, 0, 0, 1)^\top$   
**Seulement** pour les points de la forme  $(x, y, z, 0)^\top$  et

$$(x, y, z) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 0, \quad h = 0$$

(en euclidien)

# Projection de $\Omega_\infty$

- Projection des points sur  $\pi_\infty$

$$\mathbf{p} = \mathcal{K}\mathcal{R}[\mathbf{I}|\mathbf{t}] \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix} = \mathcal{K}\mathcal{R} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

- seuls l'orientation de la caméra et ses paramètres internes influencent la projection de ces points

Analogie : La position du soleil ou des étoiles ne changent pas si on translate, seulement si on change de d'orientation.



# Projection de $\Omega_\infty$

Comme  $\Omega_\infty$  est dans  $\pi_\infty$  on peut travailler avec les plans projectifs

- $\pi_\infty$
- notre image de caméra

Rappel : transformation d'une conique :

$$C' \propto \mathcal{H}^{-\top} C \mathcal{H}^{-1}$$

- Posons  $\mathcal{H} = \mathcal{K}\mathcal{R}$
- Appliquons à  $\Omega_\infty = \mathbf{I}$

$$(\mathcal{K}\mathcal{R})^{-\top} \mathbf{I} (\mathcal{K}\mathcal{R})^{-1} = \mathcal{K}^{-\top} \mathcal{R} \mathbf{I} \mathcal{R}^{-1} \mathcal{K}^{-1} = \mathcal{K}^{-\top} \mathcal{K}^{-1}$$

Image de la Conique Absolue (*Image of the Absolute Conic (IAC)*)

$$\omega = \mathcal{K}^{-\top} \mathcal{K}^{-1}$$

## Détails sur $\omega$

$$\text{Pour } \mathcal{K} = \begin{bmatrix} f & 0 & u \\ 0 & f\alpha & v \\ 0 & 0 & 1 \end{bmatrix}$$

Au long,

$$\omega \propto \begin{bmatrix} \alpha^2 & 0 & -\alpha^2 u_0 \\ 0 & 1 & -v_0 \\ -\alpha^2 u_0 & -v_0 & f^2 \alpha^2 + u_0^2 \alpha^2 + v_0^2 \end{bmatrix}$$

Pour le moment, l'IAC est inconnue, nous allons l'estimer

$$\tilde{\omega} = \begin{bmatrix} w_1 & 0 & w_4 \\ 0 & w_2 & w_5 \\ w_4 & w_5 & w_3 \end{bmatrix}$$

On peut ajouter des contraintes ici.  $\tilde{\omega}$  aura possiblement moins de variables. Par ex.  $\alpha = 1$ .

# Revenons à notre transformation $\mathcal{H}$

- On cherche  $\tilde{\omega}$
- À partir d'une homographie  $\mathcal{H}$  entre le plan de calibration et l'image (on ne parle pas de  $\pi_\infty$ )

On vérifie que pour  $\mathcal{H} \propto \mathcal{KR}$   $\begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t} \\ 0 & 0 & \end{bmatrix}$

$$\mathbf{h}_1^\top \omega \mathbf{h}_2 = 0, \text{ et } \mathbf{h}_1^\top \omega \mathbf{h}_1 - \mathbf{h}_2^\top \omega \mathbf{h}_2 = 0$$

où  $\mathbf{h}_i$  est la  $i^{\text{ème}}$  colonne de  $\mathcal{H}$ . Nous avons donc des contraintes linéaires :

$$\begin{bmatrix} h_1 h_2 & h_4 h_5 & h_7 h_8 & h_2 h_7 + h_1 h_8 & h_5 h_7 + h_4 h_8 \\ h_1^2 - h_2^2 & h_4^2 - h_5^2 & h_7^2 - h_8^2 & 2h_1 h_7 - 2h_2 h_8 & 2h_4 h_7 - 2h_5 h_8 \end{bmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \end{pmatrix} = \mathbf{0}$$

Nous pouvons alors retrouver l'IAC et ensuite calculer les paramètres internes.

# Décomposition de $\Omega$

On peut maintenant retrouver les paramètres.

- Calcul direct
- Décomposition de Cholesky.

