UNIVERSITÉ LAVAL

GLO-4001/7021
INTRODUCTION À LA ROBOTIQUE MOBILE

Capteurs Visuels

(Automne 2017)

Philippe Giguère

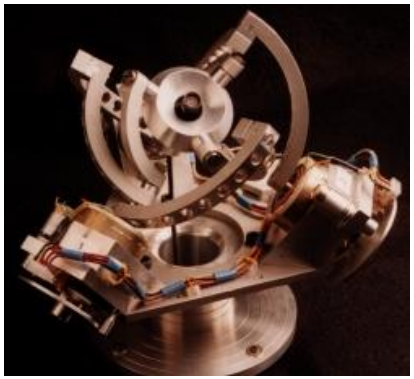
Pourquoi on s'intéresse à la vision en robotique mobile?

- Mot-clef #1 de la conférence IROS 2014 : vision
- Humains s'en servent (preuve que ça fonctionne)
- Capteur passif
- Retourne beaucoup d'information
  - millions pixels (œil humain ~550 Mpix dont ~7 Mpi fovéa)
  - position objet
  - couleur
  - texture
- Grande portée
- Environnements sont conçus pour employer la vision
  - panneaux indicateurs, etc.

Vision : problème difficile

- 40 ans de travail, résultats mitigés
- Utilise 50% des ressources du cerveau humain
  - indique grande complexité du problème
- Toujours pas de solution générale
  - image → monde 3D n'existe pas (tout à fait) encore
- En robotique, s'en servir pour des tâches spécialisées
  - se localiser avec repères visuel, détection d'obstacles, *etc.*
- Interpréter des parties de l'image, pas toute la scène
  - *active vision (*mécanique, traitement image, fovéa)

œil agile



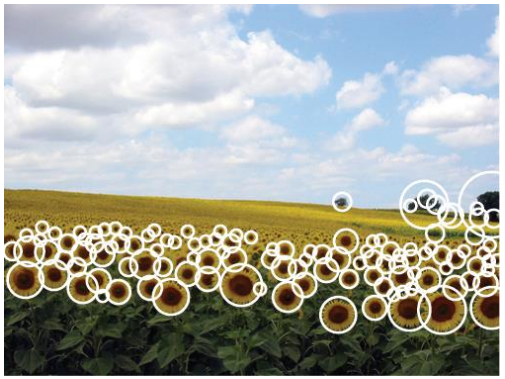
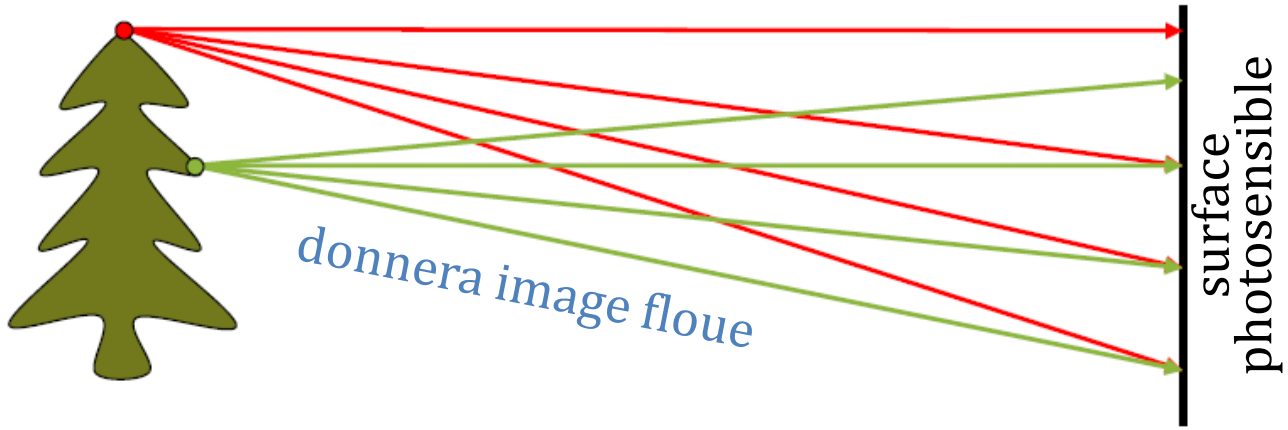


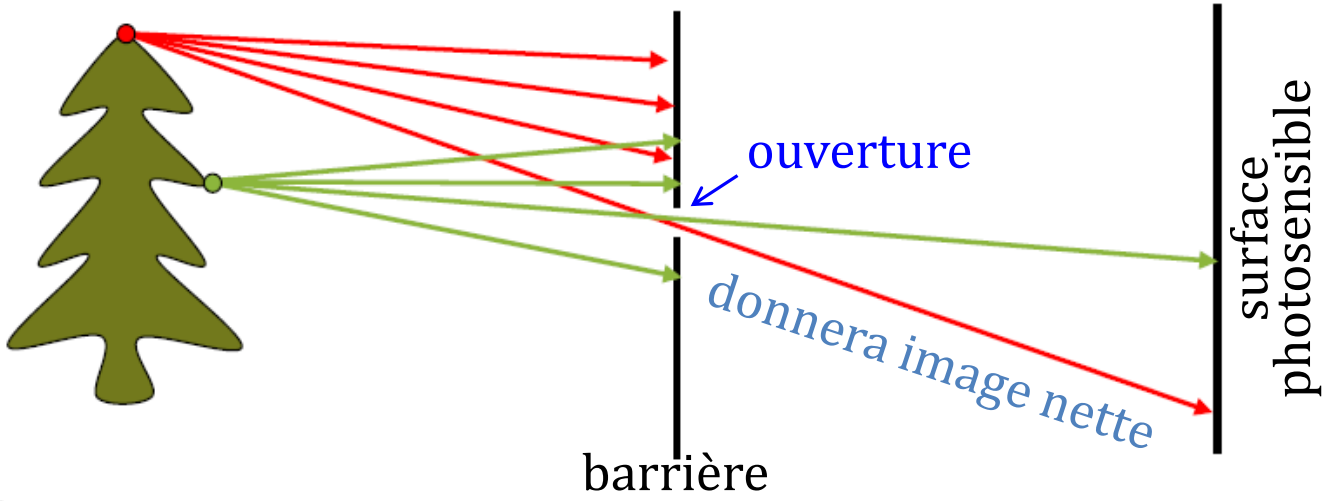
Image : processus de capture

- Il ne suffit pas de simplement avoir une surface photosensible :



plusieurs rayons incidents par « pixel »

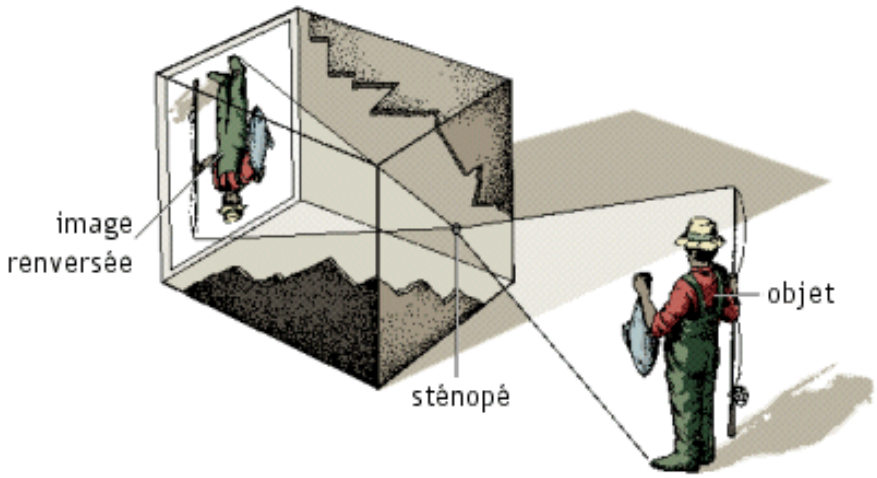
- Il faut plutôt bloquer la majorité des rayons :



un seul rayon incident par « pixel »

Modèle caméra à sténopé (*pinhole*)

- Approximer notre caméra + lentille comme une boîte avec un seul petit trou

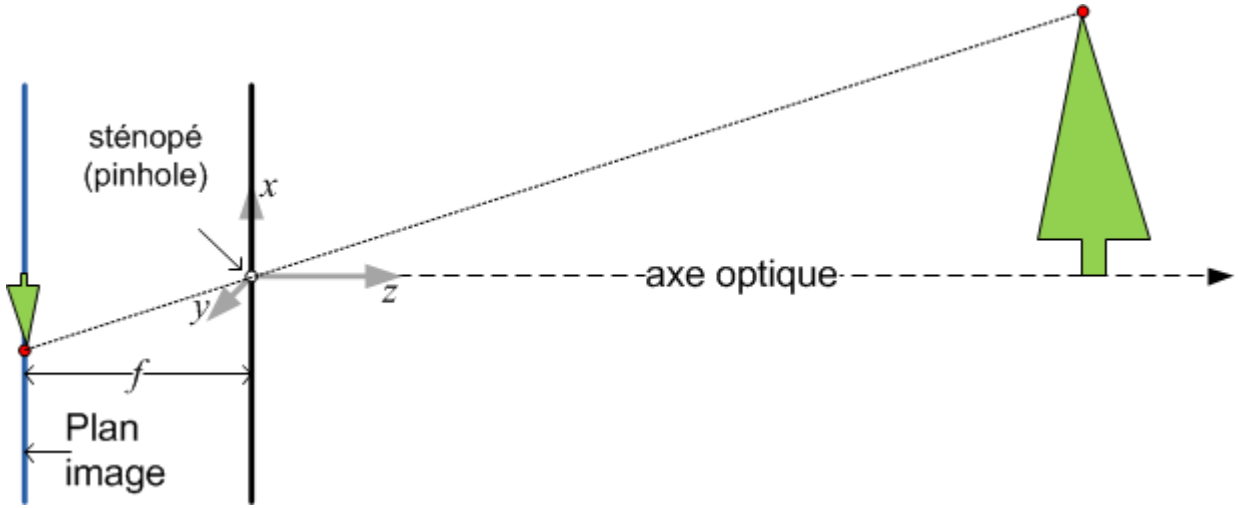


Principe de la chambre noire
Source : Encyclopédie Microsoft Encarta.

