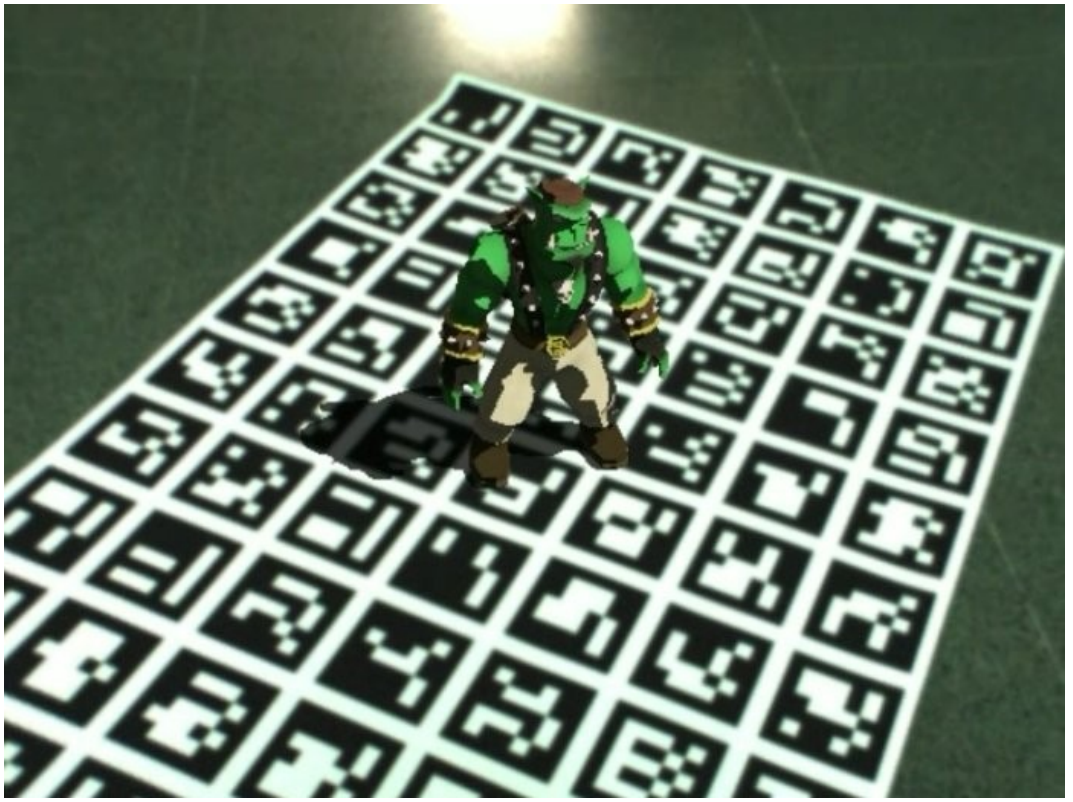
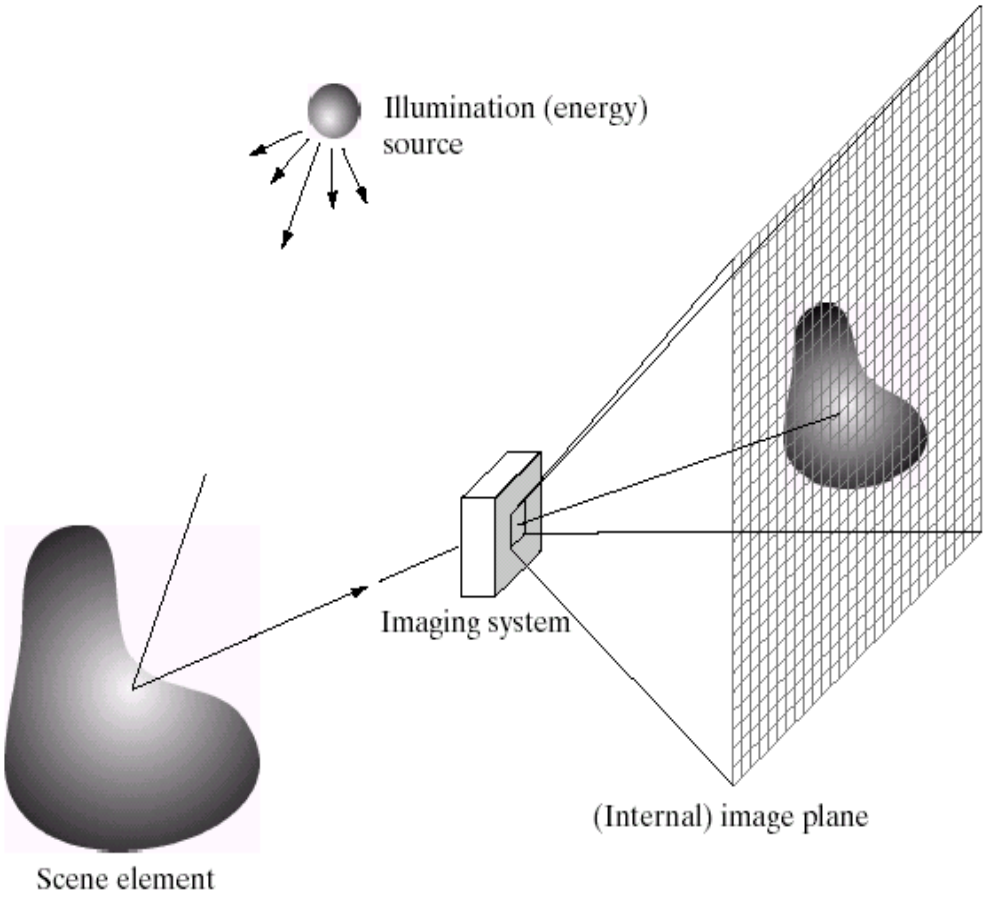
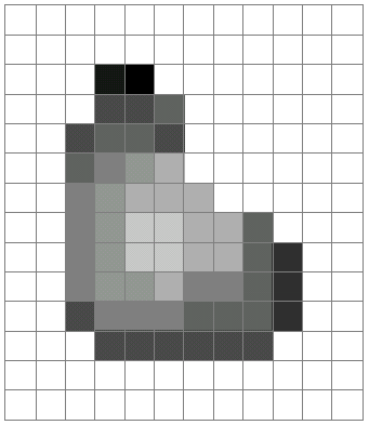
Réalité augmentée

- Inclusion d’objets 3D dans des scènes temps réel



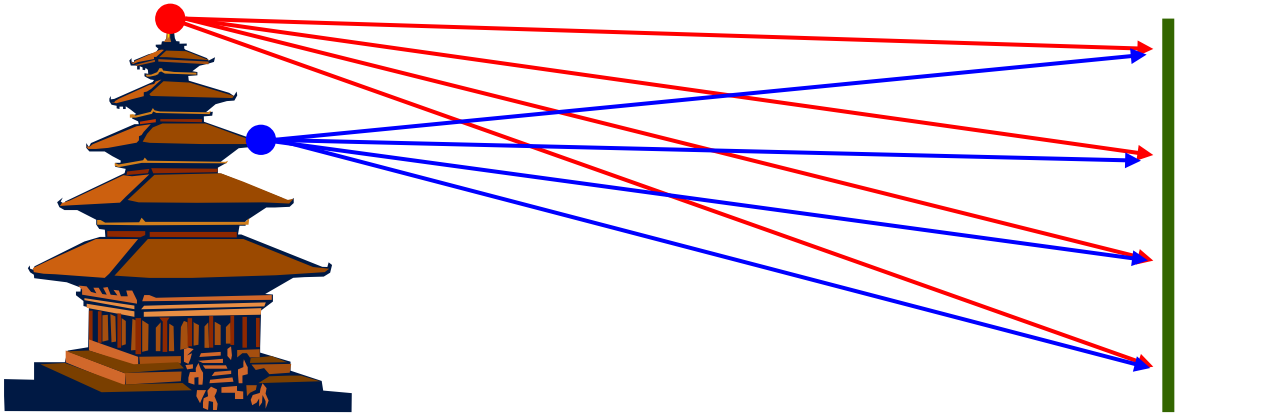
Calibration de caméra





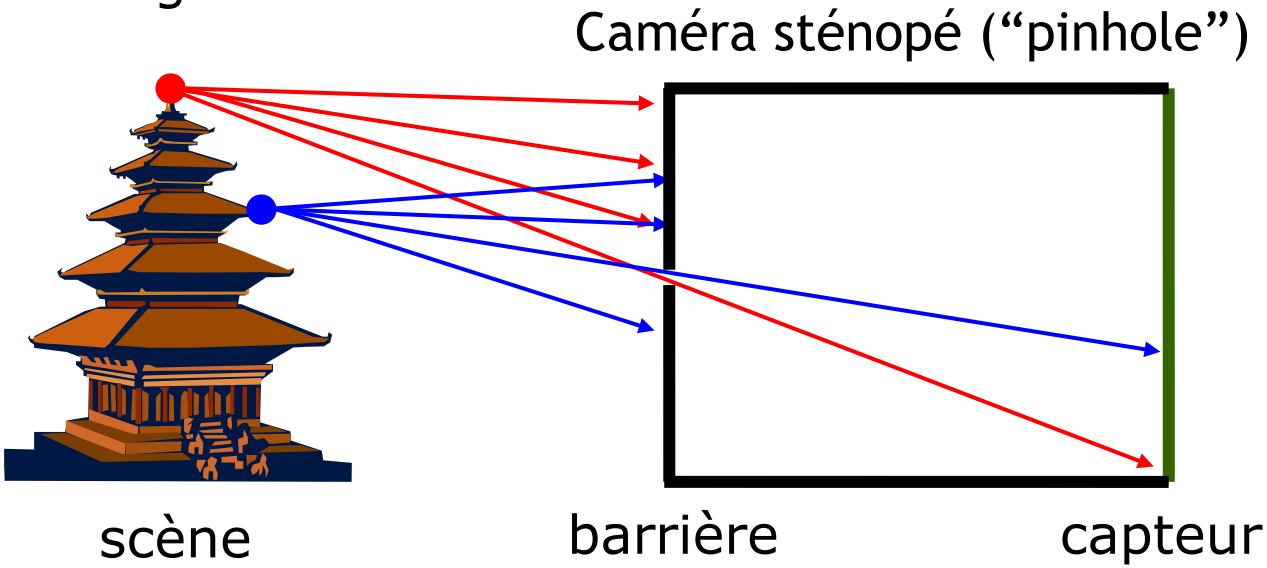
Caméra à sténopée

- Modèle « trou d'épingle » ou « pinhole »