## Exercice

- Combien faut-il d'homographie(s) pour calibrer complètement la caméra ?
- Si le ratio d'image est 1 et qu'on connaît le point principal, combien en faut-il ?

# Calcul de pose

- $\mathcal{K}$  connue

- Décomposer  $\mathcal{N} \propto \mathcal{K}^{-1}\mathcal{H} = \mathcal{R} \begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t} \\ 0 & 0 & \end{bmatrix}$

Deux façons :

- Retrouver la translation avec

$$\mathcal{H}^\top \omega \mathcal{H} \propto \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ t_x & t_y & t_x^2 + t_y^2 + t_z^2 \end{bmatrix}$$

- $\mathcal{K}, \mathbf{t}, \mathcal{H}$  maintenant connus.
- Retrouver  $\mathcal{R}$

- Retrouver  $\mathcal{R}$  avec

$$[\mathcal{N}^1 \quad \mathcal{N}^2 \quad \mathcal{N}^1 \times \mathcal{N}^2]$$

- forcer  $|\mathcal{R}| = 1$ ,
- retrouver la translation

Exercice : Notez l'ambiguïté sur le signe  $t_z$

- À quoi correspond chaque signe ?
- Si vous retrouvez deux positions de caméras ? Peut-on avoir deux signes différents ?
  - Si oui, pourquoi ?
  - Si non, comment corriger ça ?

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire**
  - Formulation linéaire
  - **Optimisation non-linéaire**
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Optimisation non-linéaire

- Solution linéaire non optimale
- erreur algébrique vs erreurs de reprojection
- Minimiser l'erreur de reprojection
- ATTENTION, ce n'est PAS de l'ajustement de faisceau (plus tard dans la cours)



# Optimisation de l'erreur de reprojection

Pour toutes les vues  $i$

$$\arg \min_{\mathcal{R}_i, \mathbf{t}_i, \mathcal{K}} \sum_{i,n} \text{dist}(\mathbf{p}_{in}, \mathcal{K} \mathcal{R}_i \begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t}_i \\ 0 & 0 & \end{bmatrix} \mathbf{P}_n)^2$$

- $\mathbf{P}_n$  : les points dans le plan de calibration
- Plus difficile que pour les homographies
  - Contraintes sur  $\mathcal{R}$

# Problème de la rotation

Première solution : Paramétrisation

- Euler
- Rodrigues
  
- Fonctions compliquées
- Relativement difficile à optimiser
- lent

Mais ça fonctionne

# Problème de la rotation

Deuxième solution : Optimisation itérative (recommandée)

- À chaque étape de l'algorithme d'optimisation (e.g. Levenberg-Marquardt)
- Rotation par rapport à l'étape précédente est **semblable**
- Simplification de la fonction à chaque itération.
- Nous avons :

$$\mathcal{R}^{n+1} = \mathcal{R}'\mathcal{R}^n, \quad \text{ou} \quad \mathcal{R}' = \mathcal{R}_x(\theta)\mathcal{R}_y(\phi)\mathcal{R}_z(\rho)$$

- $\theta, \phi, \rho$  sont **petits**
- Approximation de  $\mathcal{R}'$  :

$$\begin{aligned}\cos x &= 1 \\ \sin x &= x\end{aligned}$$

$$\mathcal{R}' \approx \begin{bmatrix} 1 & \rho & -\phi \\ \theta\phi - \rho & \theta\rho\phi + 1 & \theta \\ \theta\rho + \phi & \rho\phi - \theta & 1 \end{bmatrix}$$

# Problème de la rotation

Approximer encore plus :

$$\theta\phi \approx 0$$

$$\theta\rho \approx 0$$

$$\phi\rho \approx 0$$

$$\mathcal{R}' \approx \begin{bmatrix} 1 & \rho & -\phi \\ -\rho & 1 & \theta \\ \phi & -\theta & 1 \end{bmatrix}$$

- matrice  $\mathcal{R}^n$  est fixe,
- à la fin de l'itération on met à jour :

$\mathcal{R}^{n+1} = \mathcal{R}'\mathcal{R}^n$  avec la formule de rotation originale (avec les cos et sin).