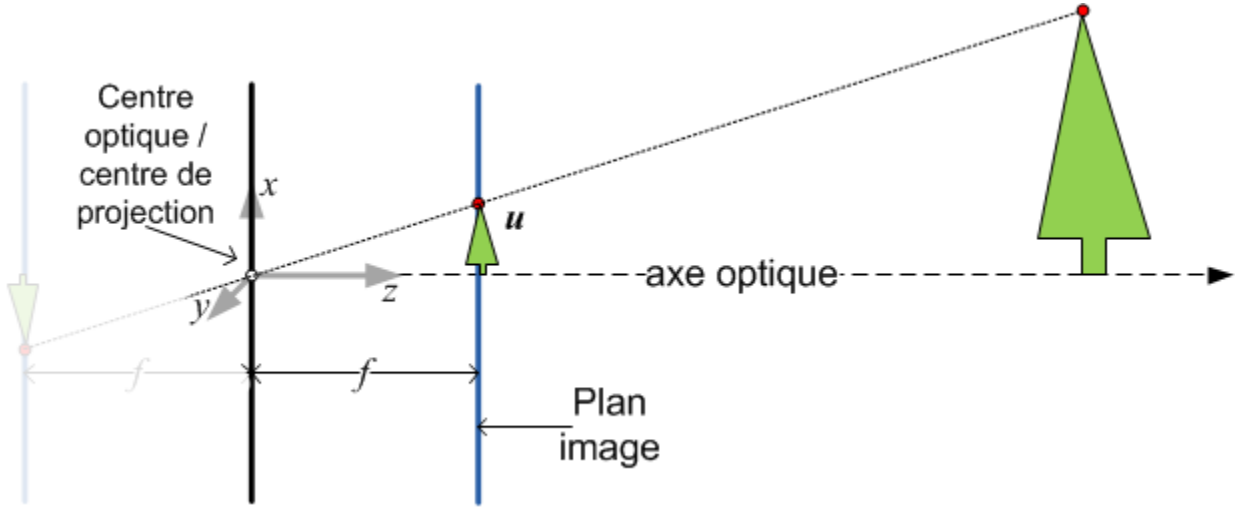
papier photo + caméra sténopé

Modèle de caméra sténopé

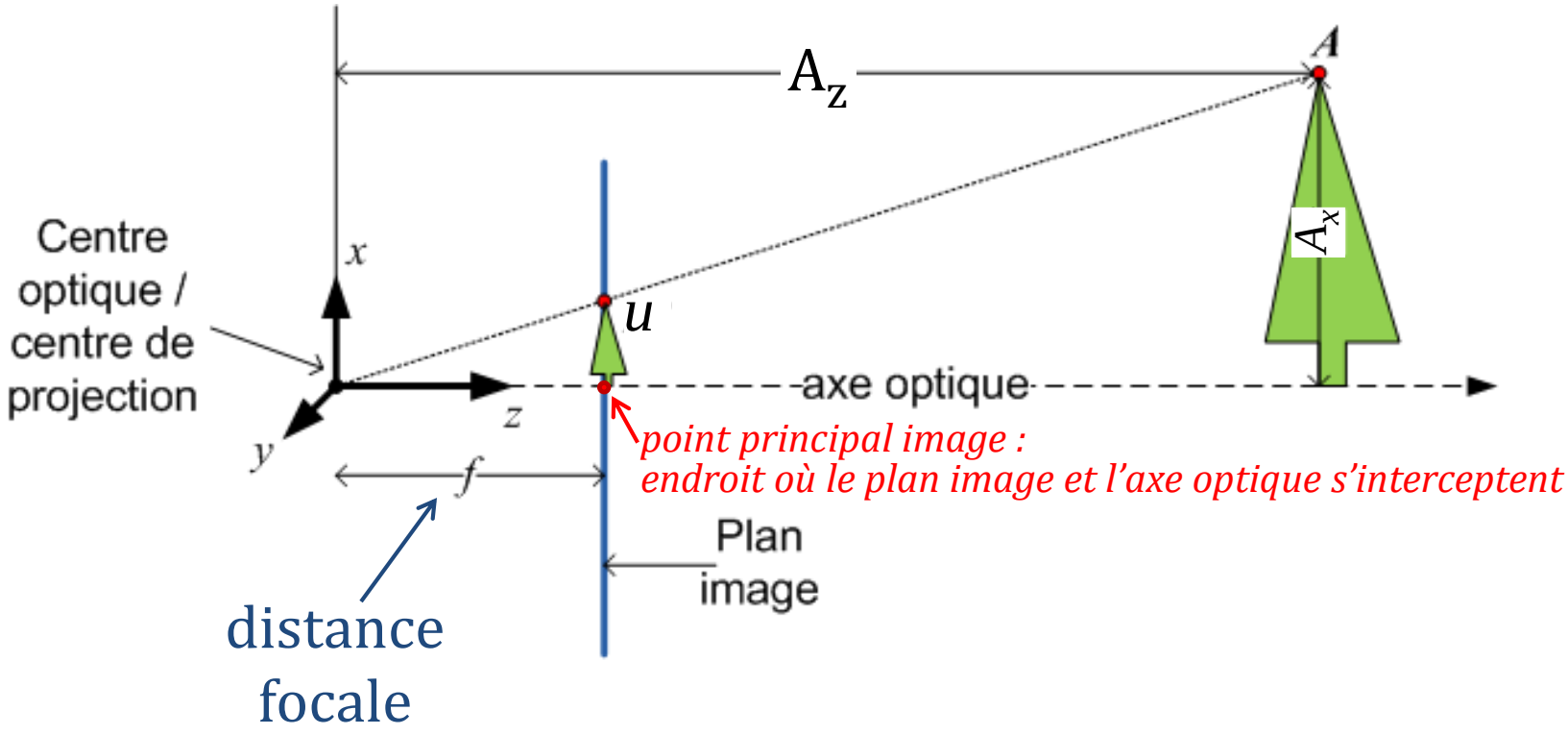
Caméra physique :
image formée à l'envers



Déplacer le plan image
en avant du trou :
image à l'endroit (mais
non-réalisable physiquement)



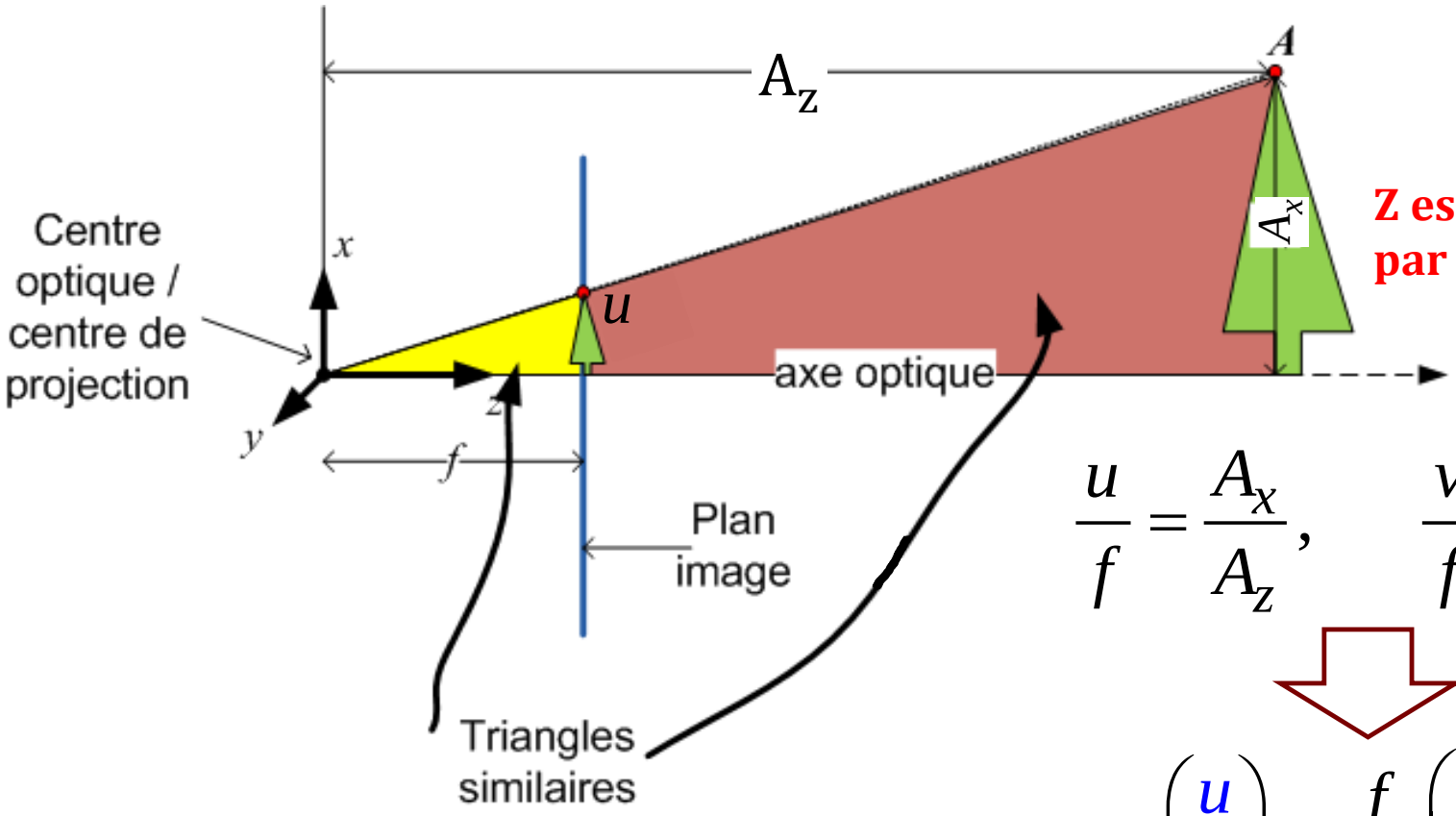
Modèle caméra sténopé : perspective



(par convention, les axes du plan image sont nommées *u* et *v*)

Caméra perspective : génération d'une image

Théorème de Thalès



Z est l'axe optique par convention

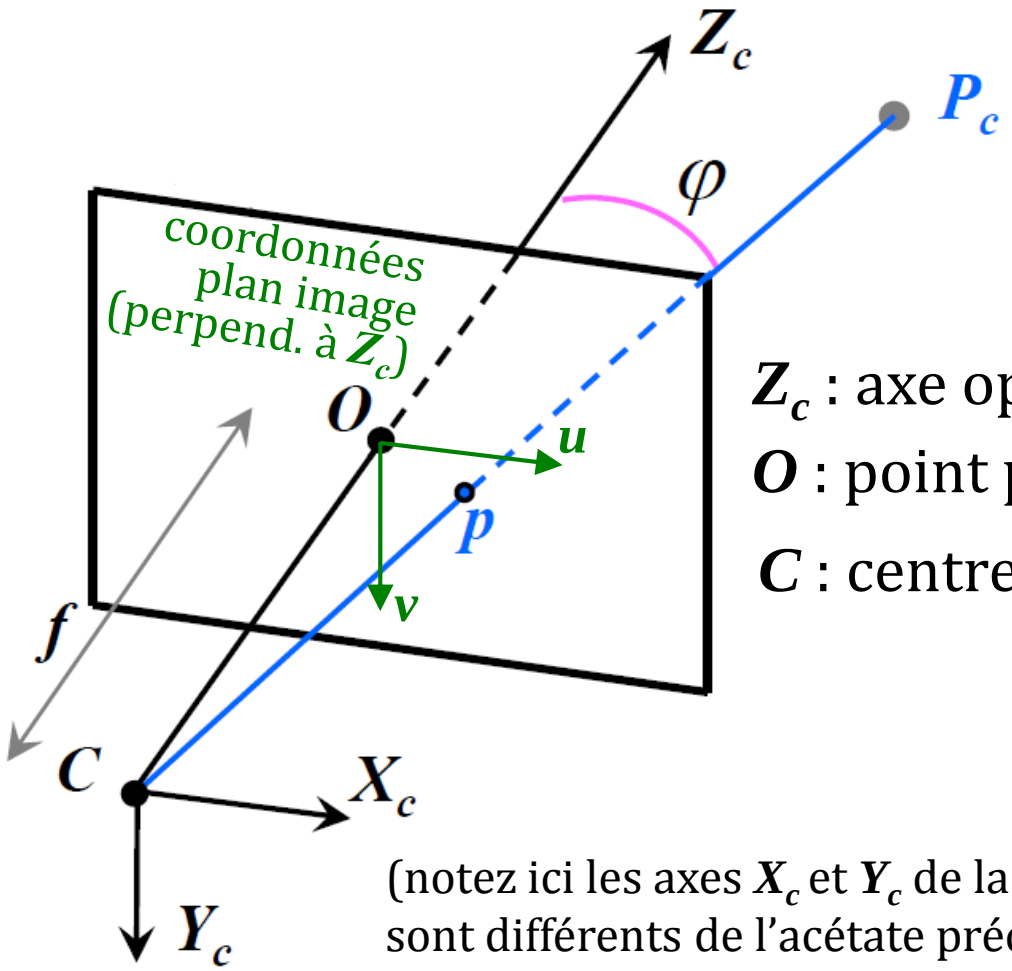
$$\frac{u}{f} = \frac{A_x}{A_z}, \qquad \frac{v}{f} = \frac{A_y}{A_z}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{f}{A_z} \begin{pmatrix} A_x \\ A_y \end{pmatrix}$$

coordonnées plan image

Plan image : autre illustration

Le plan image correspond à la surface photosensible (CCD, CMOS, film, etc.)



Z_c : axe optique

O : point principal

C : centre optique

(notez ici les axes X_c et Y_c de la caméra sont différents de l'acétate précédente. Simple convention)

Exemple de projection

- Vous avez un point situé aux coordonnées*

  $(\boldsymbol{A}_x=3, \boldsymbol{A}_y=0, \boldsymbol{A}_z=20)$, en mètres
- La distance focale $\boldsymbol{f}$ de la caméra est de 50 *mm*
- À quelles coordonnées (u,v)** du plan image, en *mm*, ce point apparaîtra-t-il?

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{f}{A_z}\begin{pmatrix} A_x \\ A_y \end{pmatrix} = \frac{50\ mm}{20\ m}\begin{pmatrix} 3\ m \\ 0\ m \end{pmatrix} = \begin{pmatrix} 7.5 \\ 0 \end{pmatrix} mm$$

- (Notez comment les unités en *m* s'annulent, et ne dépendent que des unités *mm* de la focale)

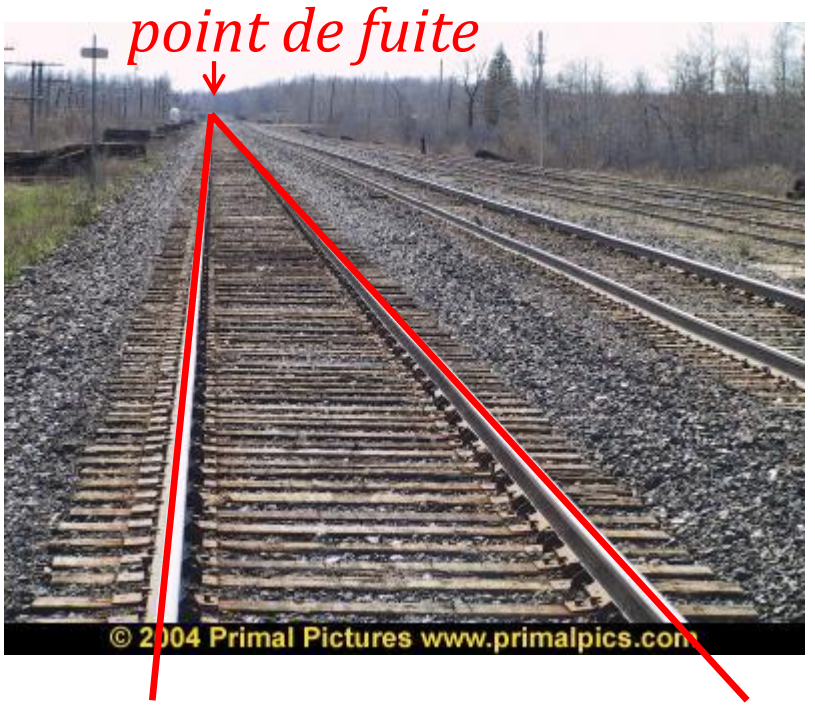
*coordonnées de la caméra, avec centre optique $\boldsymbol{C}$ à (0,0,0) et axe optique = $\boldsymbol{Z}_c$