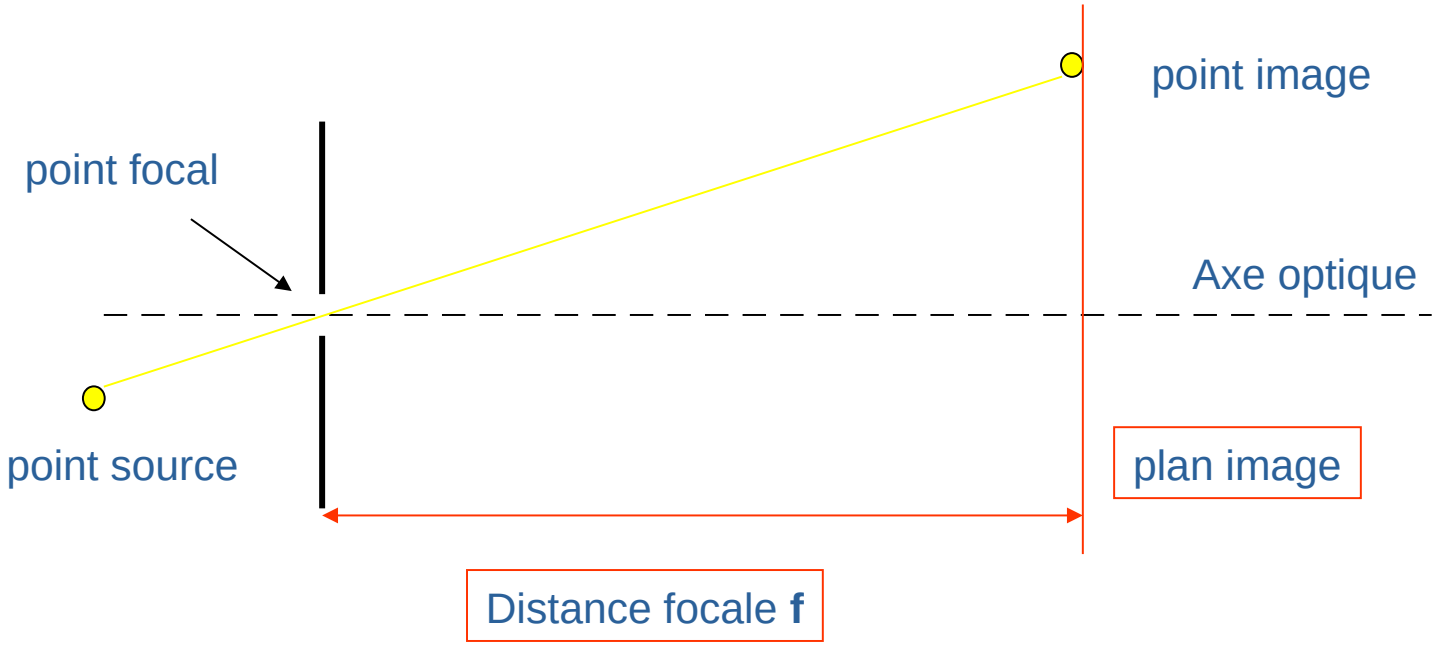


D'après Patrick Hébert

- Modèle « trou d'épingle » ou « pinhole » :
  - Ajouter une barrière pour bloquer la plupart des rayons.
    - Réduit le flou
    - image inversée