# Problème de la rotation

## Exercice (plutôt technique)

- Chercher un façon d'implémenter cela en **Mathematica**

Faire une version itérative de **FindMinimum**

En faisant des appels a FindMinimum

Comprendre pourquoi, dans Mathematica, cela n'est pas très efficace

Ceci est assez difficile si vous debutez avec Mathematica

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure**
- 5 Géométrie épipolaire
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

# Méthode par rotation pure

## Définition

Caméra qui tourne parfaitement autour de son centre optique  
( $\mathbf{t} = \mathbf{0}$ )

- Aucun effet de profondeur
- Images de caméra reliée par des homographies

Deux caméras avec

- $\mathcal{H}_1 = \mathcal{K}$  (sans rotation, ni translation)
- $\mathcal{H}_2 = \mathcal{K}\mathcal{R}$

Relier ensemble en faisant avec  $\text{Image}_1 \rightarrow \text{Monde} \rightarrow \text{Image}_2$  :

$$\mathcal{H}_{12} = \mathcal{K}\mathcal{R}\mathcal{K}^{-1}$$

# Retrouver la rotation

## Rotation conjuguée

Une matrice  $\mathcal{H} = \mathcal{U}\mathcal{R}\mathcal{U}^{-1}$ , où  $\mathcal{R}$  est une matrice orthogonale et  $\mathcal{U}$  est une transformation projective.

Pour une matrice de rotation  $\mathcal{R}$ , nous avons

- valeurs propres de la forme  $\lambda(1, e^{i\theta}, e^{-i\theta})$ ,  
(une réelle et deux complexes conjuguées)
- vecteurs propre correspondants ( $\mathbf{a}, \mathbf{I}, \mathbf{J}$ )

$\mathbf{a}$  axe de rotation de la matrice

$\theta$  l'angle

Pour une rotation conjuguée, ces propriétés sont préservées.



# Retrouver la calibration

Un cas simple :

ratio d'image à 1, point principal connu et mis à l'origine

Supposons  $\mathcal{K} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$ . Dans ce cas  $\mathcal{H} = \begin{bmatrix} r_1 & r_2 & fr_3 \\ r_4 & r_5 & fr_6 \\ \frac{r_7}{f} & \frac{r_8}{f} & r_9 \end{bmatrix}$

- $\mathcal{H}$  : presque une matrice de rotation
- solutions analytiques pour  $f$

$$h_1^2 + h_2^2 + h_3^2/f^2 = h_4^2 + h_5^2 + h_6^2/f^2$$

ou bien

$$h_1h_4 + h_2h_5 + \frac{h_3h_6}{f^2} = 0$$

Vous avez tout ce qu'il faut pour faire le TP2!

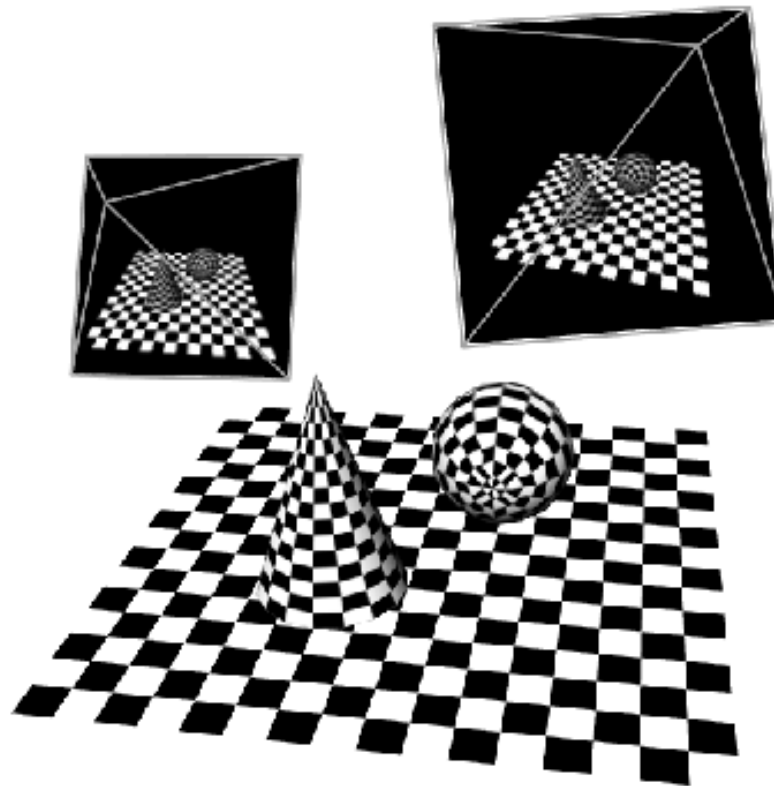
# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire**
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

## Géométrie épipolaire

Relation des correspondances images de points 3D entre deux vues.