**en supposant que l'origine de (u,v) est au point principal $\boldsymbol{O}$

UNIVERSITÉ LAVAL

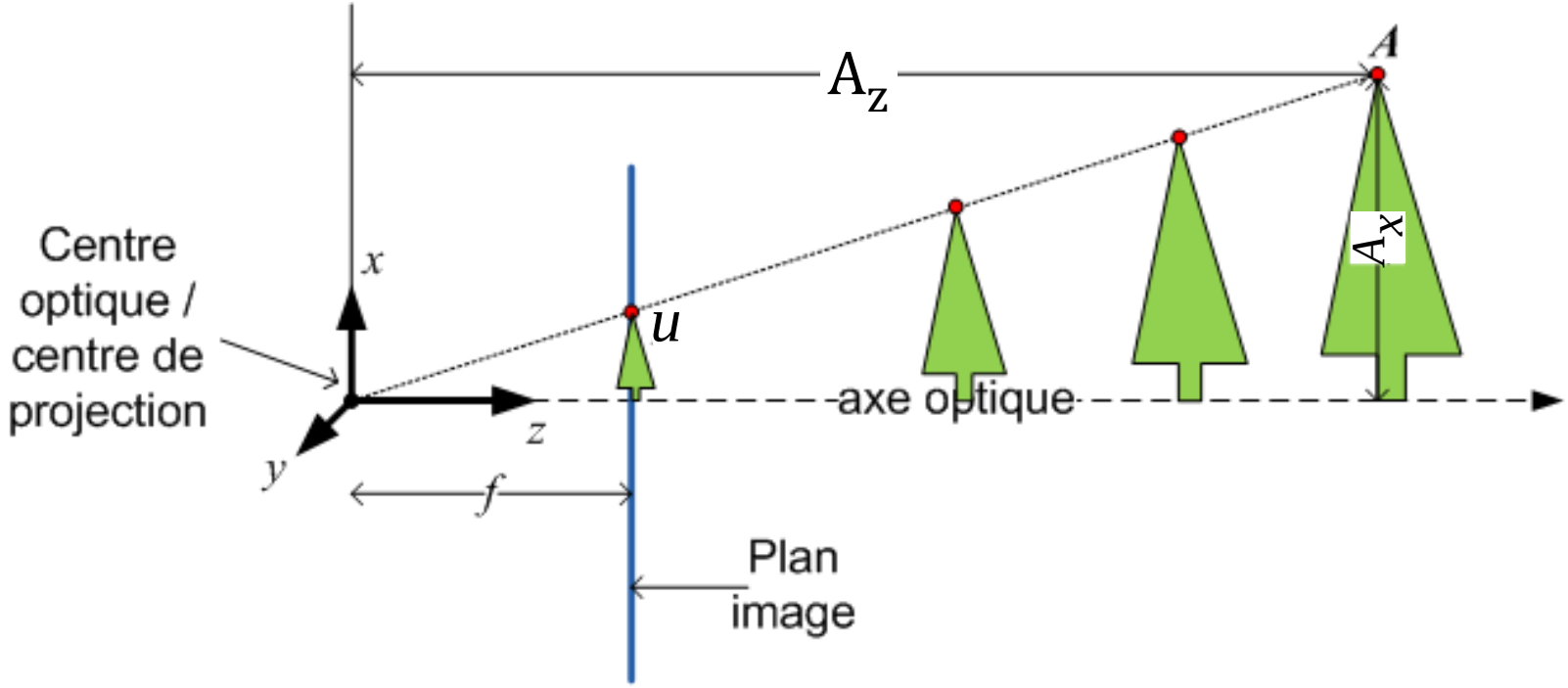
Caméra perspective : lignes

- Une ligne droite dans le monde apparaît comme une ligne droite dans le plan image.
- Des lignes parallèles vont se joindre au point de fuite (*vanishing point*)



Caméra 2D : perte de 3D

- Problème mal posé (plusieurs solutions pour l'inverse)



Jouer des tours…

Capteur numérique : plan image en pixel

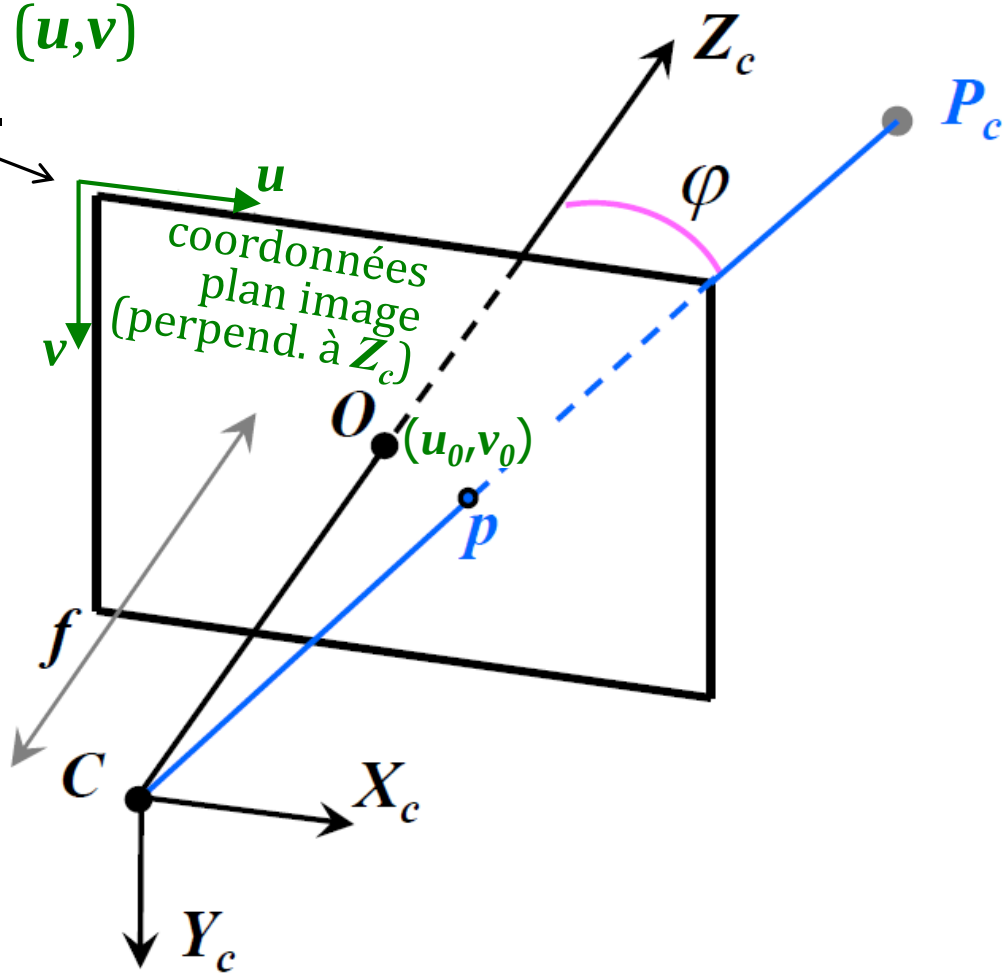
Pour une image numérique, l'origine (u,v) est souvent le coin supérieur gauche.
Le point principal O sera situé à la coordonnée (u_0,v_0).

Il faudra en tenir compte dans les équations :

$$u = u_0 + f\frac{A_X}{A_Z} \qquad v = v_0 + f\frac{A_Y}{A_Z}$$

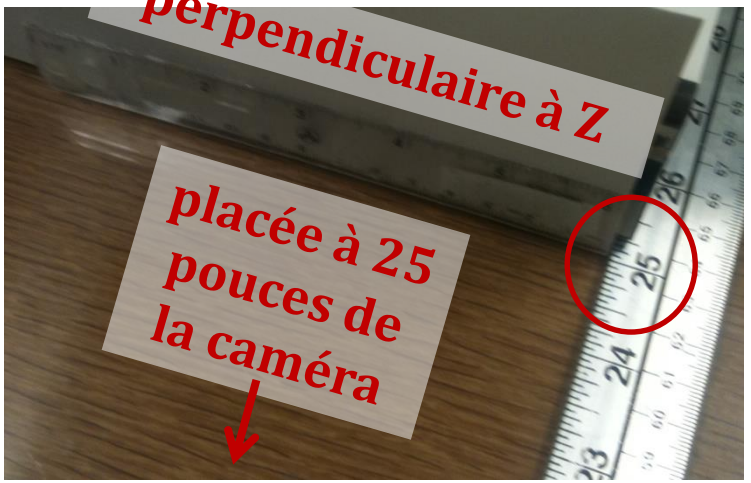
(avec f défini en pixel)

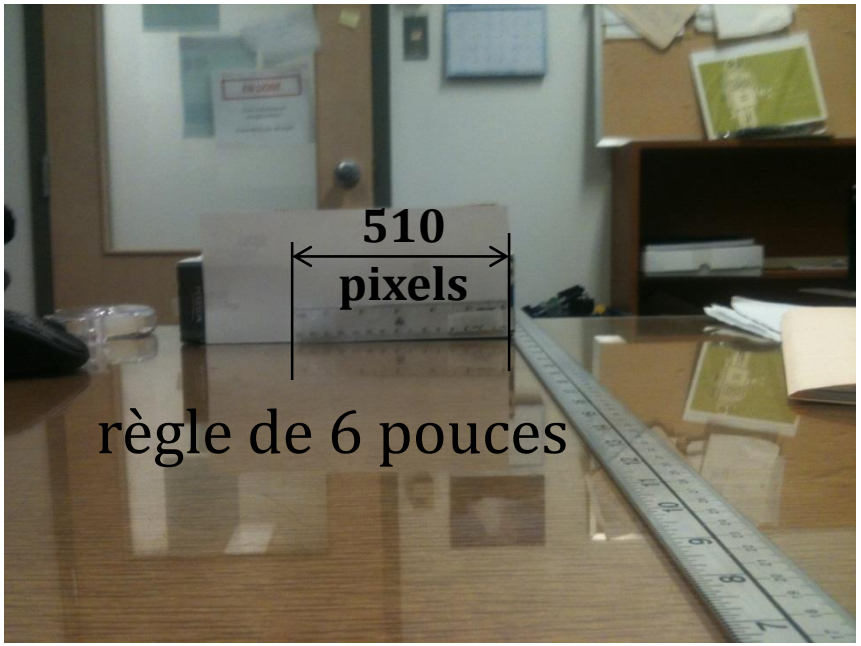
(aussi dans matlab, les fonctions comme `imagesc` ou `imshow` vont avoir les axes intervertis, et l'axe vertical est inversé)



Valeur de la focale *f*

- Cette valeur change d'une caméra à l'autre
- Sera constante (sauf si zoom optique)
- On l'identifie avec une calibration
- Calibration *rudimentaire** :





$$\frac{510\,pixels}{f} = \frac{6\,pouces}{25\,pouces}$$

$$f = \frac{25}{6}510\,pixels = 2125\,pixels$$

Note: on assume ici des pixels carrés sur la cellule

**Pour calibration complète, voir GIF-4100 Vision Numérique*

Coordonnées homogènes & Généralisation du modèle de projection

Math peu élégante

- Caméra est une fonction non-linéaire en A_z:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{f}{A_z} \begin{pmatrix} A_x \\ A_y \end{pmatrix}$$

- Chercher quelque chose plus élégant :

$$A' = PA$$

A' : coordonée plan image

A: coordonnées point 3D

P : modèle de caméra

Coordonnées homogènes

homogène ←→ cartésien

- Représentation point 2D avec 3 composantes:

  $\mathbf{x} = (x_1, x_2, x_3) \leftrightarrow (x_1/x_3,\ x_2/x_3)$

  $(3, 2) \rightarrow (3, 2, 1)$ ou $(6, 4, 2)$ ou… en homogène

- Représentation point 3D avec 4 composantes:

  $\mathbf{x} = (x_1, x_2, x_3, x_4) \leftrightarrow (x_1/x_4,\ x_2/x_4,\ x_3/x_4)$

  $(2, 5, 7) \rightarrow (2, 5, 7, 1)$ ou $(6, 15, 21, 3)$ ou … en homogène

(Sera très pratique pour rotations et translation, un peu plus tard…)

p 54-57 du manuel

Caméra perspective en coord. homogène

- ~ : signifie **identique à un facteur d'échelle**

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \sim \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}$$

position dans le plan image

coordonnées d'un point dans l'espace 3D

$$A' \sim P \quad A$$

(matrice intrinsèque)

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

coordonnées homogènes

Dérivation de la matrice P perspective

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fA_x / A_z \\ fA_y / A_z \\ 1 \end{pmatrix} = \frac{f}{A_z} \begin{pmatrix} A_x \\ A_y \\ A_z / f \end{pmatrix} \sim \begin{pmatrix} A_x \\ A_y \\ A_z / f \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ A_z \\ 1 \end{pmatrix}$$

Thalès — factorise — échelle

Caméra perspective : exemple

- Exemple

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

coordonnées homogènes

Point dans l'espace à $[4\ 6\ 2]^T \rightarrow [4\ 6\ 2\ 1]^T$

focale $f = 0.1$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ 6 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \\ 20 \end{bmatrix}$$

Passe d'homogène en cartésien dans plan image $[4\ 6\ 20]^T \rightarrow [0.2\ 0.3]^T$