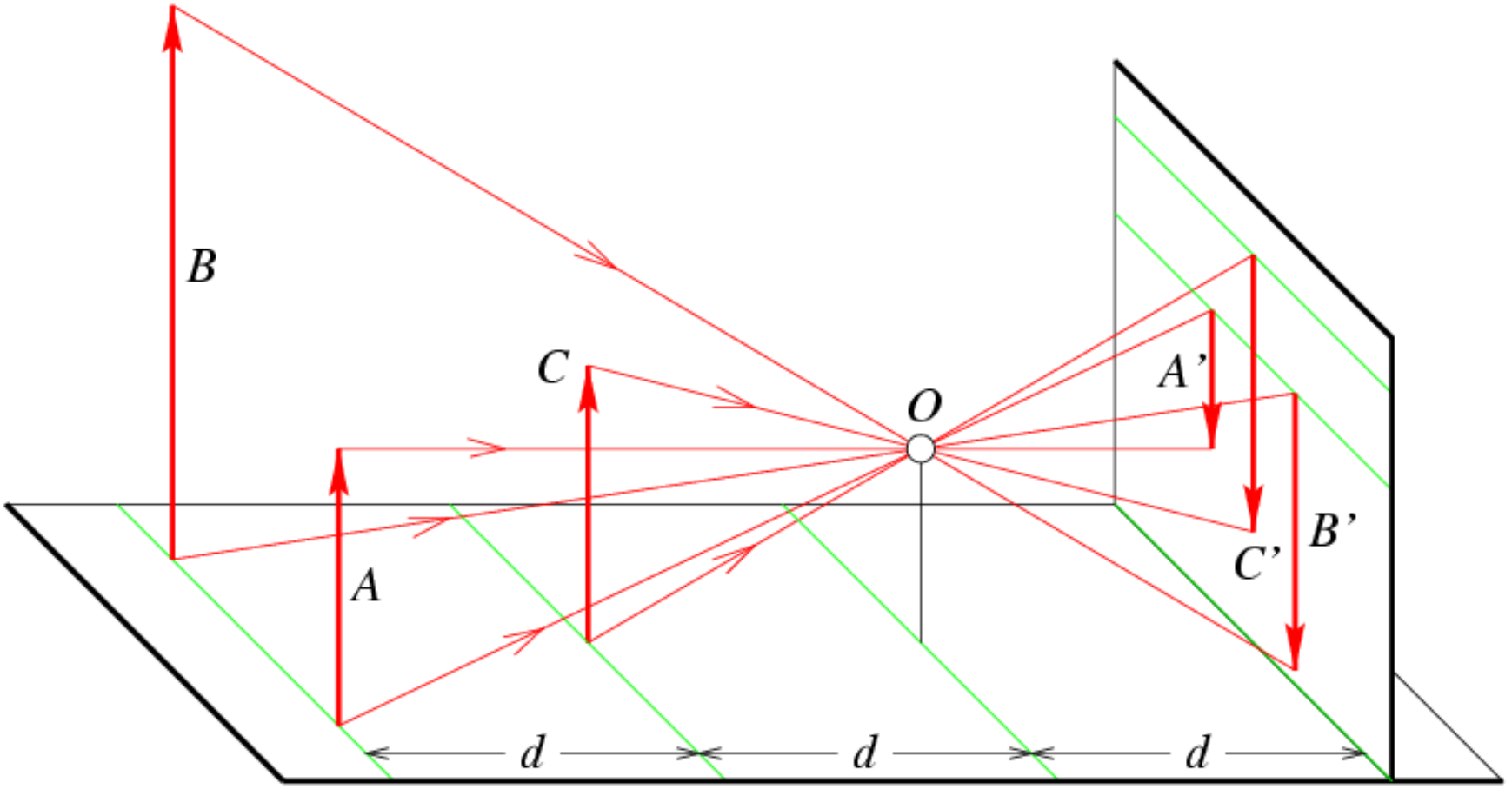


D'après Patrick Hébert

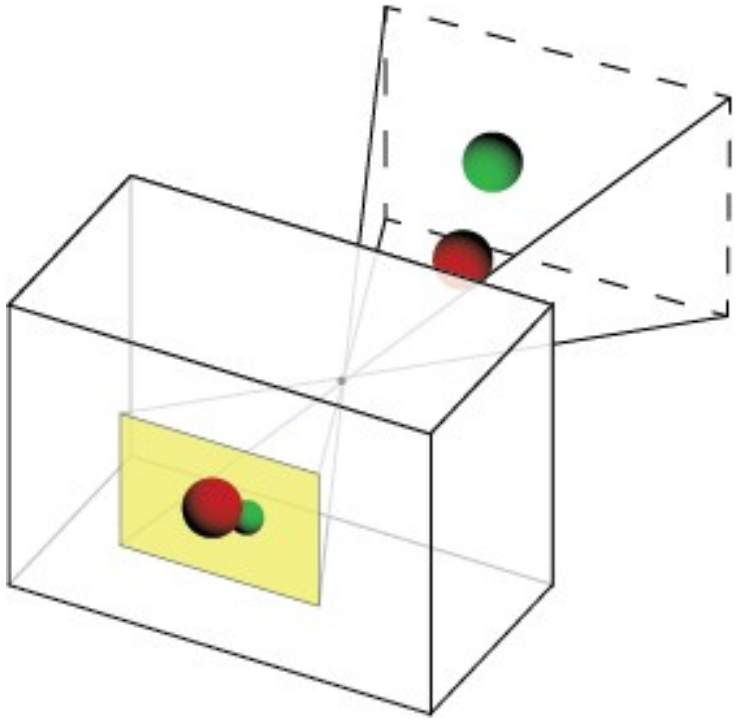


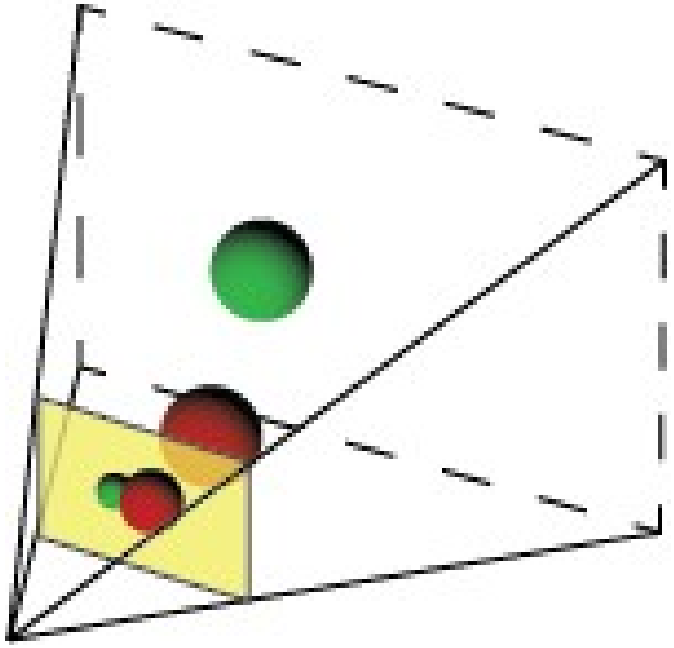
D’après William Puech

- Les objets distants sont plus petits

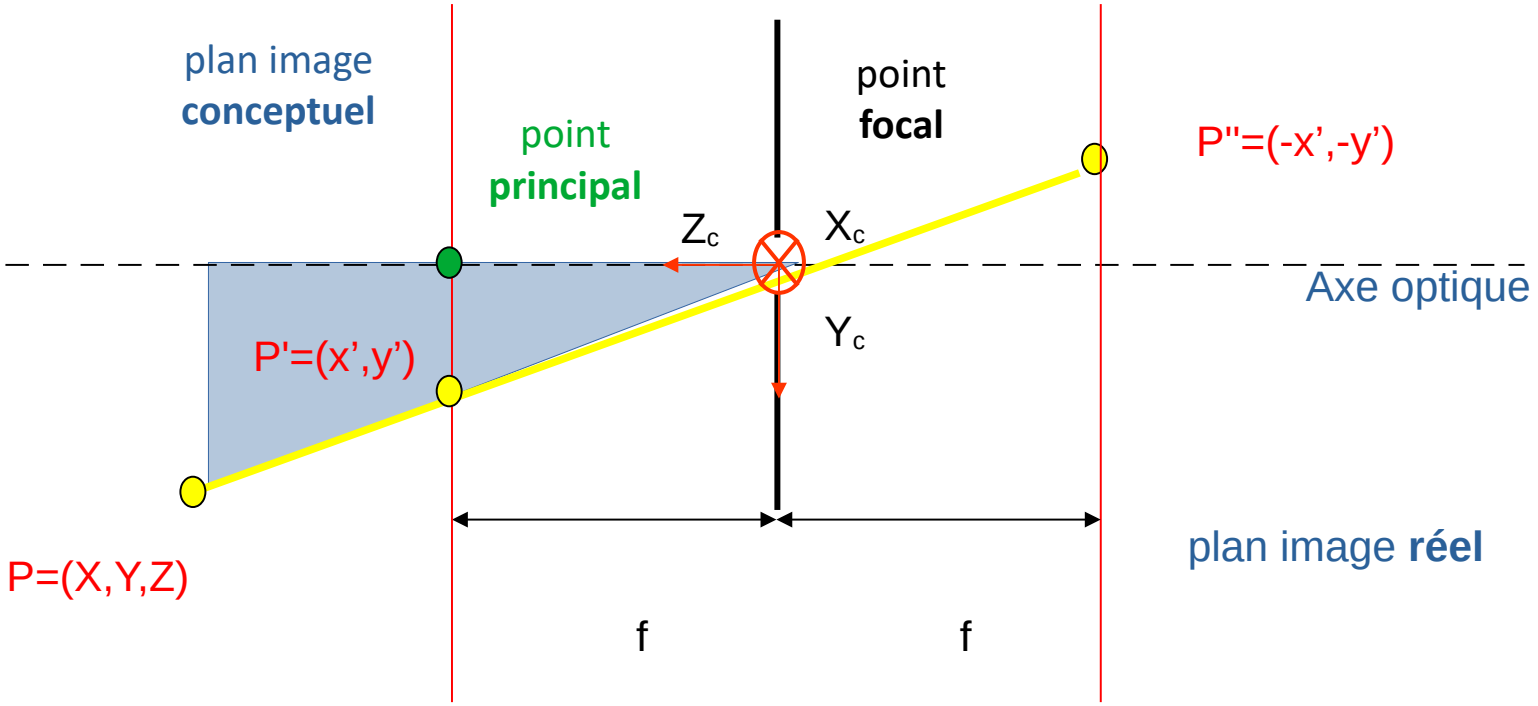


Souvent représenté avec un plan image « conceptuel » devant la caméra





Souvent représenté avec un plan image « conceptuel » devant la caméra

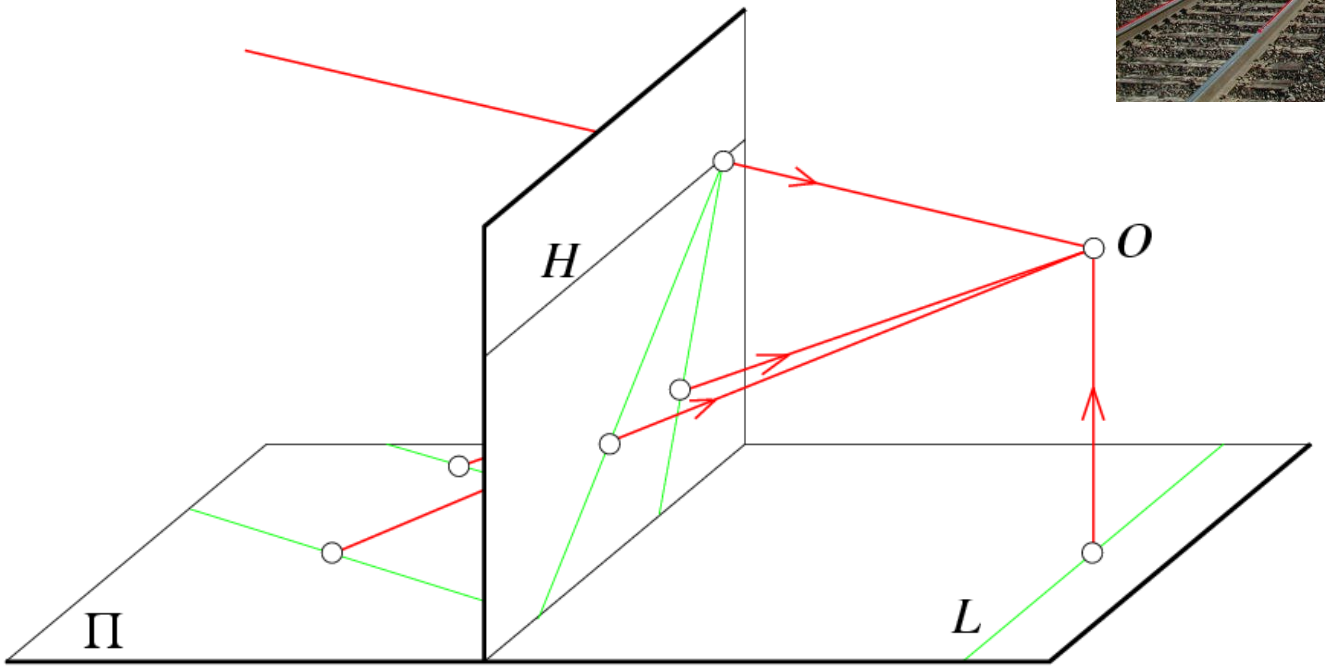


$$y' = f\frac{Y}{Z}$$

Thalès (idem pour *x'=f X/Z)*

Changer la focale = zoomer

- Les // se coupent : point de fuite
- Transformation projective
- Homographie



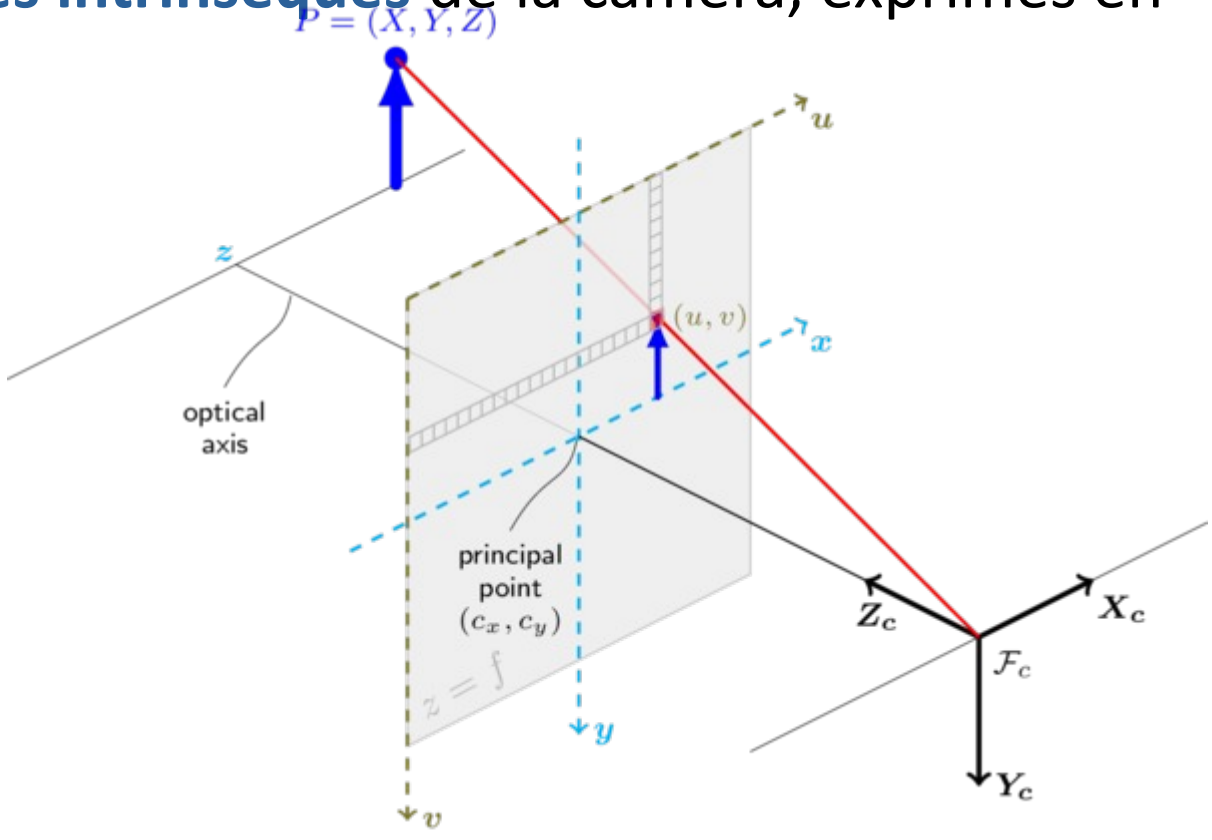
Paramètres intrinsèques

- (u,v) dans l'image dépend de :
  - (c_x ,c_y) : centre d'image→ translation
  - f : zoom = échelle

    f_x et f_y si pixels non carrés (rectangulaires...)
  - Ce sont les **paramètres intrinsèques** de la caméra, exprimés en pixels

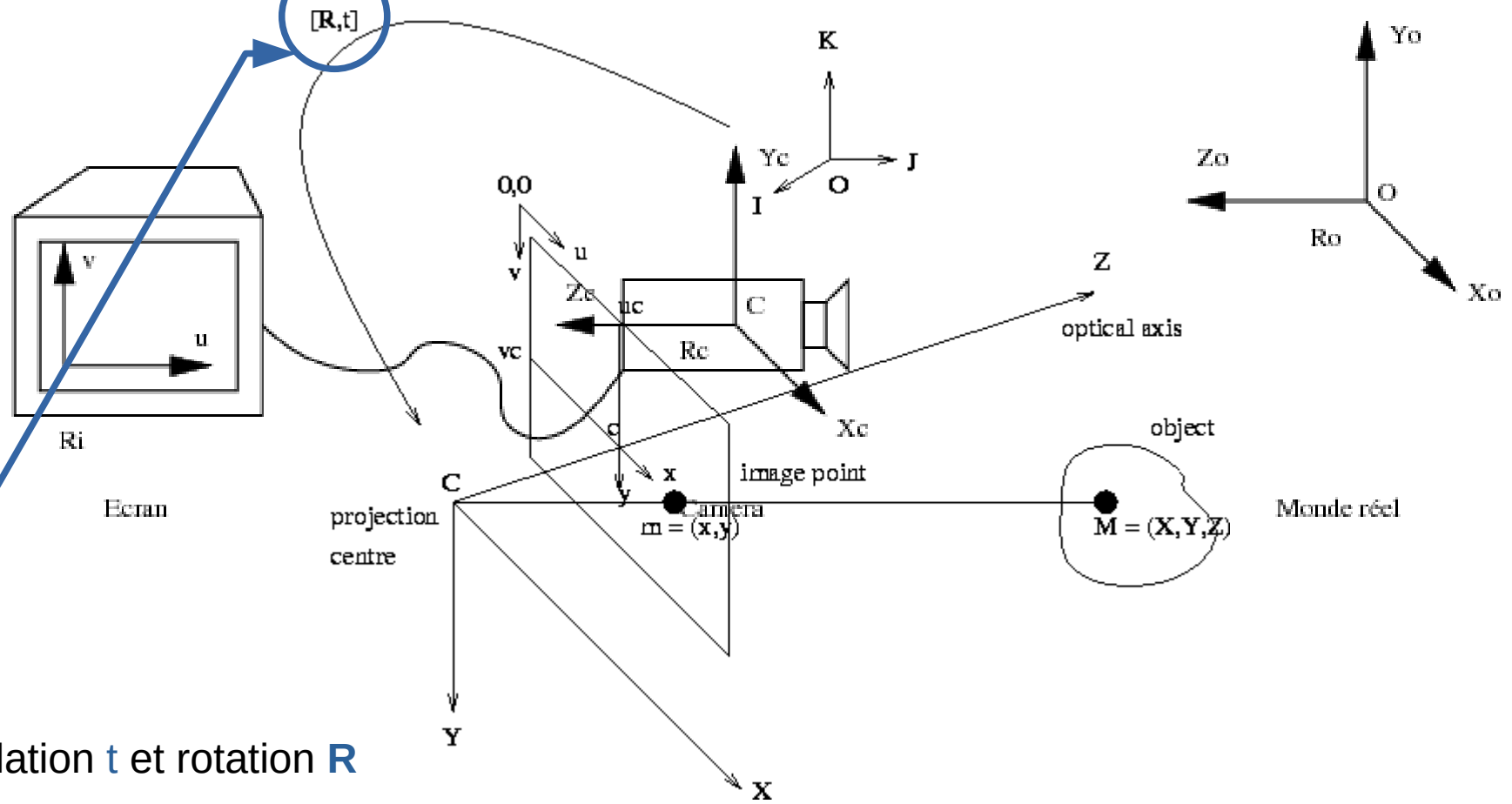
$u=f_xX/Z+c_x$

$v=f_yY/Z+c_y$



Paramètres extrinsèques

- Les coordonnées du monde réel doivent être exprimées dans le repère de la caméra