- points 3D inconnus



# Calibration

## Problème

Calculer les paramètres d'une caméra en utilisant uniquement des points mis en correspondance entre deux images d'une même scène, prises sous des points de vue différents.

→ Exprime la relation entre les paramètres externes des deux caméras, pas entre les caméras et la scène.

## 2 cas : Paramètres internes

- inconnus (→ matrice **Fondamentale**  $\mathcal{F}$ )
- connus (→ matrice **Essentielle**  $\mathcal{E}$ )

# Première étape

Entre deux vues :

- Établir des correspondances entre les images
- Deux solutions :
  - “Tracking” de points saillants au long d’une séquence
  - Mise en correspondances automatique des points descripteurs SIFT ou autre  
voir aussi  
<http://www.robots.ox.ac.uk/~vgg/research/affine/index.ht>

# Aperçu

## Matrice fondamentale

$$\mathbf{p}'^T \mathcal{F}_{3 \times 3} \mathbf{p} = 0$$

$\mathbf{p}'$  et  $\mathbf{p}$  sont des points images

## Matrice essentielle

$$\hat{\mathbf{p}}'^T \mathcal{E}_{3 \times 3} \hat{\mathbf{p}} = 0$$

$\hat{\mathbf{p}}'$  et  $\hat{\mathbf{p}}$  sont des points caméras :

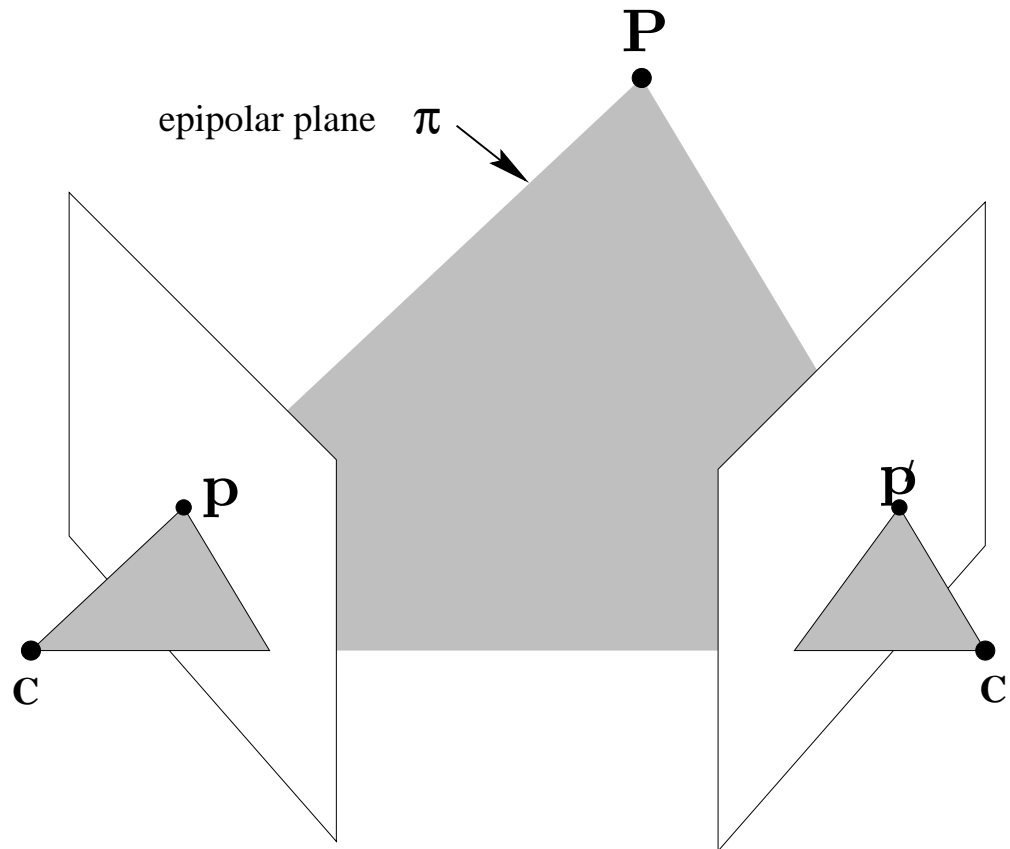
$$\mathbf{p} = \mathcal{K} \hat{\mathbf{p}}$$