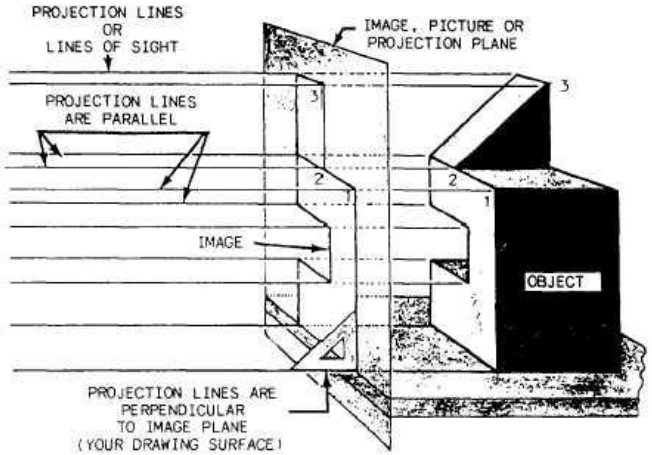
Autres types de projection de caméra

- Perspective faible:
  - distance moyenne $\bar{z}$

satellite altitude 20,000 *km*

montagne 4 *km*

- Orthographique :
  - rayons parallèles



$$A' \sim P \quad A$$

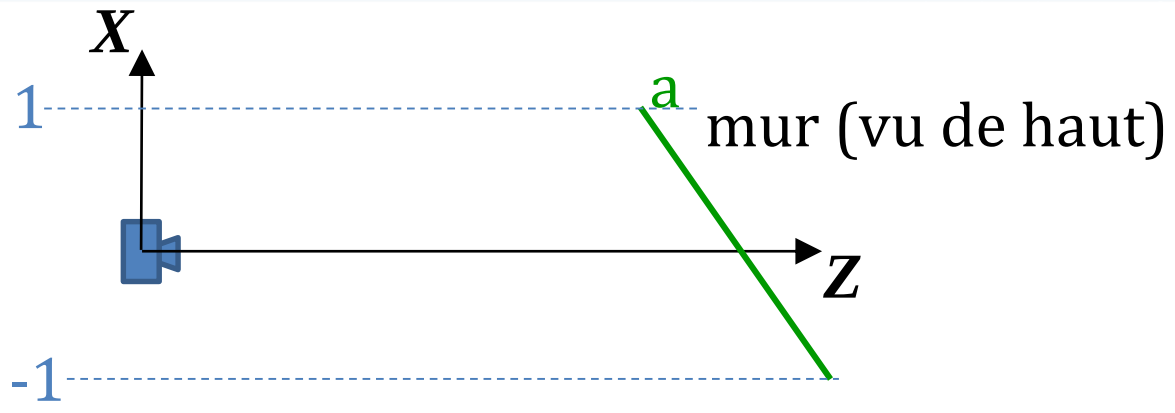
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \bar{z}/f \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

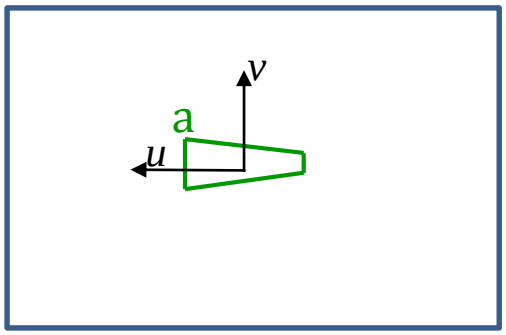
distance A_z n'intervient plus

$$A' \sim P \quad A$$

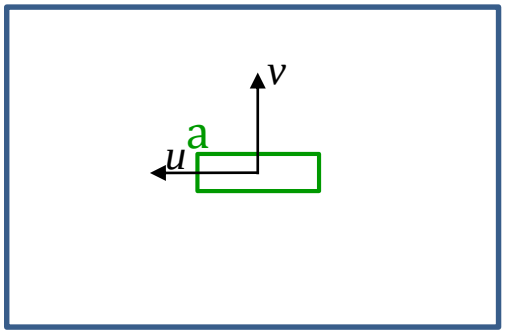
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

Exemples de projection

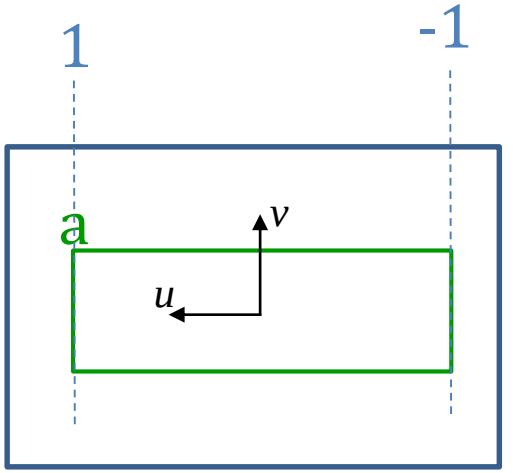




Perspective



Perspective faible



Orthographique

Localisation en 2D par caméra

Note: pour simplifier le problème, on fait l'hypothèse que le robot est sur un plancher plat, et que tous les points de repère sont à la hauteur de la caméra du robot

Localisation par caméra (en 2D)

Vous avez une carte avec repères l_i

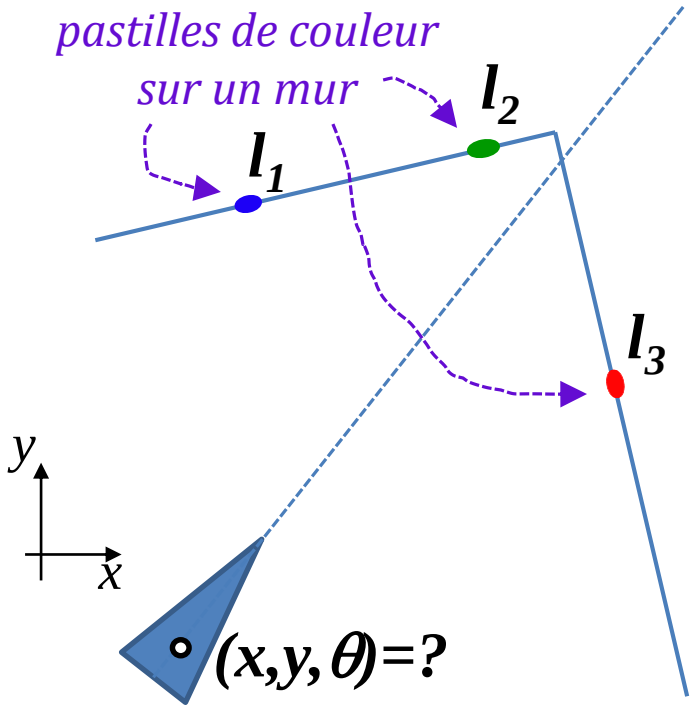
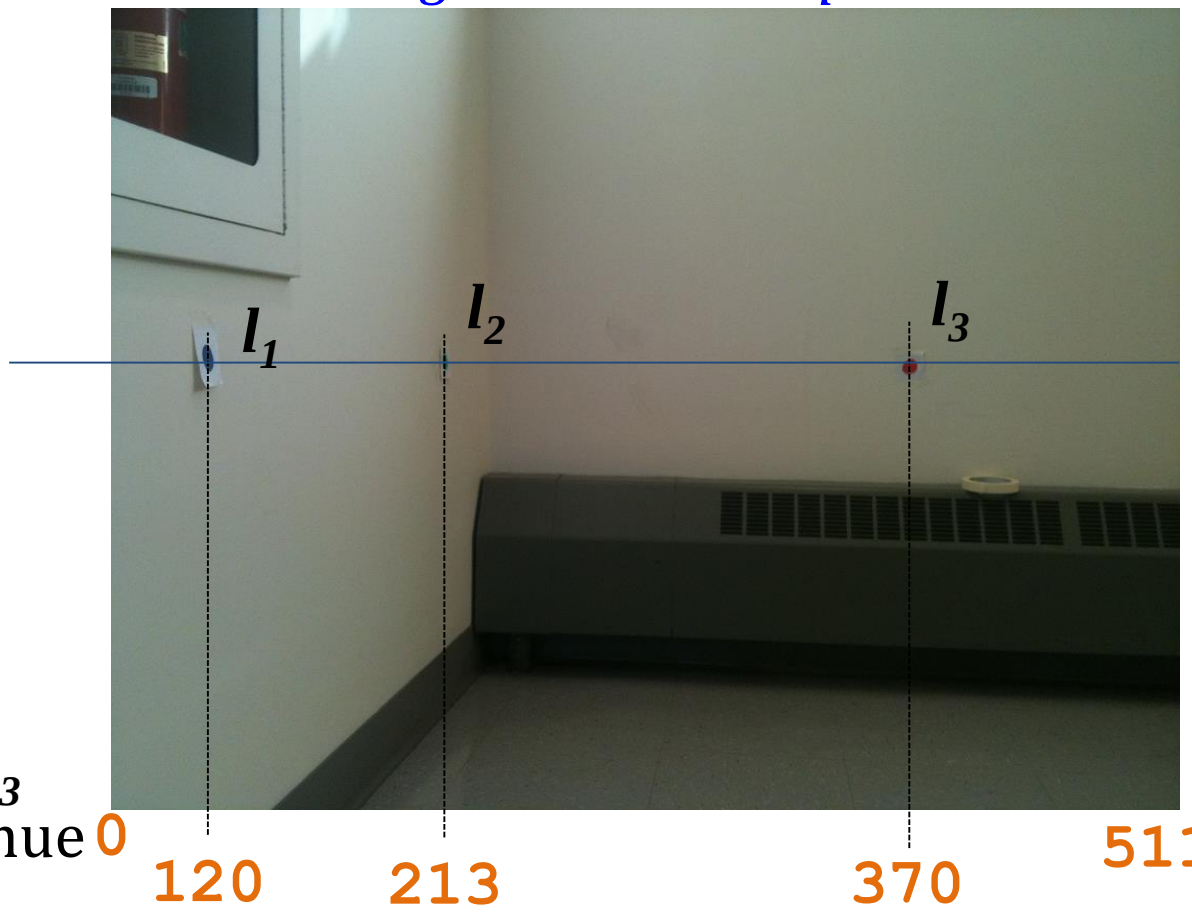


image de 512x384 pixels



On connait la position de l_1, l_2, l_3

Robot a une pose (x,y,θ) inconnue

Sa caméra prend une photo

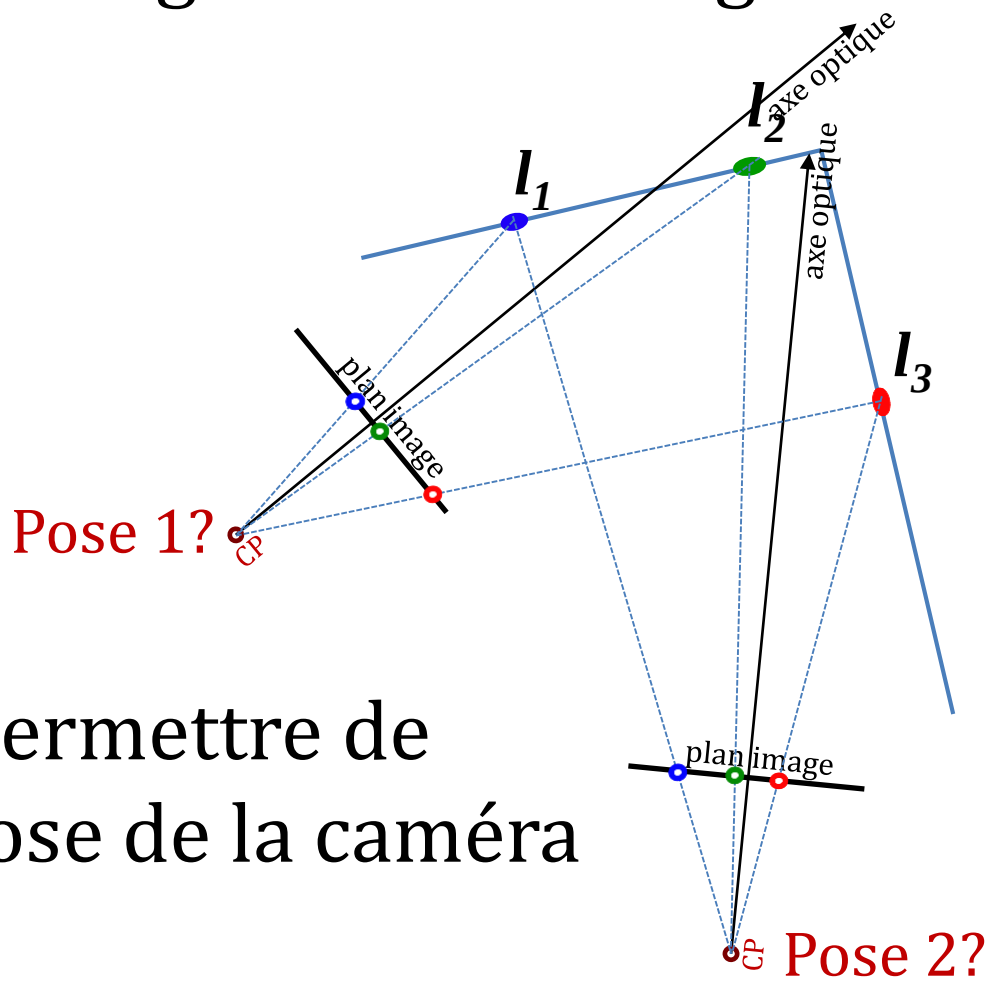
Retrouve les points de repères visuels dans l'image *(data association problem)*