Translation t et rotation **R**

- Ce qui s'écrit :

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t$$

$$x' = x/z$$
$$y' = y/z$$
$$u = f_x . x' + c_x$$
$$v = f_y . y' + c_y$$

En coordonnées homogènes :

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

- Passage 3D → 2D : l'échelle de profondeur (w) est perdue. Les coordonnées homogènes sont équivalentes par changement d'échelle

$$\begin{pmatrix} w * x \\ w * y \\ w \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- En pratique il suffit de diviser toutes les coordonnées par w pour obtenir les coordonnées cartésiennes

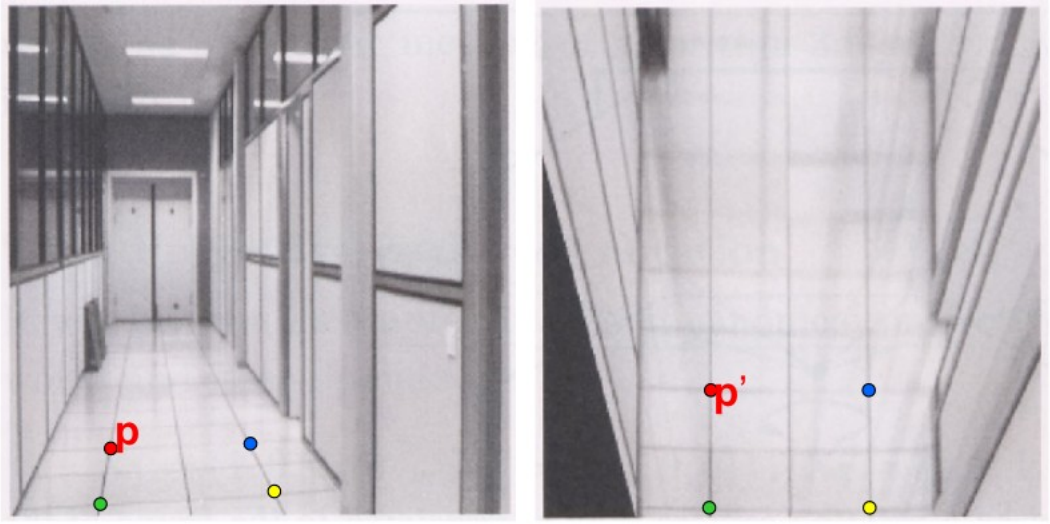
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Coordonnées homogènes

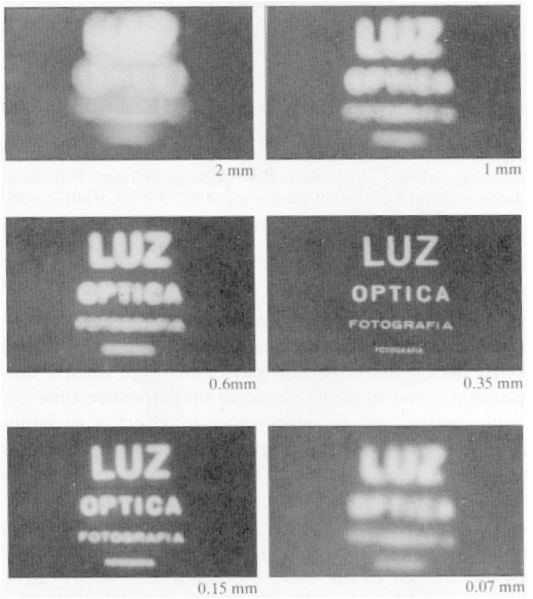
- Les r_{ij} et t_j doivent être estimés

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

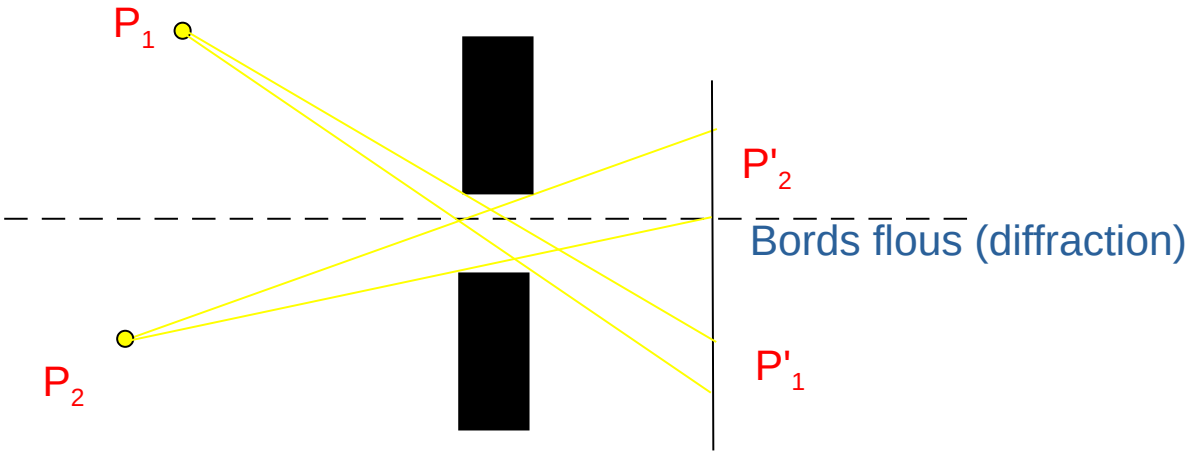
- Dans le cas où les points utilisés pour l'estimation sont coplanaires, 4 points suffisent

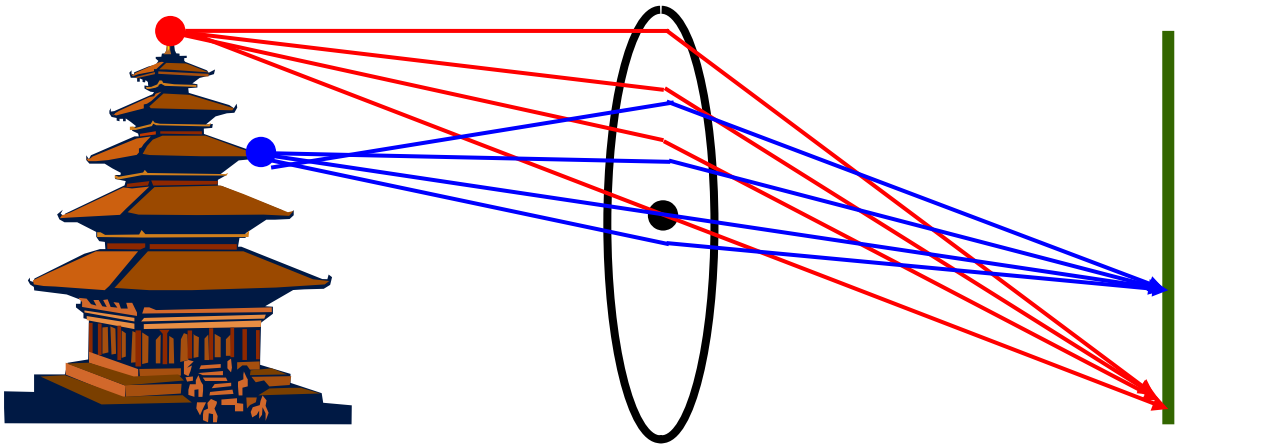


Sténopée



- Inconvenients :
  - Flou :
    - Trou trop large : on moyenne trop de rayons lumineux
    - Trou trop petit : c'est la diffraction qui engendre le flou
  - Sombre : peu de rayons atteignent l'écran
  - sensible à l'épaisseur de la plaque et aux positions des sources lumineuses.





Une lentille focalise la lumière sur le capteur.

- Mais les lentilles peuvent entrainer des distorsions qu'il faut corriger…

  https://fr.wikipedia.org/wiki/Distorsion_(optique)

- Mais il faut faire de l'optique (théorique...)

- Au final, il faut appliquer les transformations dans l'ordre :
  - Passage coordonnées globales à caméra (extrinsèque)
  - Projection dans l'image (intrinsèque)
  - Correction des déformations (distorsions)

Au final...

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t$$

$$x' = x/z$$

$$y' = y/z$$

$$x'' = x' \frac{1+k_1 r^2+k_2 r^4+k_3 r^6}{1+k_4 r^2+k_5 r^4+k_6 r^6} + 2p_1 x' y' + p_2(r^2+2x'^2) + s_1 r^2 + s_2 r^4$$

$$y'' = y' \frac{1+k_1 r^2+k_2 r^4+k_3 r^6}{1+k_4 r^2+k_5 r^4+k_6 r^6} + p_1(r^2+2y'^2) + 2p_2 x' y' + s_3 r^2 + s_4 r^4$$

$$\text{où} \quad r^2 = x'^2 + y'^2$$

$$u = f_x . x'' + c_x$$

$$v = f_y . y'' + c_y$$

Coefficients de distorsion :
- k1, k2, k3, k4, k5 et k6
- p1 et p2
- s1, s2, s3 et s4

Voir https://docs.opencv.org/3.4.4/d9/d0c/group__calib3d.html

Calibration JeVois

- La JeVois propose une calibration par défaut ainsi qu'un programme de calibration :

  http://jevois.org/tutorials/UserArUcoCalib.html

- En général, les paramètres intrinsèques par défaut sont suffisants.