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire**
  - **Matrice fondamentale**
  - Matrice Essentielle
  - Retrouver les caméras



# Dérivation



- Deux caméras :  $\mathcal{M}$  (centre en  $C$ ) et  $\mathcal{M}'$ , (centre en  $C'$ )
- Un point 3D :  **$P$  inconnu**
- points image :
  - $p \propto \mathcal{M}P$
  - $p' \propto \mathcal{M}'P$

# Dérivation

Pour la première caméra, déprojection de  $\mathbf{p}$

$$\mathbf{P}(\lambda) = \mathcal{M}^\dagger \mathbf{p} + \lambda \mathbf{C}$$

Deux points sur cette ligne

- $\mathcal{M}^\dagger \mathbf{p}$
- $\mathbf{C} (\lambda = \infty)$

On projette dans l'image 2 et on joint les points

$$\mathbf{l}' = (\mathcal{M}' \mathbf{C}) \times (\mathcal{M}' \mathcal{M}^\dagger \mathbf{p})$$

# Dérivation

## Épipole

Projection du centre de la caméra dans la deuxième

$$\mathbf{e}' \propto \mathcal{M}'\mathbf{C}$$

$$\mathbf{l}' = \mathbf{e}' \times (\mathcal{M}'\mathcal{M}^\dagger \mathbf{p}) = [\mathbf{e}']_{\times} \mathcal{M}'\mathcal{M}^\dagger \mathbf{p}$$

Comme  $\mathbf{p}'$  est quelque part sur  $\mathbf{l}'$

$$\mathbf{p}'^T \mathbf{l}' = \mathbf{p}'^T \underbrace{[\mathbf{e}']_{\times} \mathcal{M}'\mathcal{M}^\dagger}_{\mathcal{F}} \mathbf{p} = 0$$

## Matrice Fondamentale

$$\mathcal{F} = [\mathbf{e}']_{\times} \mathcal{M}'\mathcal{M}^\dagger$$

# Propriétés

- $3 \times 3$
- Rang 2, car  $[\mathbf{e}']_{\times}$  est rang 2
- 7 degrés de libertés (à un facteur d'échelle + rang 2)
- donc  $\det(\mathcal{F}) = 0$

Points :

- vues 1  $\leftrightarrow$  2       $\mathbf{p}'^{\top} \mathcal{F} \mathbf{p} = 0$
- vues 2  $\leftrightarrow$  1       $\mathbf{p}^{\top} \mathcal{F}^{\top} \mathbf{p}' = 0$

Lignes épipolaires

- vues 1  $\rightarrow$  2       $\mathbf{l}' = \mathcal{F} \mathbf{p}$
- vues 2  $\rightarrow$  1       $\mathbf{l} = \mathcal{F}^{\top} \mathbf{p}'$

# Exemple

Deux caméras

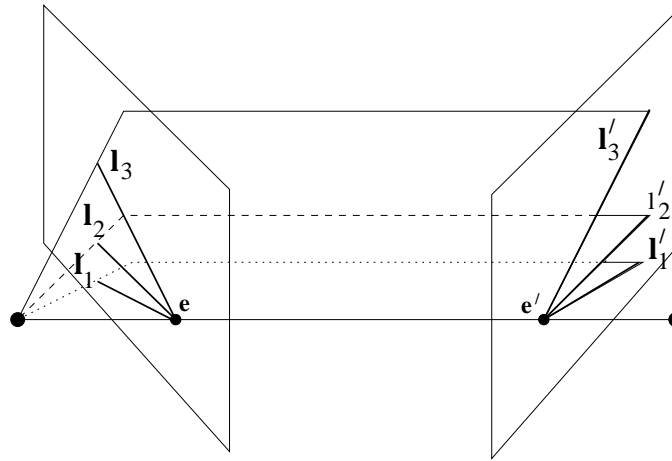
- $\mathcal{M} = \mathcal{K} [\mathbf{I} \quad \mathbf{0}]$
- $\mathcal{M} = \mathcal{K}' [\mathcal{R} \quad \mathbf{t}]$