i.e. position horizontale en pixel de l_1, l_2, l_3 pour ce problème en 2D

On veut retrouver x, y et θ à partir des positions en pixels de l_1, l_2, l_3

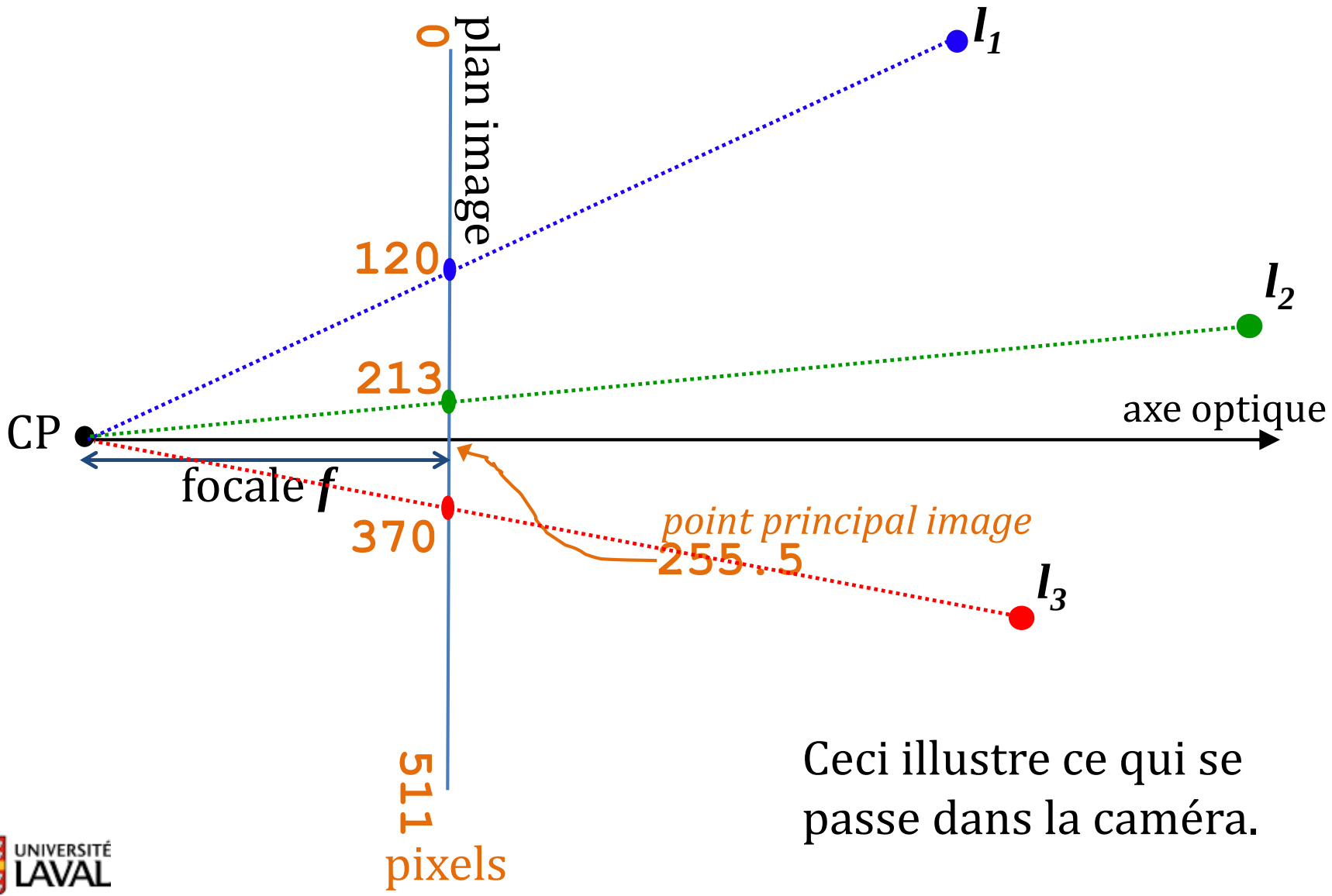
Localisation par caméra (en 2D)

- Pour notre problème, on se fie sur le fait que chacune des poses génère une image différente



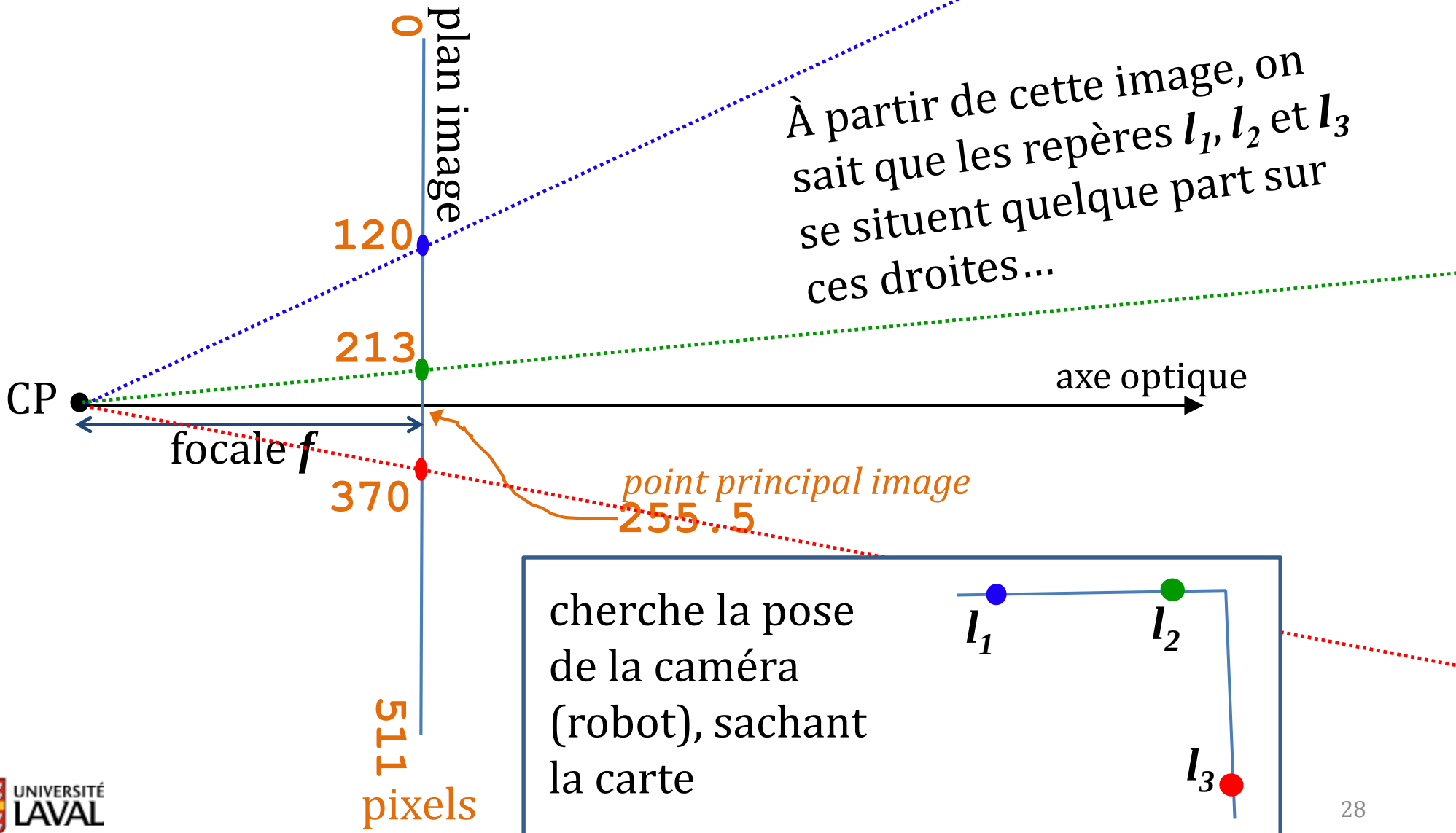
- Cela va nous permettre de retrouver la pose de la caméra

Caméra à sténopé : prise de la photo



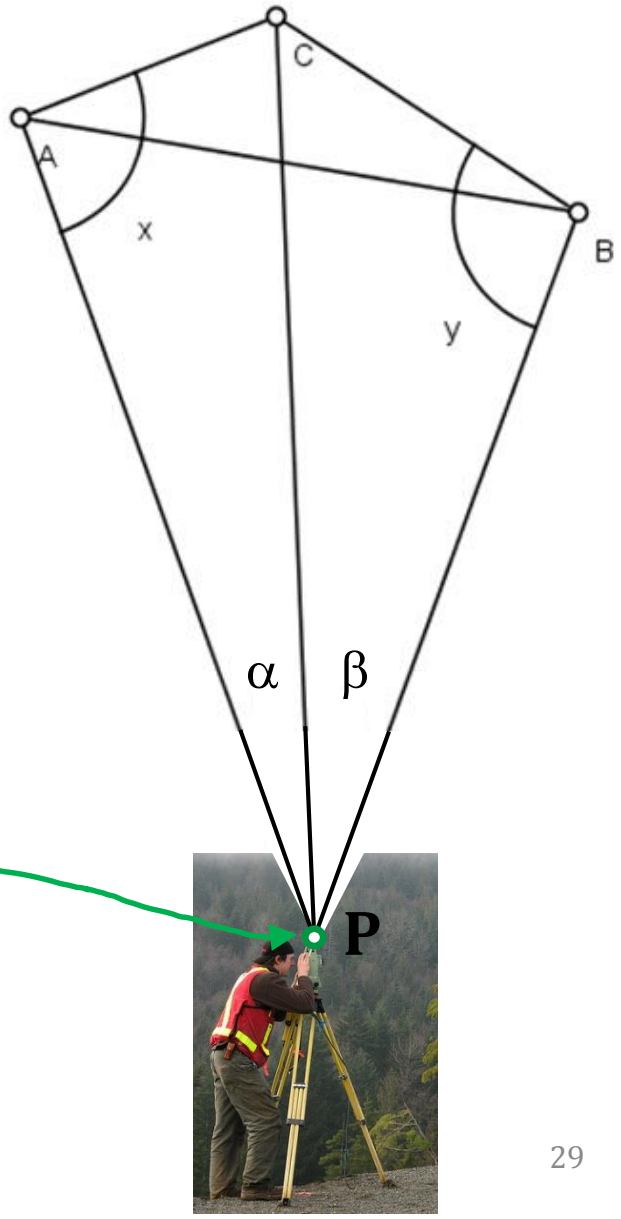
Ceci illustre ce qui se passe dans la caméra.

Caméra à sténopé : prise de la photo



Le problème Snellius-Pothenot

- Étudié en 1615 en arpentage
- Soient 3 points, A B et C, avec coordonnées connues sur une carte
- Soient deux angles mesurés à ***P***
  - α et β
- **Trouver position de P**
- Commençons par trouver α et β…

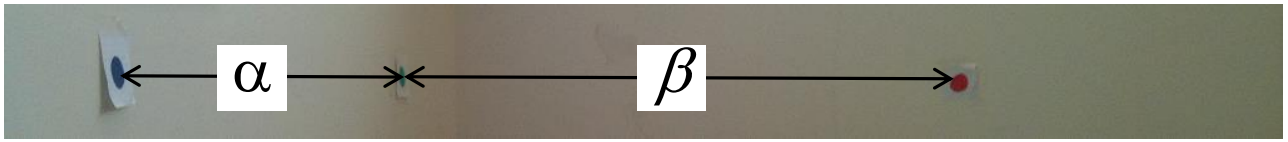


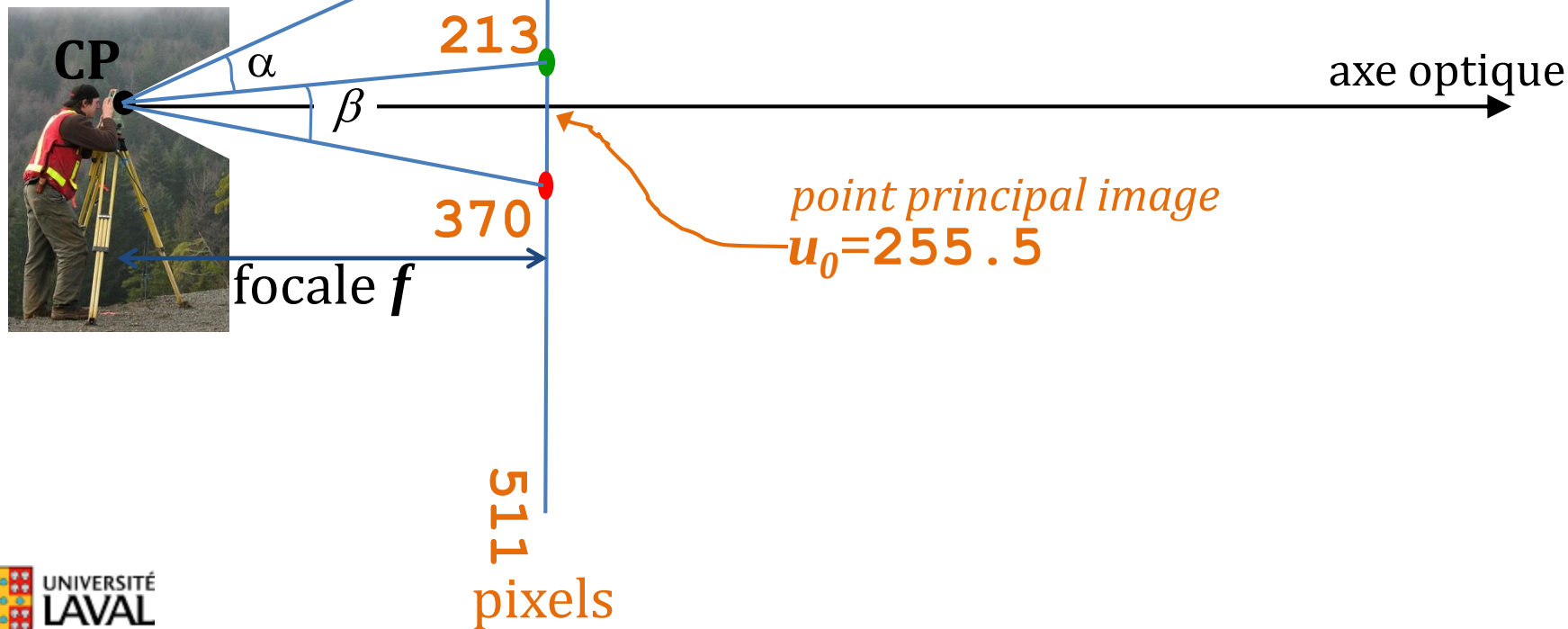
http://en.wikipedia.org/wiki/Snellius-Pothenot_problem