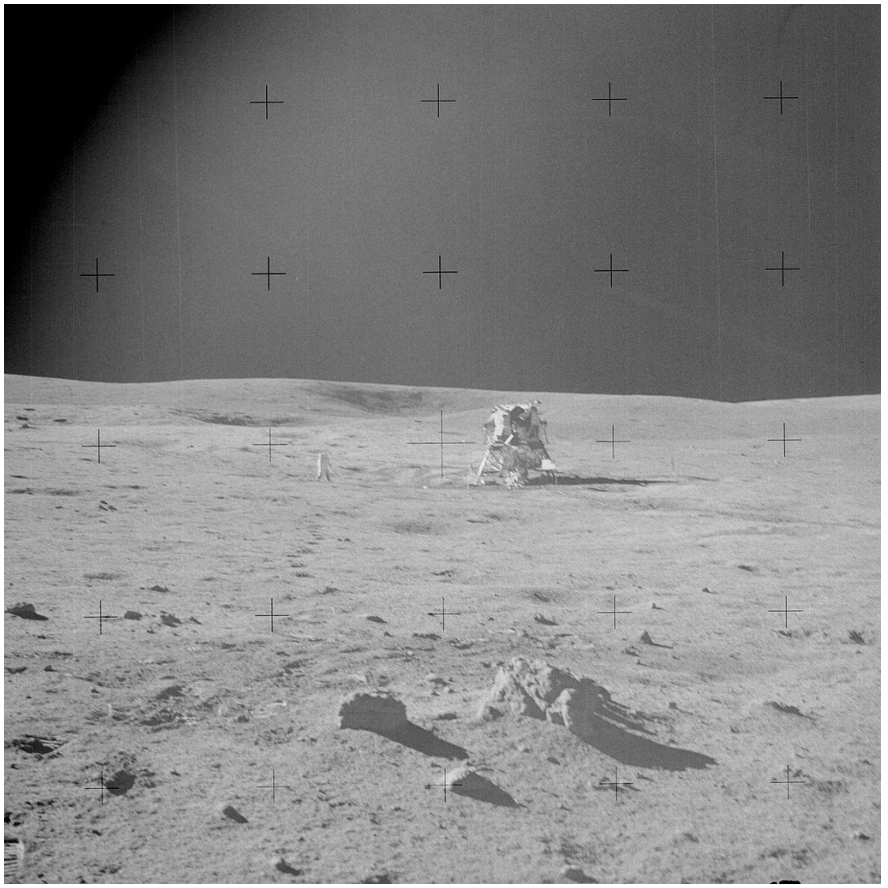
Réalité augmentée

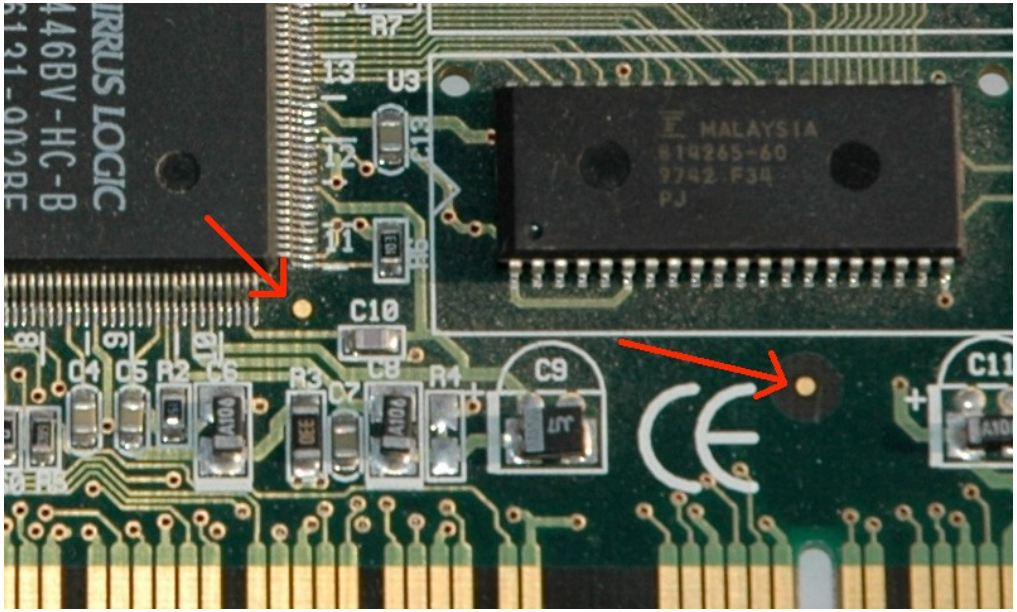
- Si on connaît les matrices de transformation on peut inclure facilement des objets dans une scène
- Par exemple à partir des points d'intérêt (Sift,ORB)

Création/détection de repères

- « Fiducial markers » : marque repère
- objet placé dans une scène pour servir de point de référence, de mesure, d'identifiant.

Repères





Identification

- Codes barres :
  - 1D
  - 2D

	code-barre	code carré
		
Date	1952	1999
Protocoles	EAN	QR code (ISO 18004); flashcode
Dimensions	1	2
Données encodées	8, 10, 13 chiffres (EAN), possibilité de 15 et 18.	Jusqu'à 7089 caractères numériques; ou 4296 caractères alphanumériques.
Utilisations	Gestion et vente de produits (y compris publications), étiquetage de biens, distribution de courrier ou de cargaison, indexation de documents.	Lien rapide vers contenu en ligne, échange de coordonnées, paiement, suivi de lots ou pièces.
Appareils de saisie	Lecteur de code-barres	Smartphones, capteur spécial

- Essentiellement utile

  pour l'identification,

  très nombreuses variantes

  https://en.wikipedia.org/wiki/Barcode

- De plus en plus utilisés pour d'autres applications dont la réalité augmentée (estimation de pose).

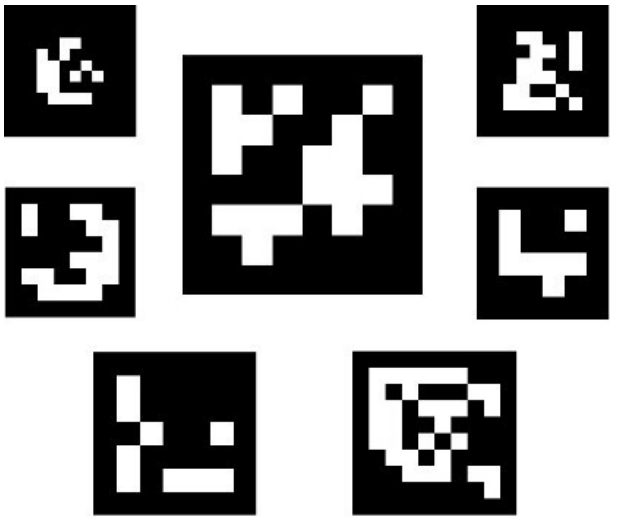
Aruco

S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. 2014.

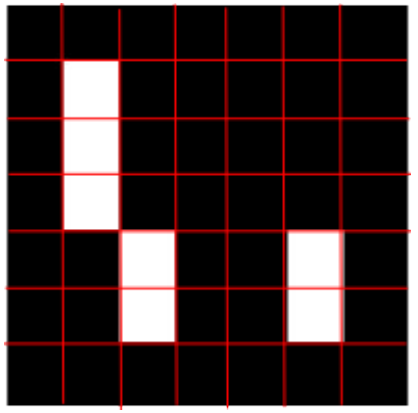
"Automatic generation and detection of highly reliable fiducial markers under occlusion". Pattern Recogn. 47, 6 (June 2014), 2280-2292. DOI=10.1016/j.patcog.2014.01.005

- Estimation de la pose : déplacement de robots, réalité augmenté
- Principe : mise en correspondance entre l'environnement réel (3D) et l'image ou la vidéo (2D)
  - Difficile en général mais facilité par la présence de marqueurs.

- Exemples :



- Composition :
  - une bordure extérieure noire
  - un motif binaire blanc :
    - Plus ou moins de bits, plus il y a de bits plus on peut avoir de mots et moins il y a de risque d'erreur
    - Mais plus la résolution devra être élevée.

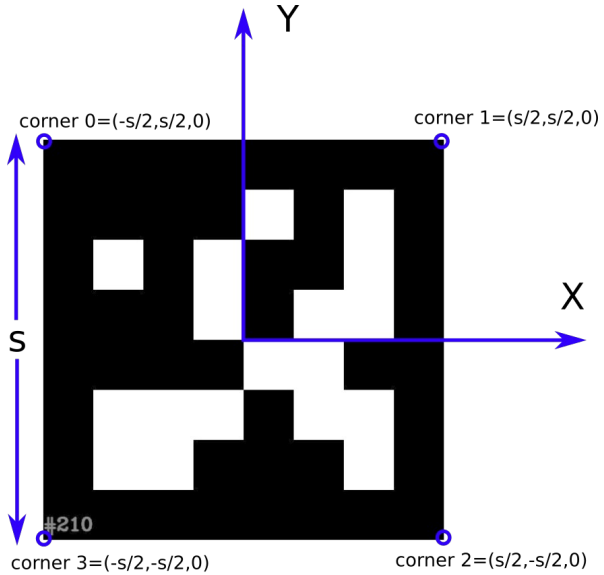


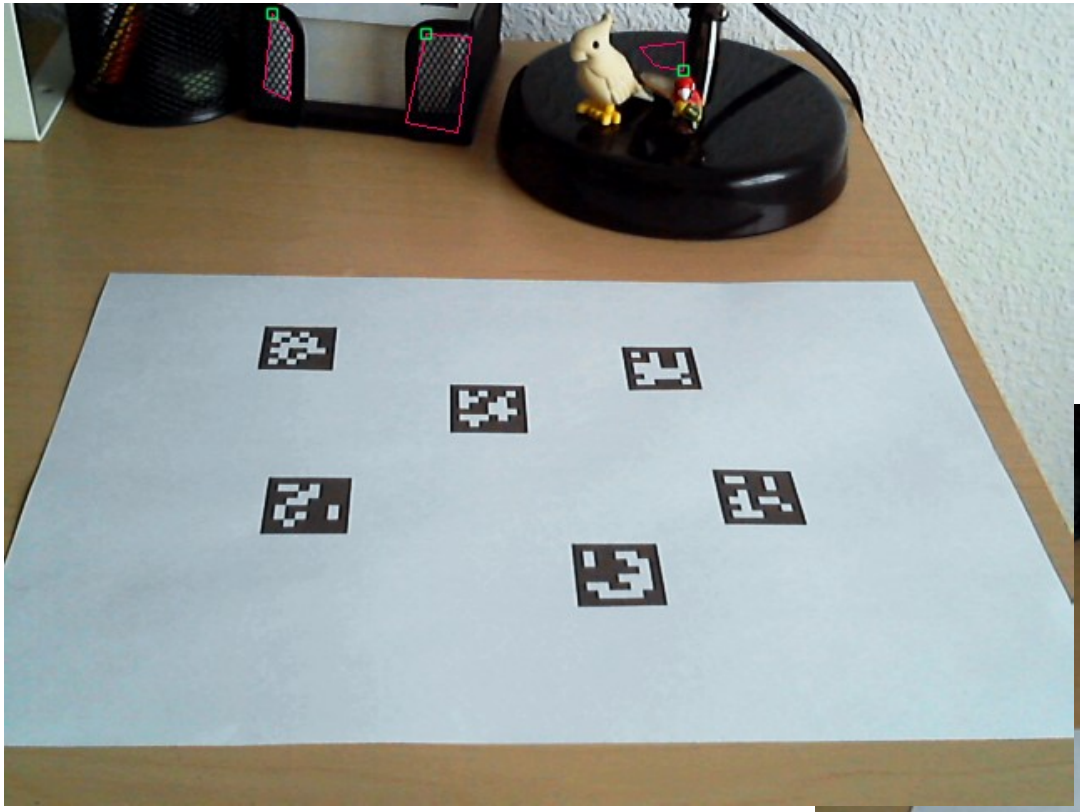
- Système de coordonnées
- S = la taille
- Configurations