d'où

- $\mathcal{M}^\dagger = \begin{bmatrix} \mathcal{K}^{-1} \\ \mathbf{0}^\top \end{bmatrix}$
- $\mathbf{C} = (\mathbf{0}^\top \quad 1)^\top$

$$\begin{aligned} \mathcal{F} &= [\mathcal{M}'\mathbf{C}]_\times \mathcal{M}'\mathcal{M}^\dagger \\ &= [\mathbf{t}]_\times \mathcal{K}'\mathcal{R}\mathcal{K}^{-1} \\ &= \mathcal{K}'^{-\top} [\mathbf{t}]_\times \mathcal{R}\mathcal{K}^{-1} \end{aligned}$$

# Épipoles



## Épipole

Intersection de toutes les lignes épipolaires

$$l'_1 = \mathcal{F}p_1 \quad l'_2 = \mathcal{F}p_2$$

et

$$e'^T l'_1 = e'^T l'_2 = 0 \Rightarrow e'^T \mathcal{F} = 0$$

- $e'$  est le noyau **gauche** de  $\mathcal{F}$
- $e$  est le noyau **droit** de  $\mathcal{F}$

## Algorithme des 8 points

- Algorithme simple et populaire.
- Formulation linéaire
- Contrainte de rang forcée indirectement

Soit  $N$  points (avec  $N \geq 8$ ) mis en correspondance

On a l'équation

$$\mathbf{p}^T \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \mathbf{q} = 0$$

- Une correspondance  $\rightarrow$  une contrainte

# Estimation

$$\mathbf{A} \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{pmatrix} = 0$$

avec

$$\mathbf{A} = \begin{bmatrix} p_1q_1 & p_1q_2 & p_1q_3 & p_2q_1 & p_2q_2 & p_2q_3 & p_3q_1 & p_3q_2 & p_3q_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Systeme linéaire homogène... → SVD.

Il faut absolument **NORMALISER** les points



# Estimation

Forcer la contrainte de rang 2 :

Nous avons  $\mathcal{F}$  :

- trouvée avec l'algorithme des 8 points
- Rang 3

On cherche  $\mathcal{F}'$  :

- Rang 2
- minimise  $\|\mathcal{F} - \mathcal{F}'\|$

$$\mathcal{F} = \mathcal{U} \text{diag}(\sigma_1, \sigma_2, \sigma_3) \mathcal{V}^T$$

$$\mathcal{F}' = \mathcal{U} \text{diag}(\sigma_1, \sigma_2, 0) \mathcal{V}^T$$

Effectuer cela avant la dénormalisation de  $\mathcal{F}'$

# Estimation

Formulation linéaire algébrique :

$$\sum_i (\mathbf{p}_i^\top \mathcal{F} \mathbf{p}'_i)^2$$

Error géométrique optimale :

$$\sum_i \text{dist}^2(\mathbf{p}_i, \mathcal{F} \mathbf{p}'_i) + \text{dist}^2(\mathbf{p}'_i, \mathcal{F}^\top \mathbf{p}_i)$$

Minimiser la distance des points à la ligne épipolaire

# Estimation

$$\sum_i \text{dist}^2(\mathbf{p}_i, \mathcal{F}\mathbf{p}'_i) + \text{dist}^2(\mathbf{p}'_i, \mathcal{F}^\top \mathbf{p}_i)$$

Minimiser la distance des points à la ligne épipolaire

- paramétrisation pour  $\mathcal{F}$ ,
- pour forcer les **contraintes**

ex :