Caméra == rapporteur d'angles

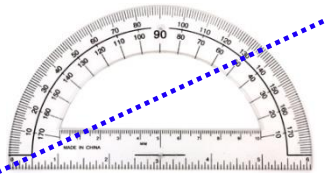
On va calculer des angles à partir des pixels de l'image

Note : valide seulement pour notre cas où les repères sont tous à la même hauteur



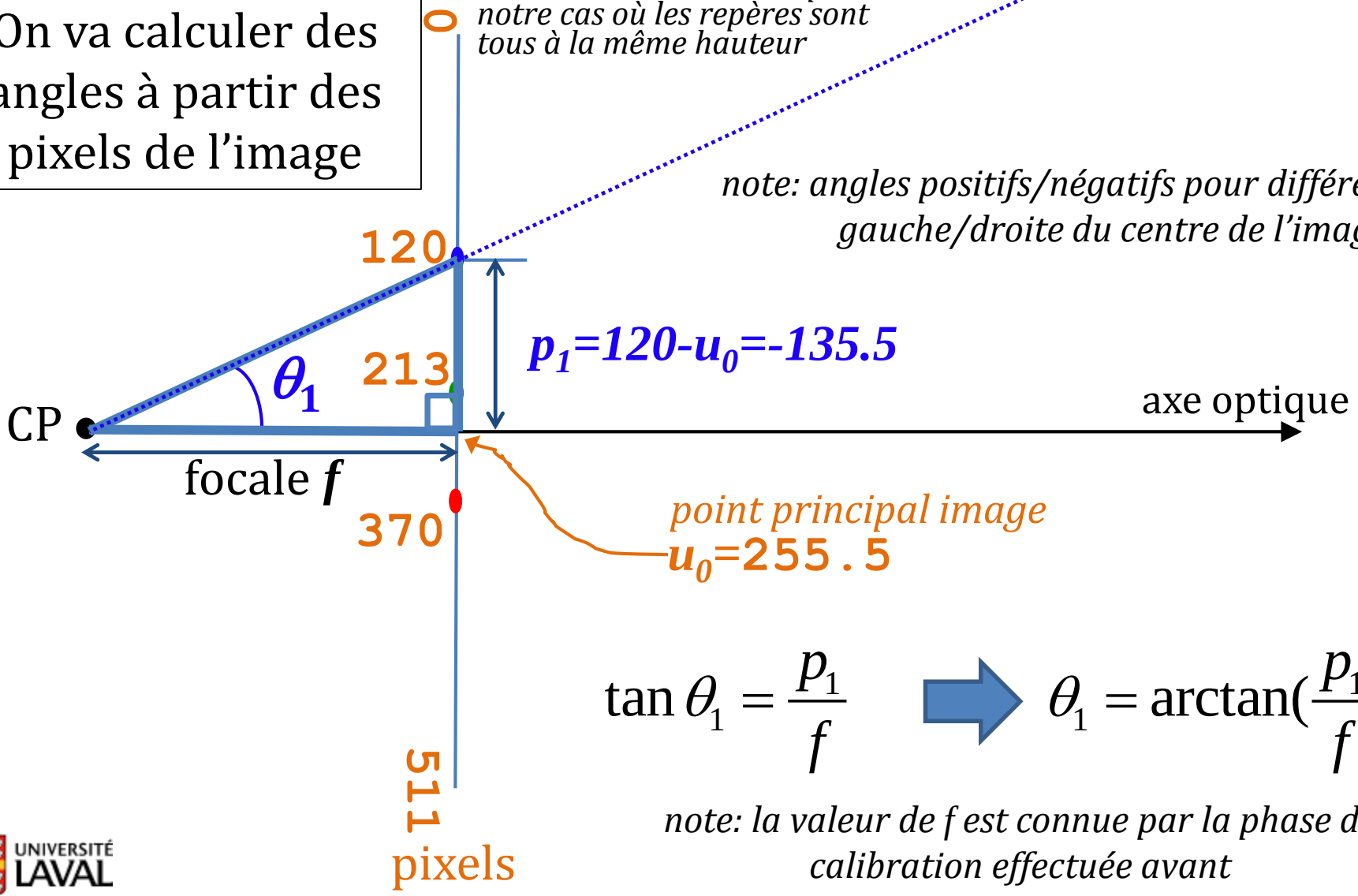


Caméra == rapporteur d'angles



On va calculer des angles à partir des pixels de l'image

Note : valide seulement pour notre cas où les repères sont tous à la même hauteur

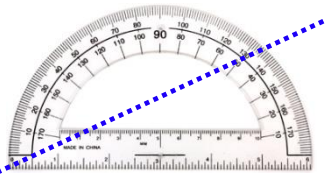


note: angles positifs/négatifs pour différentier si à gauche/droite du centre de l'image

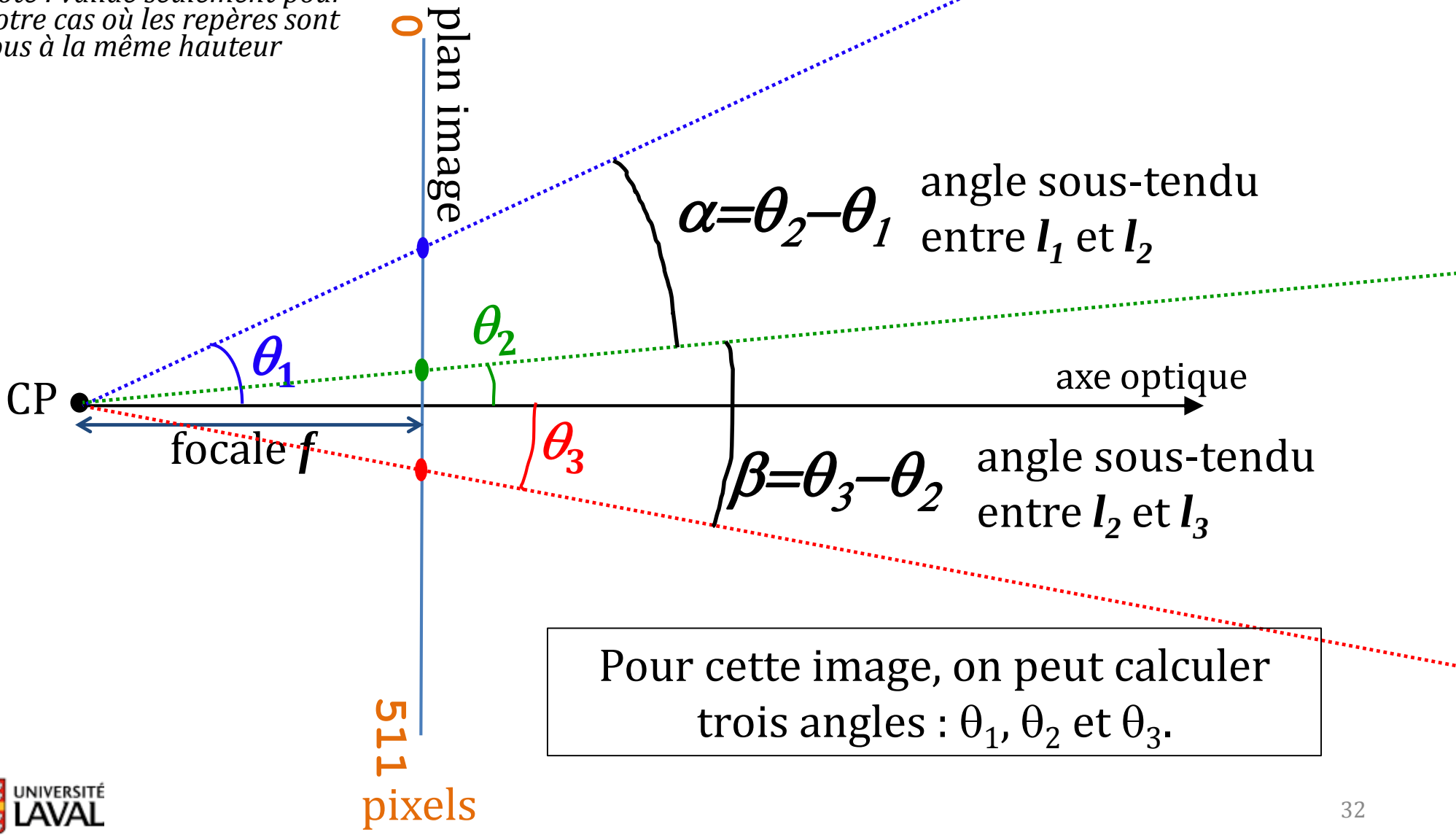
$$\tan\theta_1 = \frac{p_1}{f} \quad \Rightarrow \quad \theta_1 = \arctan\left(\frac{p_1}{f}\right)$$

note: la valeur de f est connue par la phase de calibration effectuée avant

Caméra == rapporteur d'angles

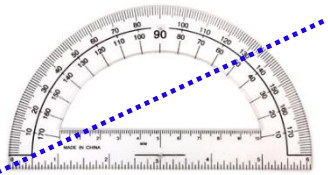


Note : valide seulement pour notre cas où les repères sont tous à la même hauteur



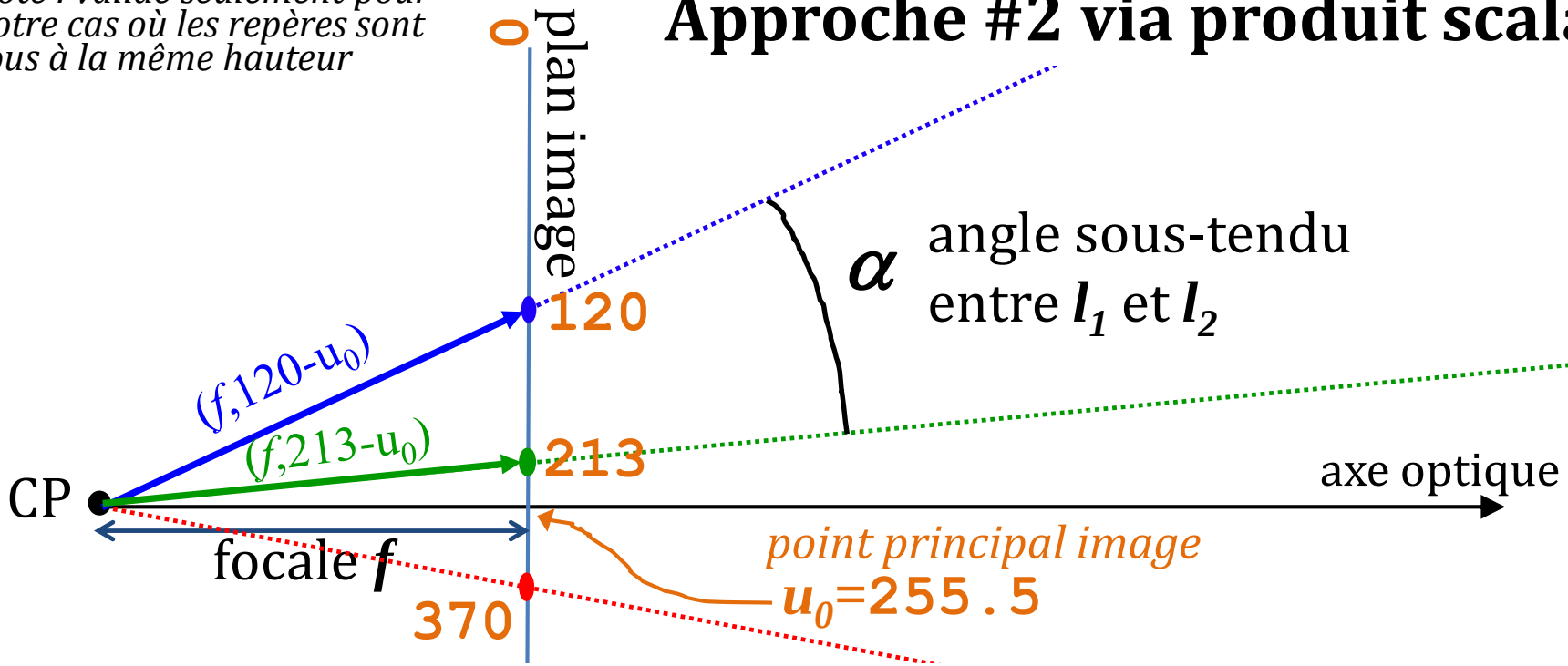
Pour cette image, on peut calculer trois angles : θ_1, θ_2 et θ_3.