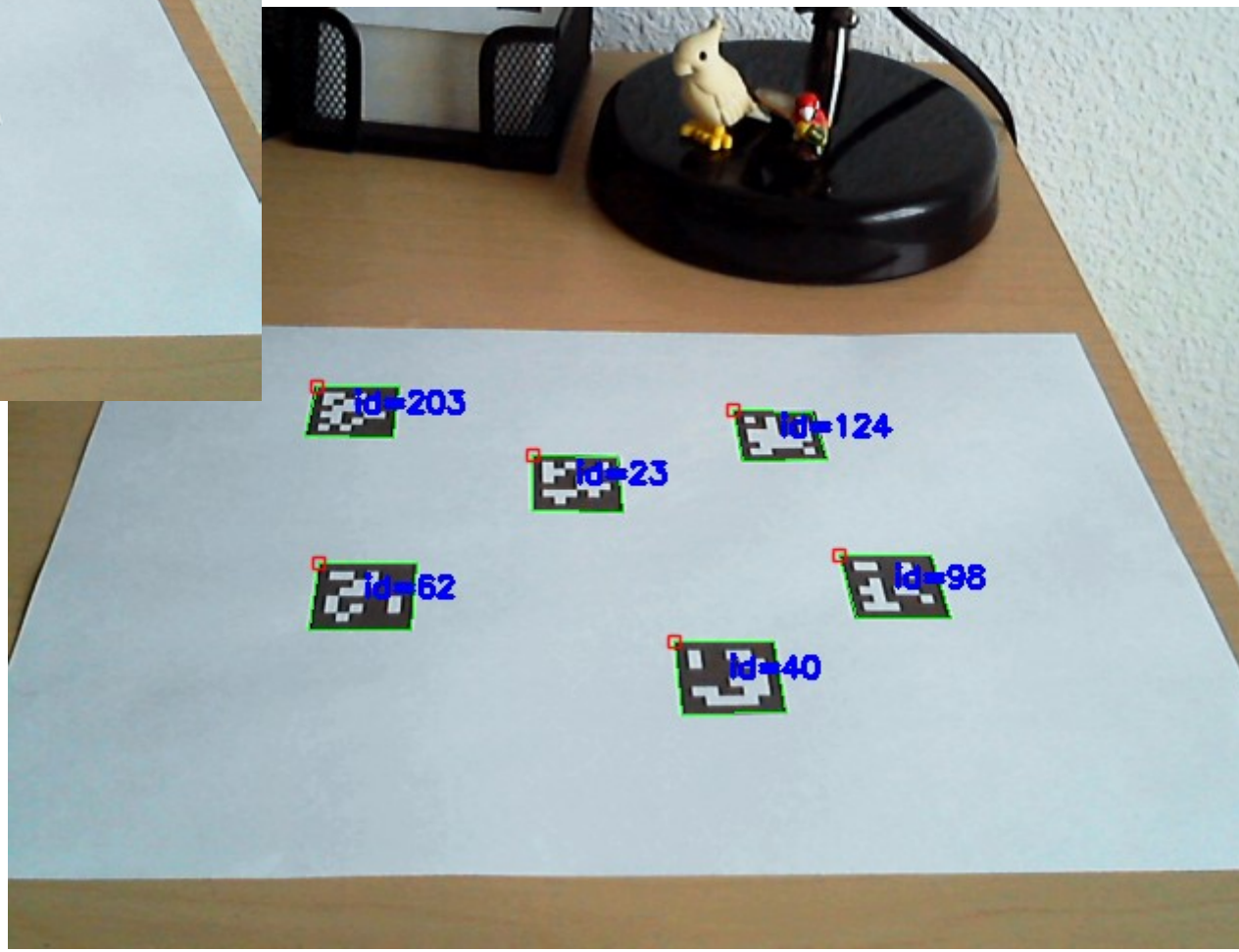
  stockées dans des dictionnaires

  – Permet d'identifier les configurations correctes

  – Permet d'associer un identifiant à chaque marqueur






id=203
id=124
id=23
id=62
id=98
id=40

Détection des marqueurs

- Passage en niveaux de gris
- Seuillage adaptatif pour détecter les points candidats contours
- Suivi des contours à partir des points candidats : il en reste beaucoup trop qu'on va éliminer...
- Suppression des contours avec trop peu de points
- Approximation polygonale du contour. Seuls les contours convexes avec exactement 4 coins sont conservés (i.e., rectangles)
- Suppression des rectangles trop proches : le seuil adaptatif détecte les contours extérieurs et intérieurs. On conserve les plus externes.

Seuillage adaptatif

- c un seuil fixé, wt largeur de fenêtre fixée
- Parcours de l'image pixel par pixel (algo //) :
  - Pour chaque fenêtre de taille wt, calcul de la moyenne m centré en p
  - Si le niveau de gris de p est supérieur à m-c alors le pixel résultat est à 1

    sinon 0

Détection des marqueurs

- Passage en niveaux de gris
- Seuillage adaptatif pour détecter les points candidats contours
- Suivi des contours à partir des points candidats : il en reste beaucoup trop qu'on va éliminer...
- Suppression des contours avec trop peu de points
- Approximation polygonale du contour. Seuls les contours convexes avec exactement 4 coins sont conservés (i.e., rectangles)
- Suppression des rectangles trop proches : le seuil adaptatif détecte les contours extérieurs et intérieurs. On conserve les plus externes.

Suivi des contours et élimination des petits segments

- Suzuki, S. and Abe, K., Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, pp 32-46 (1985)

  → findContours + élimination des petits segments

Détection des marqueurs

- Passage en niveaux de gris
- Seuillage adaptatif pour détecter les points candidats contours
- Suivi des contours à partir des points candidats : il en reste beaucoup trop qu'on va éliminer...
- Suppression des contours avec trop peu de points
- Approximation polygonale du contour. Seuls les contours convexes avec exactement 4 coins sont conservés (i.e., rectangles)
- Suppression des rectangles trop proches : le seuil adaptatif détecte les contours extérieurs et intérieurs. On conserve les plus externes.

Algorithme de Douglas-Peucher

- Algorithme récursif de simplification de polylignes

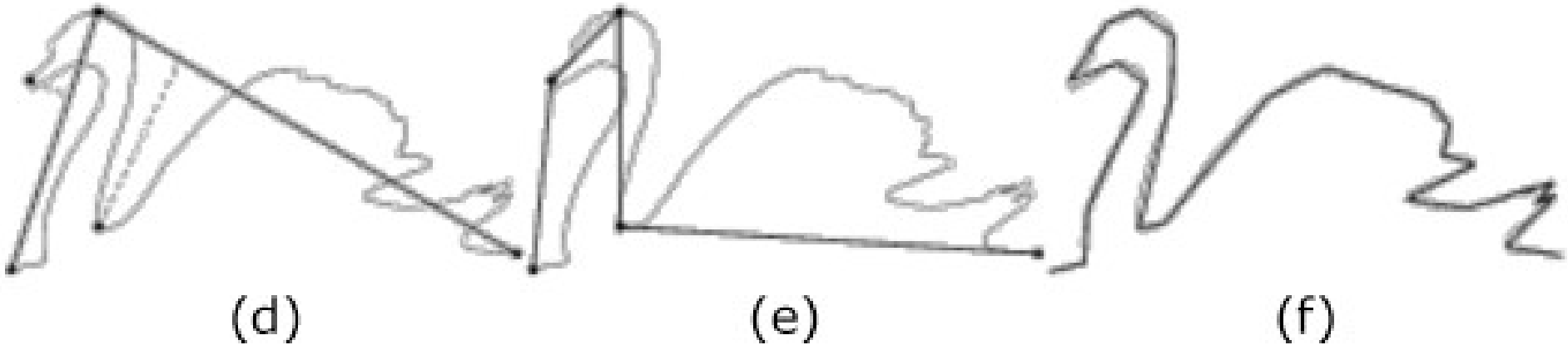


Figure 1: The basic Douglas-Peucker algorithm.

Douglas-Peucker

```
function DouglasPeucker(PointList[], epsilon)
  // Trouve le point le plus éloigné du segment
  index = indexPointPlusEloignéExtrémités(PointList[])

  // Si la distance dmax est supérieure au seuil, on simplifie
  if dmax > epsilon
    // Appel récursif de la fonction
    recResults1[] = DouglasPeucker(PointList[1…index], epsilon)
    recResults2[] = DouglasPeucker(PointList[index…end], epsilon)

    // Construit la liste des résultats à partir des résultats partiels
    ResultList[] = {recResults1[1…end-1] recResults2[1…end]}

  else
    // Tous les points sont proches → renvoie un segment avec les extrémités
    ResultList[] = {PointList[1], PointList[end]}
  end

  // Renvoie le nouveau résultat
  return ResultList[]
end
```

0. Départ

1. Distance b du point

   le plus éloigné (c) > seuil :