- sur-paramétrisation :

$$\mathcal{F} = [\mathbf{t}]_{\times} \mathcal{M}$$

- dernière colone comme combinaison linéaire des deux premières :

$$\mathcal{F} = \begin{bmatrix} a & b & \alpha a + \beta b \\ c & d & \alpha c + \beta d \\ e & f & \alpha e + \beta f \end{bmatrix}$$

- Singularité si  $F_1$  et  $F_2$  sont linéairement dépendante

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire**
  - Matrice fondamentale
  - Matrice Essentielle**
  - Retrouver les caméras

# Dérivation

$$\hat{\mathbf{p}}' \mathcal{E}_{3 \times 3} \hat{\mathbf{p}} = 0$$

$\hat{\mathbf{p}}'$  et  $\hat{\mathbf{p}}$  sont des points caméras :

$$\mathbf{p} = \mathcal{K} \hat{\mathbf{p}}$$

$$\mathcal{F} = \mathcal{K}'^{-\top} \underbrace{[\mathbf{t}]_{\times} \mathcal{R}}_{\mathcal{E}} \mathcal{K}^{-1}$$

On enlève les matrices de paramètres internes

$$\mathcal{E} = \mathcal{K}'^{\top} \mathcal{F} \mathcal{K} = [\mathbf{t}]_{\times} \mathcal{R} = \mathcal{R} [\mathcal{R}^{\top} \mathbf{t}]_{\times}$$

# Propriétés

- 5 degrés de libertés
  - + 3 pour  $\mathcal{R}$
  - + 3 pour  $\mathbf{t}$
  - 1 pour facteur d'échelle
- 2 valeur singulières égales  
Famille de décomposition

Paramétrisation SVD de  $\mathcal{E}$  :

$$\mathcal{E} = \mathcal{U} \text{diag}(1, 1, 0) \mathcal{V}^T$$

n'est pas unique

$$\mathcal{E} = (\mathcal{U} \text{diag}(\mathcal{R}_{2 \times 2}, 1)) \text{diag}(1, 1, 0) \text{diag}(\mathcal{R}_{2 \times 2}^T, 1) \mathcal{V}^T \quad \forall \mathcal{R}$$

donc

- +3 pour  $\mathcal{U}$
- +3 pour  $\mathcal{V}$
- -1 pour  $\mathcal{R}$

# Sommaire

- 1 Introduction
- 2 Calibration avec objet 3D
- 3 Calibration planaire
  - Formulation linéaire
  - Optimisation non-linéaire
- 4 Calibration par Rotation pure
- 5 Géométrie épipolaire**
  - Matrice fondamentale
  - Matrice Essentielle
  - Retrouver les caméras

## À partir $\mathcal{F}$

On ne peut retrouver que des caméras perspective  
(non-euclidienne)

On peut choisir

$$\begin{aligned}\mathcal{M} &= [\mathbf{I} \quad \mathbf{0}] \\ \mathcal{M}' &= [[\mathbf{e}']_{\times} \mathcal{F} \quad \mathbf{e}']\end{aligned}$$

( $\mathcal{M}'$  est rang 2, qu'est-ce que ça implique?)

Mais de façon générale :

$$\begin{aligned}\mathcal{M} &= [\mathbf{I} \quad \mathbf{0}] \\ \mathcal{M}' &= [[\mathbf{e}']_{\times} \mathcal{F} + \mathbf{e}' \mathbf{v}^T \quad \lambda \mathbf{e}']\end{aligned}$$