Caméra == rapporteur d'angles



Note : valide seulement pour notre cas où les repères sont tous à la même hauteur

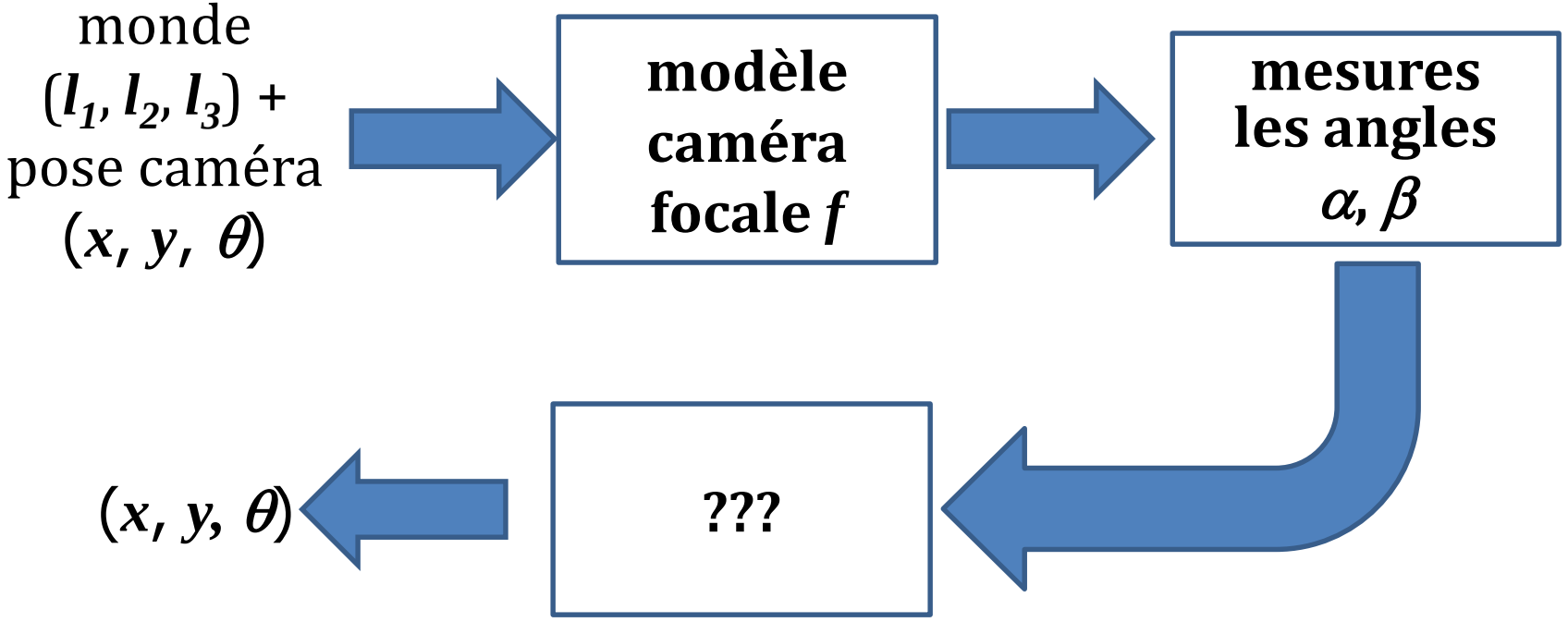
Approche #2 via produit scalaire



$$\alpha = \cos^{-1}\left(\frac{(f,-135.5)\bullet(f,-42.5)}{\|(f,-135.5)\|\|(f,-42.5)\|}\right) = 0.1590 \text{ rad}$$

511 pixels

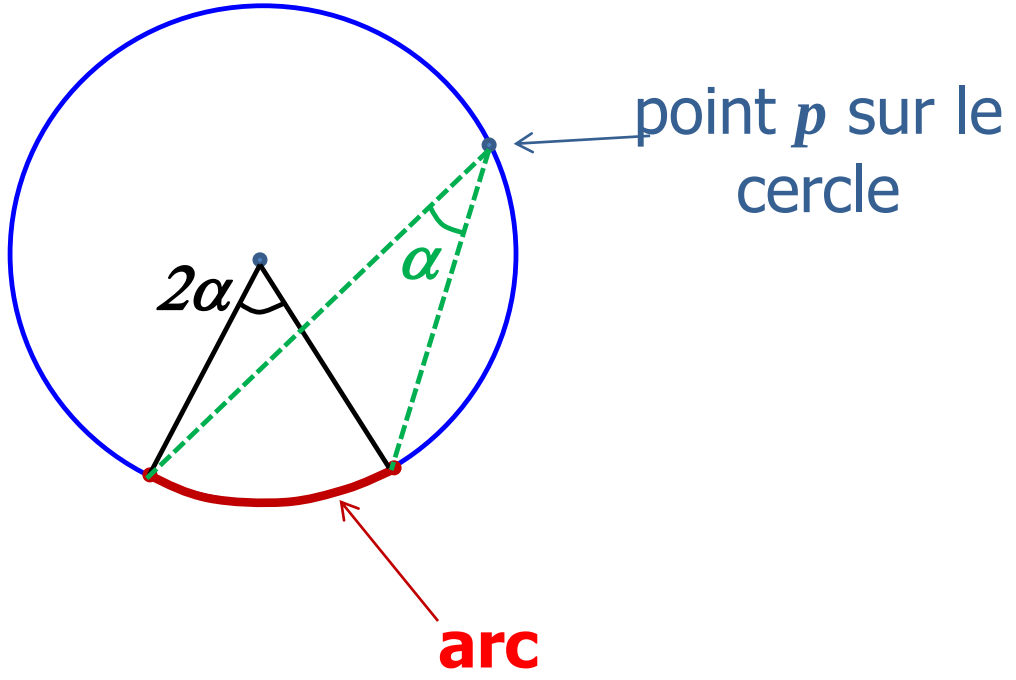
Localisation 2 D



Grande question : comment retrouver la pose de la caméra?

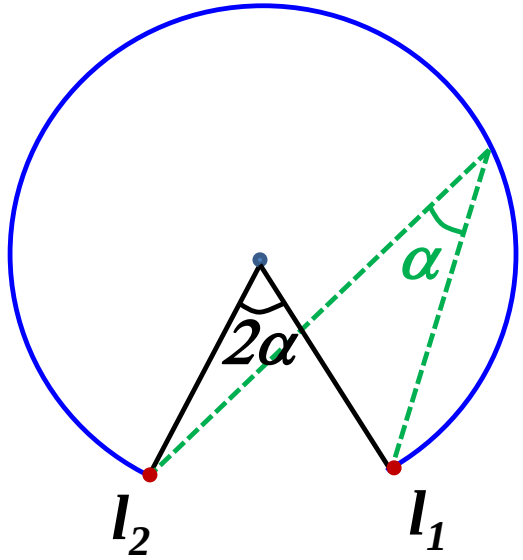
Théorème de l'angle inscrit et de l'angle au centre

- Pour un cercle, l'angle au centre mesure le double d'un angle inscrit interceptant le même **arc**.



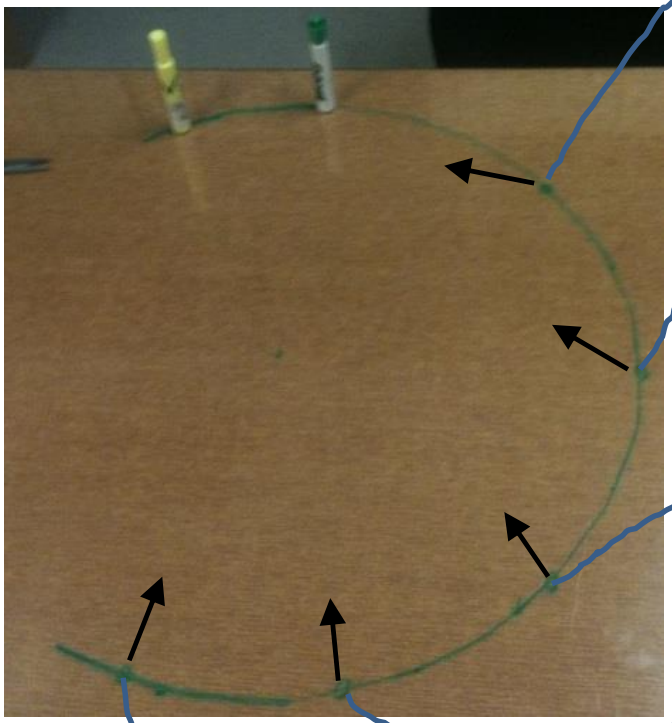
Corollaire : cercle unique pour l_1, l_2 et α

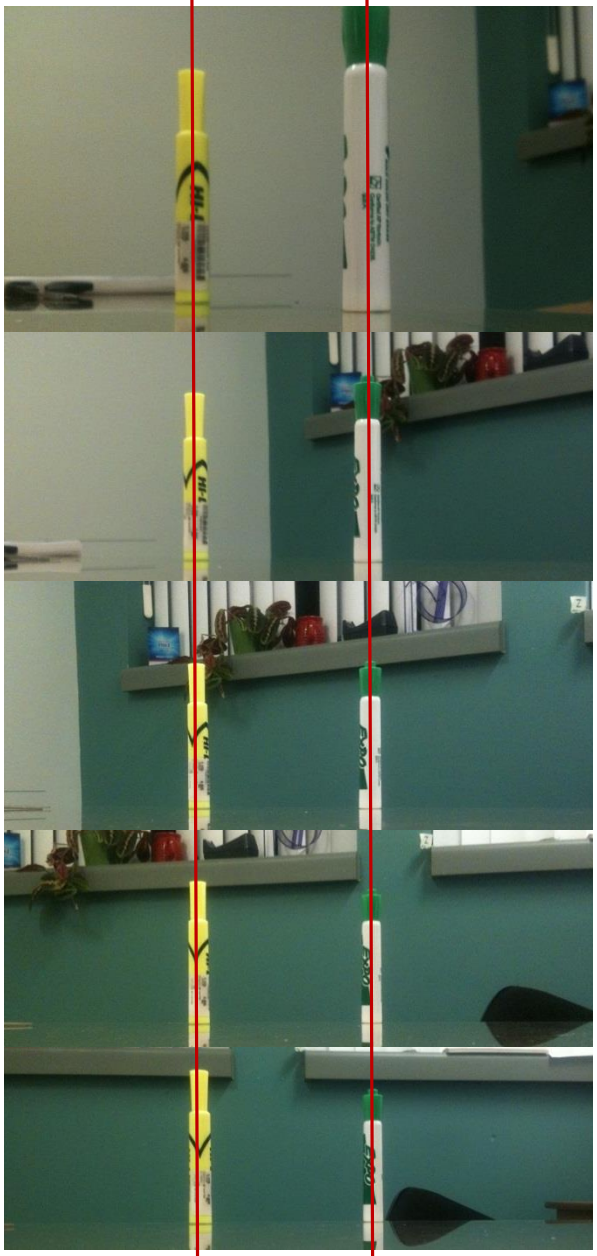
- Je connais la position de l_1, l_2 sur ma carte
- Je connais α, à partir de l'image



- On peut identifier un cercle unique avec ces trois paramètres
- La caméra se trouve quelque part sur l'arc de cercle bleu

Exemple avec ma caméra iPhone sur mon bureau





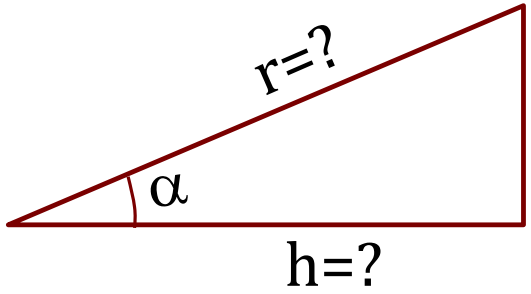
(faire attention ici, car il faut toujours calculer les angles à partir du centre de l'image. Pour des fins de simplifications, je ne regarde ici que la distance entre les marqueur)

toutes les photos ont la même largeur

même angle α

Identifier le cercle pour l_1, l_2 et α

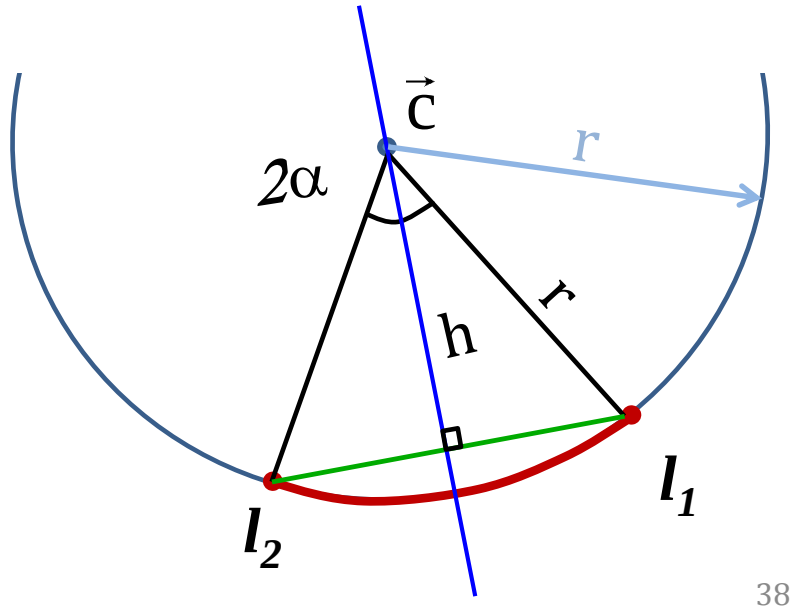
- Un cercle se paramétrise par
  - position du centre $\vec{c} = (c_x, c_y)$ et rayon r
- Corde l_2 l_1
- $\vec{c}$ est situé sur la médiatrice de la corde
- Triangle rectangle



$$a = \frac{|l_1 - l_2|}{2}$$

$$r = \frac{a}{\sin\alpha} = \frac{|l_1 - l_2|}{2\sin\alpha}$$

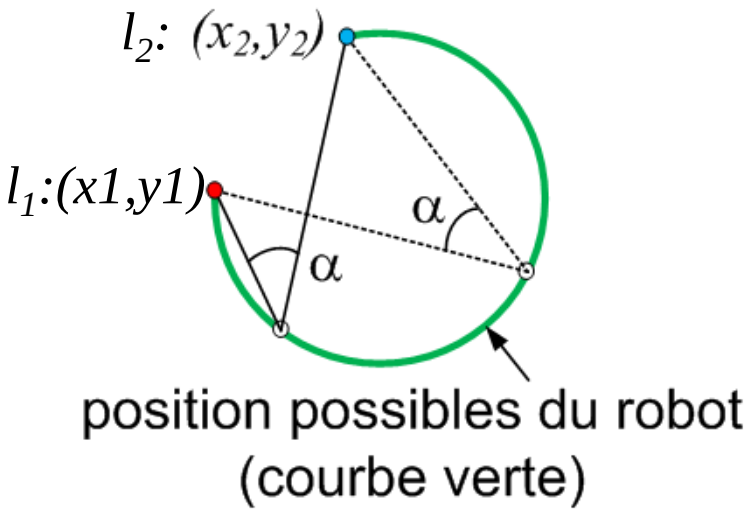
$$h = \frac{a}{\tan\alpha} = \frac{|l_1 - l_2|}{2\tan\alpha}$$