   Appel récursif sur

   - gauche = début:c

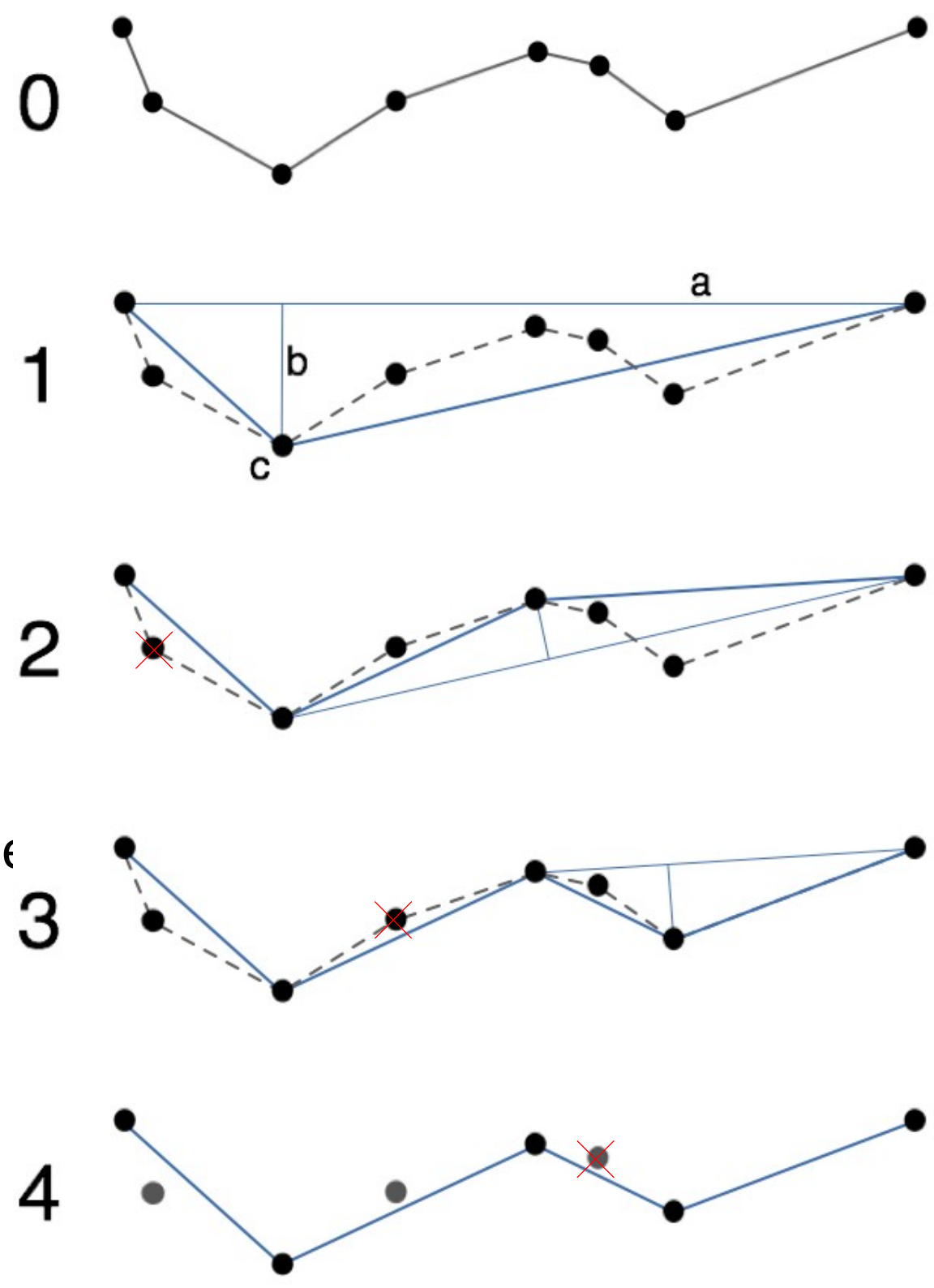
   - droit = c:fin

2. Suppression d'un point

   sur le segment gauche

   et appel récursif sur droite

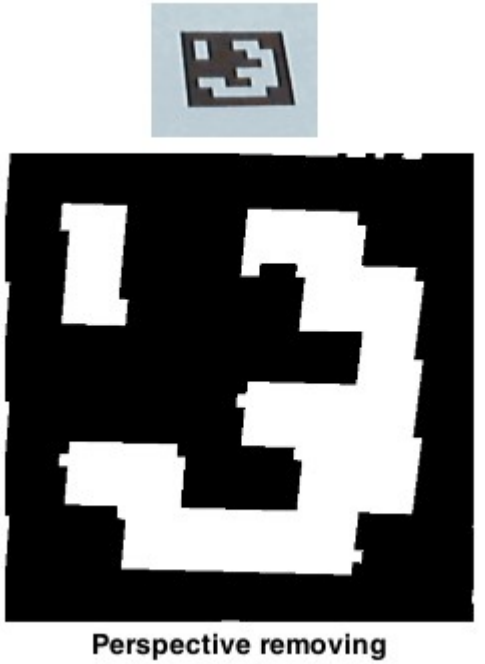
3. etc



Identification des marqueurs

- Redressement par homographie

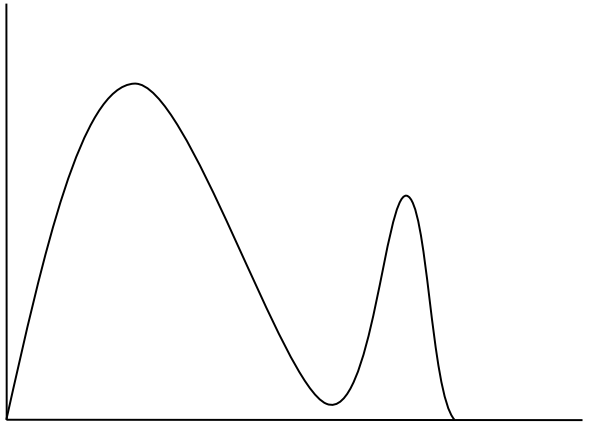
  et seuillage automatique par Otsu



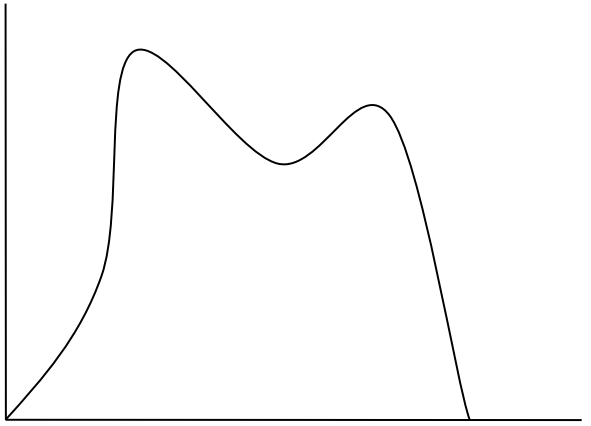
Segmentation automatique de l'histogramme

Principe

- Déterminer le seuil « optimal » en deux classes à partir de l'histogramme.



Deux modes séparables



Pas si clair…

Méthode d'Otsu

Sélection le seuil qui minimise la **variance intra-groupe** des deux classes (ou qui maximise la **variance inter-groupe**)

On commence :

On note P(i) l'histogramme normalisé (somme à 1)

Otsu

$\sigma_1^2(t)$ variance du groupe plus petit ou égal à t

$\sigma_2^2(t)$ variance du groupe plus grand que t

$q_1(t)$ probablité du groupe 1

$q_2(t)$ probabilité du groupe 2

$\mu_1(t)$ moyenne 1

$\mu_2(t)$ moyenne 2

σ_w^2 somme pondérée des 2 groupes (intra - groupe) :

$$\sigma_w^2(t) = q_1(t)\,\sigma_1^2(t) + q_2(t)\,\sigma_2^2(t)$$

avec

$$q_1(t) = \sum_{i=1}^{t} P(i), q_2(t) = \sum_{i=t+1}^{I} P(i)$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [\, i - \mu_1(t)\,]^2 P(i) / q_1(t)$$

$$\sigma_2^2(t) = \sum_{i=t+1}^{I} [\, i - \mu_2(t)\,]^2 P(i) / q_2(t)$$

$$\mu_1(t) = \sum_{i=1}^{t} i\, P(i) / q_1(t), \mu_2(t) = \sum_{i=t+1}^{I} i\, P(i) / q_2(t)$$

Otsu

- Un parcours systématique de t dans [0,255] permet de trouver $\sigma^2_w(t)$ minimal
- Mais on a une relation entre la variance intra-groupe $\sigma^2_w(t)$ et la variance totale $\sigma^2 = \sum_{i=1}^{I} (i-\mu)^2 P(i)$ *où* $\mu = \sum_{i=1}^{I} i\, P(i)$
- Et σ^2 ne dépend pas du seuil

Otsu

- On va pouvoir simplifier le calcul

$$\sigma^2 = \sum_{i=1}^{t}[i-\mu_1(t)+\mu_1(t)-\mu]^2 P(i) + \sum_{i=t+1}^{I}[i-\mu_2(t)+\mu_2(t)-\mu]^2 P(i)$$

$$= \sum_{i=1}^{t}\{[i-\mu_1(t)]^2 + 2[i-\mu_1(t)][\mu_1(t)-\mu] + [\mu_1(t)-\mu]^2\}P(i)$$

$$+ \sum_{i=t+1}^{I}\{[i-\mu_2(t)]^2 + 2[i-\mu_2(t)][\mu_2(t)-\mu] + [\mu_2(t)-\mu]^2\}P(i)$$

mais

$$\sum_{i=1}^{t}[i-\mu_1(t)][\mu_1(t)-\mu]P(i) = 0 \quad et$$

$$\sum_{i=t+1}^{I}[i-\mu_2(t)][\mu_2(t)-\mu]P(i) = 0$$

car

$$\sum_{i=1}^{t} [i - \mu_1(t)] [\mu_1(t) - \mu] P(i) = [\mu_1(t) - \mu] \sum_{i=1}^{t} [i - \mu_1(t)] P(i)$$

$$= [\mu_1(t) - \mu] \sum_{i=1}^{t} iP(i) - \sum_{i=1}^{t} \mu_1(t) P(i)$$

$$= [\mu_1(t) - \mu] \left[q_1(t)\mu_1(t) - \mu_1(t) \sum_{i=1}^{t} P(i) \right]$$

$$= [\mu_1(t) - \mu] [q_1(t)\mu_1(t) - \mu_1(t) q1(t)]$$

$$= 0$$

$$q_1(t) = \sum_{i=1}^{t} P(i) \quad \text{et} \quad q_2(t) = \sum_{i=t+1}^{I} P(i)$$

$$\sigma^2 = \sum_{i=1}^{t} [i - \mu_1(t)]^2 P(i) + [\mu_1(t) - \mu]^2 q_1(t)$$

$$+ \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 P(i) + [\mu_2(t) - \mu]^2 q_2(t)$$

$$= [q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)]$$

$$+ \{q_1(t)[\mu_1(t) - \mu]^2 + q_2(t)[\mu_2(t) - \mu]^2\}$$

Premier terme= σ^2_w.

Second terme σ^2_B=variance inter groupe . « distance moyenne des moyennes à la moyenne générale »

σ^2_B peut-être simplifié.

$$\mu = q_1(t)\mu_1(t) + q_2(t)\mu_2(t)$$

en replacant $q_2(t)$ *par* $1 - q_1(t)$ on a :

$$\sigma^2 = \sigma_w^2(t) + q_1(t)[1 - q_1(t)][\mu_1(t) - \mu_2(t)]^2 = \sigma_w^2(t) + \sigma_B^2(t)$$

Comme σ^2 ne dépend pas de t, il suffit de maximiser $\sigma_B^2(t)$

$$\sigma_B^2(t) = q_1(t)[1 - q_1(t)][\mu_1(t) - \mu_2(t)]^2$$

- Pour maximiser σ^2_B , on peut écrire récursivement

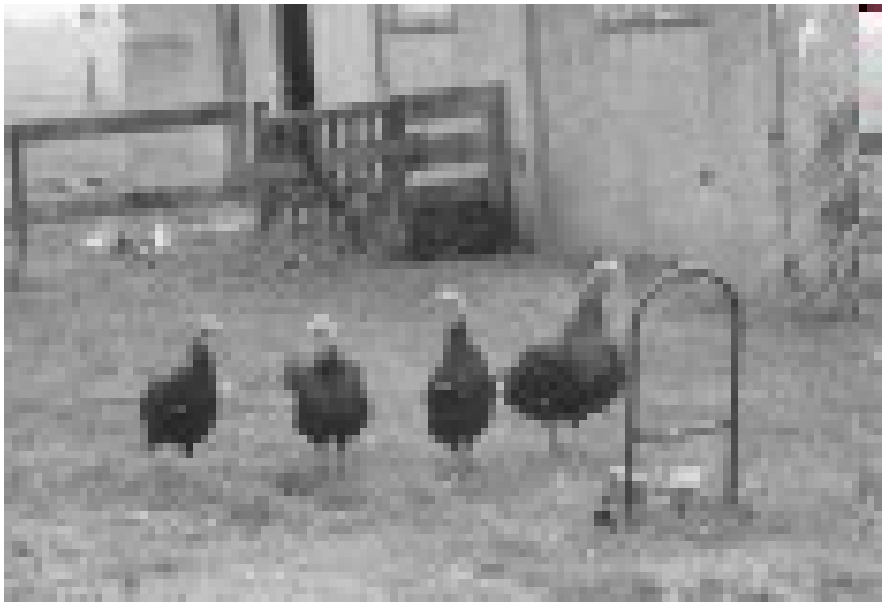
$$q_1(t+1) = q_1(t) + P(t+1) \qquad \text{avec} \quad q_1(1) = P(1)$$