pour tout  $\mathbf{v}$  et pour  $\lambda$  non-nul



# Ambiguïté de $\mathcal{F}$

Plus tard dans le cours

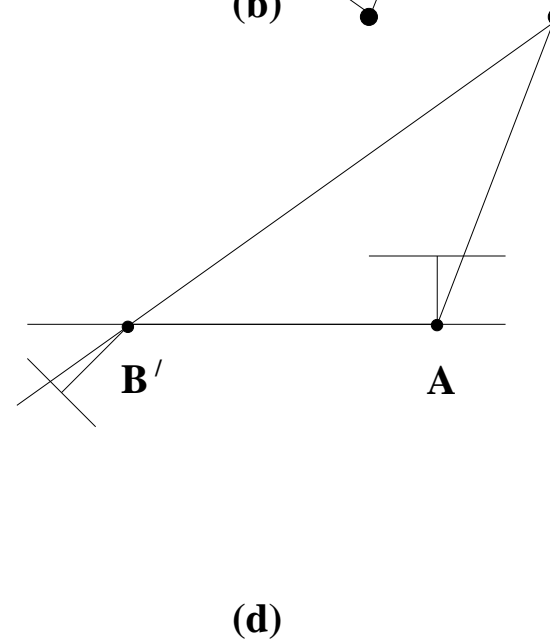
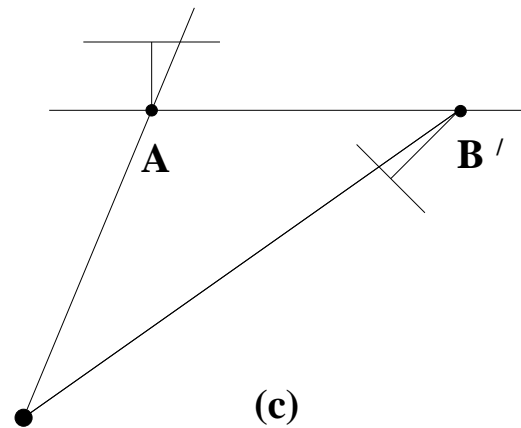
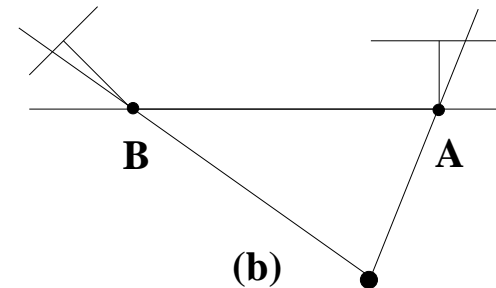
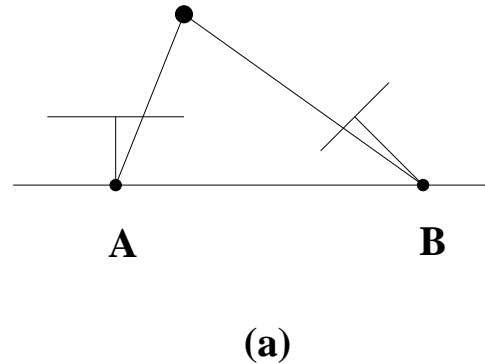
$\mathcal{M}$  et  $\mathcal{M}'$  sont connus à une transformation projective  $\mathcal{H}_{4 \times 4}$

- Passage de projectif à euclidien
- Auto-calibrage

# Pose à partir de $\mathcal{E}$

Rappel :  $\mathcal{K}$  et  $\mathcal{K}'$  sont connus

- 4 solutions possibles :



- 1 seule valide

# Pose à partir de $\mathcal{E}$

$$\mathcal{E} = \mathcal{U} \text{diag}(1, 1, 0) \mathcal{V}^T$$

$$\text{Première caméra : } \mathcal{M} = [\mathbf{I} \quad \mathbf{0}]$$

Deuxième caméra :

$$\mathcal{M}' = [\mathcal{U}\mathcal{W}\mathcal{V}^T \mid + \mathbf{u}_3]$$

$$\mathcal{M}' = [\mathcal{U}\mathcal{W}\mathcal{V}^T \mid - \mathbf{u}_3]$$

$$\mathcal{M}' = [\mathcal{U}\mathcal{W}^T \mathcal{V}^T \mid + \mathbf{u}_3]$$

$$\mathcal{M}' = [\mathcal{U}\mathcal{W}^T \mathcal{V}^T \mid - \mathbf{u}_3]$$

$$\text{avec } \mathcal{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{u}_3 \text{ la dernière colonne de } \mathcal{U}$$

# Pose à partir de $\mathcal{E}$

Il faut choisir  $\mathcal{M}'$  :

- points devant les deux caméras
- formule de la profondeur des points doit être positive

FIN

Certaines figures sont la courtoisie de  
R. Hartley et A. Zisserman  
Multiple View Geometry in Computer Vision, 1ère Édition, 2002