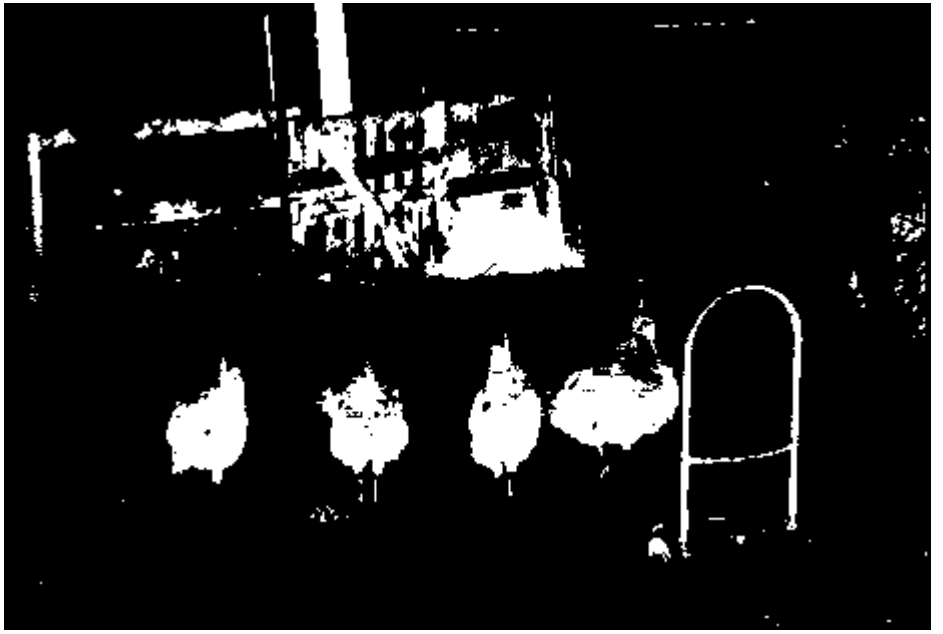
On obtient

$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)P(t+1)}{q_1(t+1)}$$

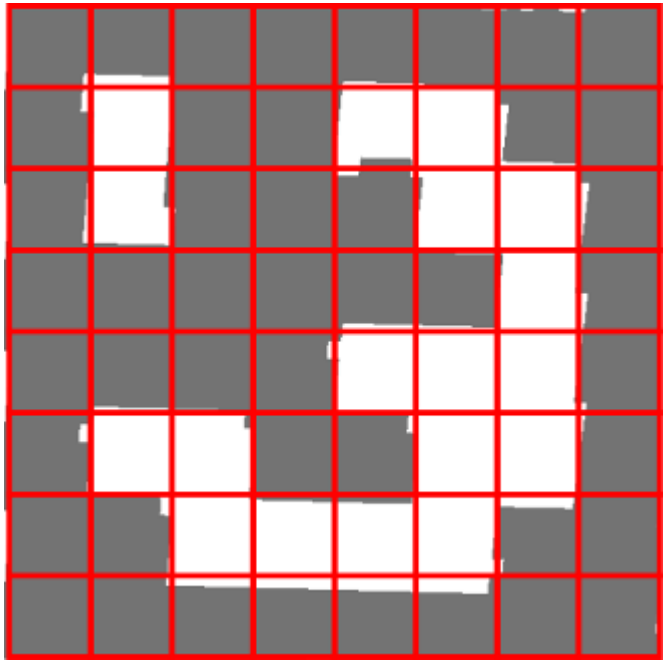
avec $\mu_1(0) = 0.$ *De même*

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

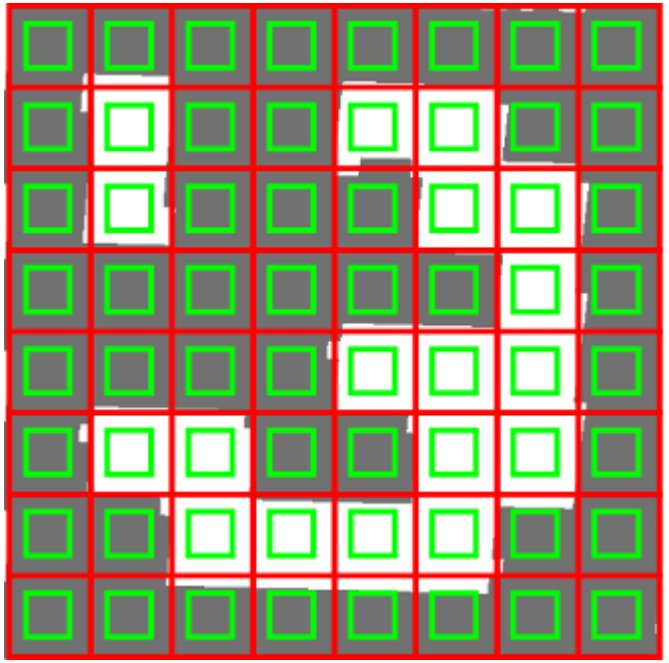


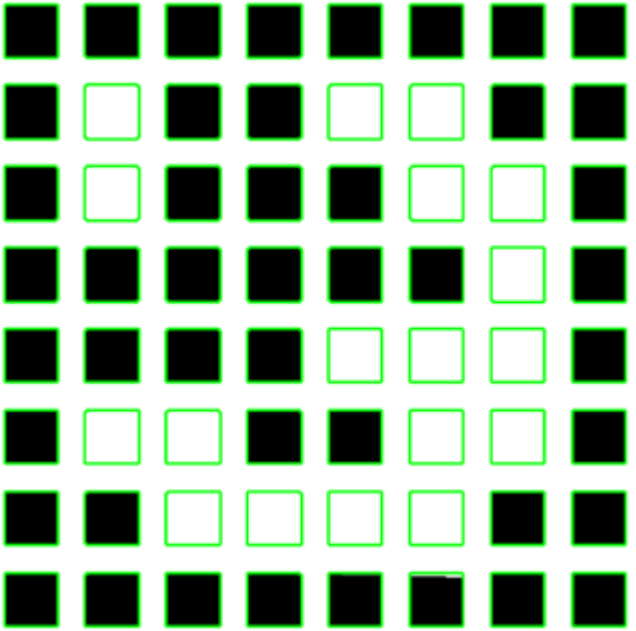


Conversion en code binaire

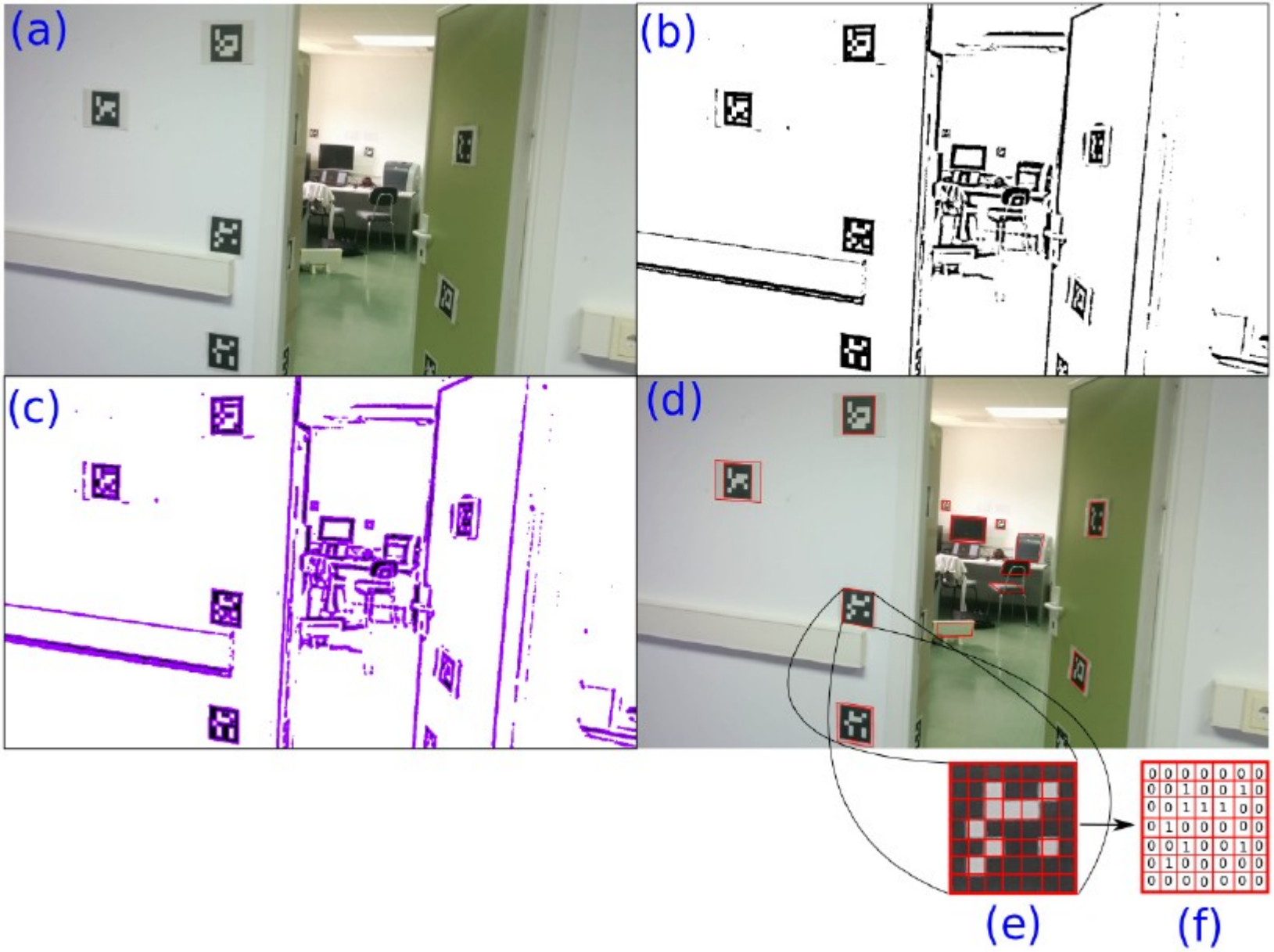


- Placement sur une grille
- Utilisation d'une marge pour robustifier



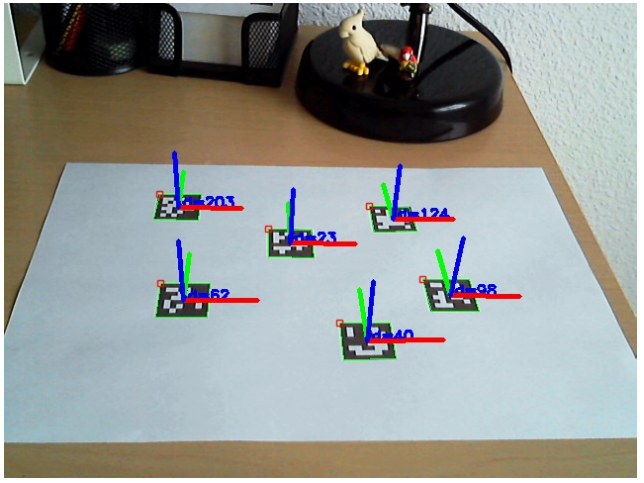


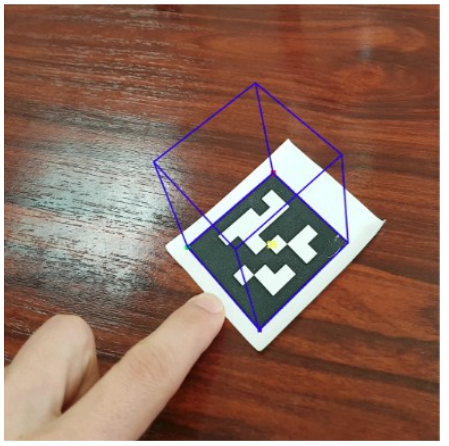
Résumé



Aruco et réalité augmentée

- A partir de la détection on peut estimer la transformation et donc placer des objets dans la scène :





- Mieux avec une

  carte de marqueurs :

  https://youtu.be/ilhN3c7RjCI