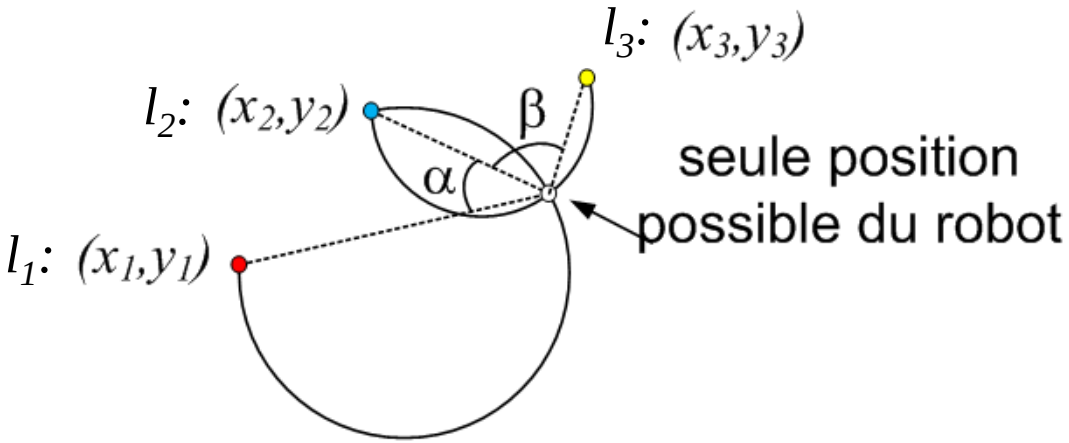
Localisation 2D par triangulation (angles)

- Angles α et β entre repères dans l'image de la caméra
- Connait position des points de repères $\boldsymbol{l_1}$, $\boldsymbol{l_2}$ et $\boldsymbol{l_3}$

Deux repères, un angle α.



Trois repères, deux angles α, β.



manuel p. 243-246

Autres solutions

Trouver la position du robot à partir de l_1, l_2, l_3, α et β?

- Approche géométrique
  - Règle et compas

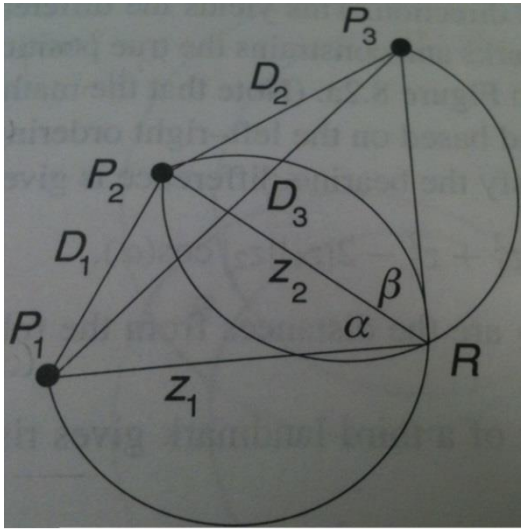
à l'examen!

- Satisfaction de contraintes non-linéaires (p.246 manuel)

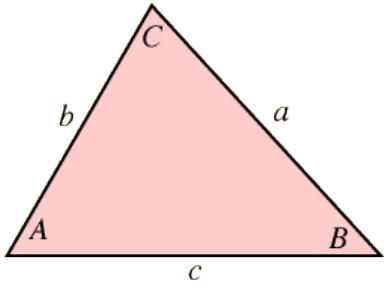
loi des cosinus

$$D_1^2 = z_1^2 + z_2^2 - 2\,|z_1|\,|z_2|\cos(\alpha)$$
$$D_2^2 = z_2^2 + z_3^2 - 2\,|z_2|\,|z_3|\cos(\beta)$$
$$D_3^2 = z_1^2 + z_3^2 - 2\,|z_1|\,|z_3|\cos(\alpha+\beta)$$

- Recherches itératives
- *Newton-Raphson*

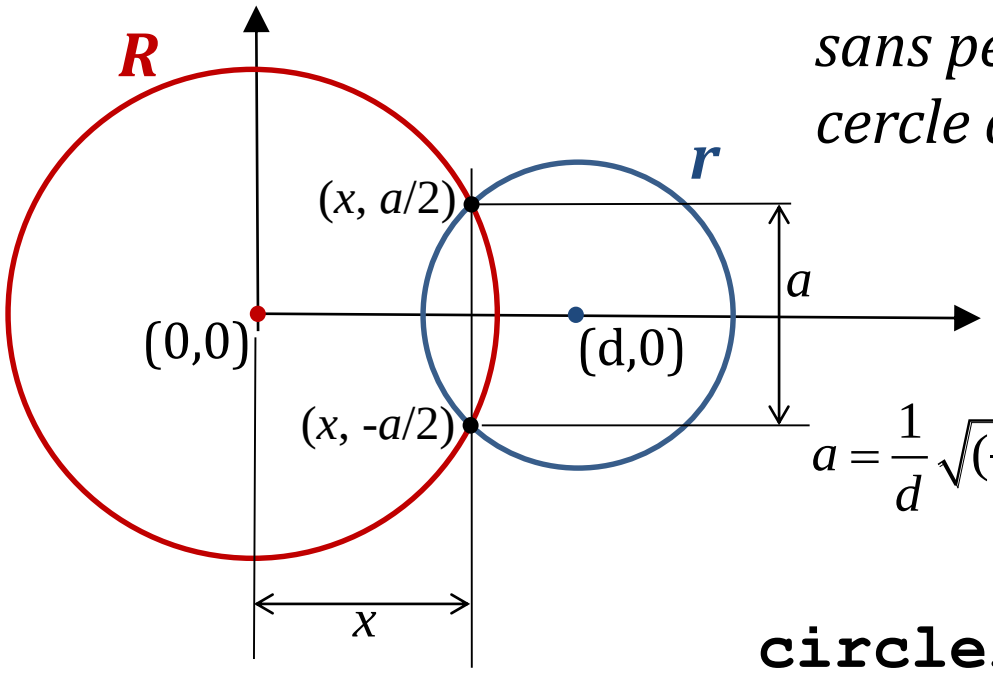


$$c^2 = a^2 + b^2 - 2ab\cos C$$



Approche géométrique

- Trouver le centre et rayon des deux cercles
- Utiliser les formules de l'intersection de deux cercles *http://mathworld.wolfram.com/Circle-CircleIntersection.html*



sans perte de généralité, si vous avez un cercle à (0,0) et l'autre à (d,0)

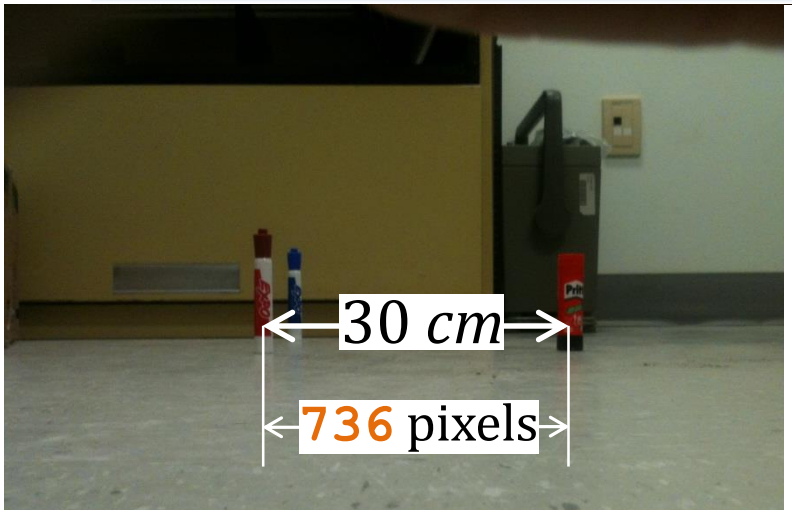
$$x = \frac{d^2 - r^2 + R^2}{2d}$$

$$a = \frac{1}{d}\sqrt{(-d+r-R)(-d-r+R)(-d+r+R)(d+r+R)}$$

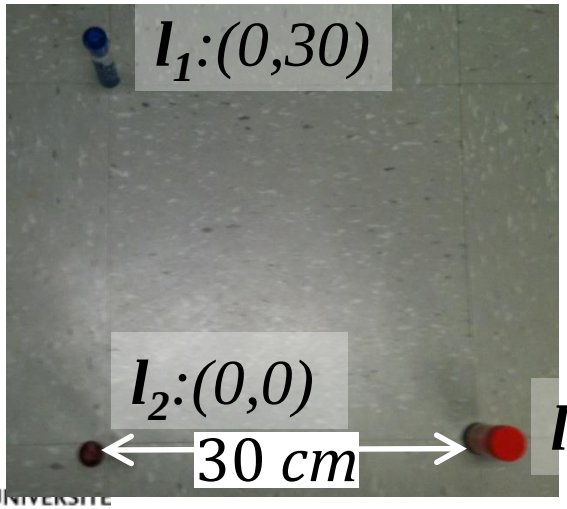
Encore mieux! Google **`circleintersect.m`** par Peter Kovesi

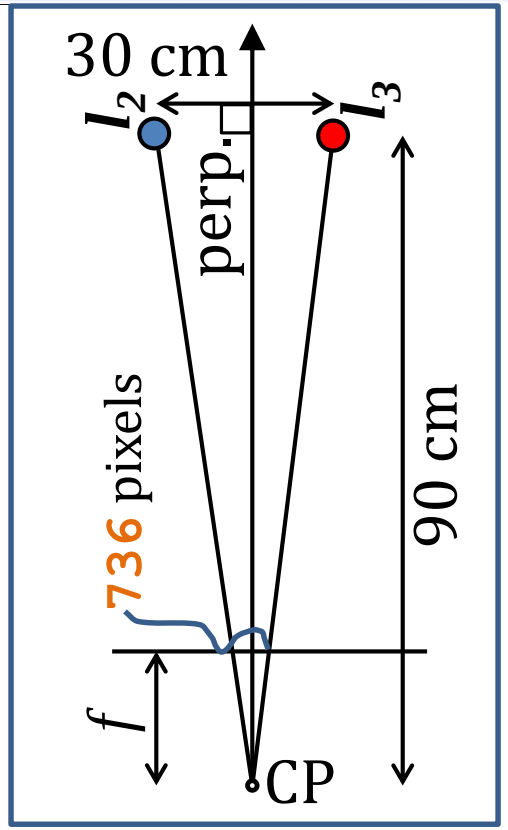
ou **`circcirc`** dans mapping toolbox de matlab

Localisation : étape 1, calibration de *f*



vue de haut



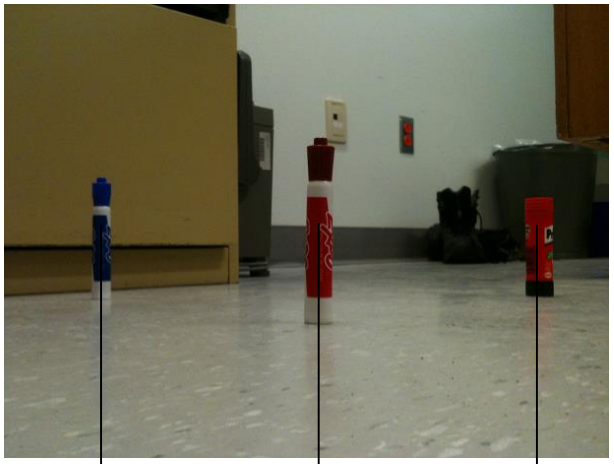


(Notez que cette étape n'est faite qu'une seule fois dans un système robotique, au préalable)

$$\frac{f}{736\,pixels} = \frac{90cm}{30cm}$$

$$f = \frac{90}{30}736\,pixels = 2208\,pixels$$

Localisation : étape 2, calcul des angles



328 1066 1815 ← Position horizontale des objets dans l'image

- Image de 2048 pixels horizontaux
- Centre : u_0=1024,5 (normalement on doit calibrer cette valeur aussi!)
- Distance focale f = 2208
- Trouve centre des objets

(Rappel : parce que le robot se déplace à plat, localisation 2D)

- Calcul les trois angles θ_1, θ_2 et θ_3 :

$$\theta_1 = \arctan(\frac{328-1024.5}{f}) = -0.3056 \text{ rad}$$

$$\theta_2 = \arctan(\frac{1066-1024.5}{f}) = 0.0188 \text{ rad}$$

$$\theta_3 = \arctan(\frac{1815-1024.5}{f}) = 0.3438 \text{ rad}$$

- Calcul les angles α et β :

$\alpha=\theta_2-\theta_1$=0.3244 rad

$\beta=\theta_3-\theta_2$=0.3250 rad ou

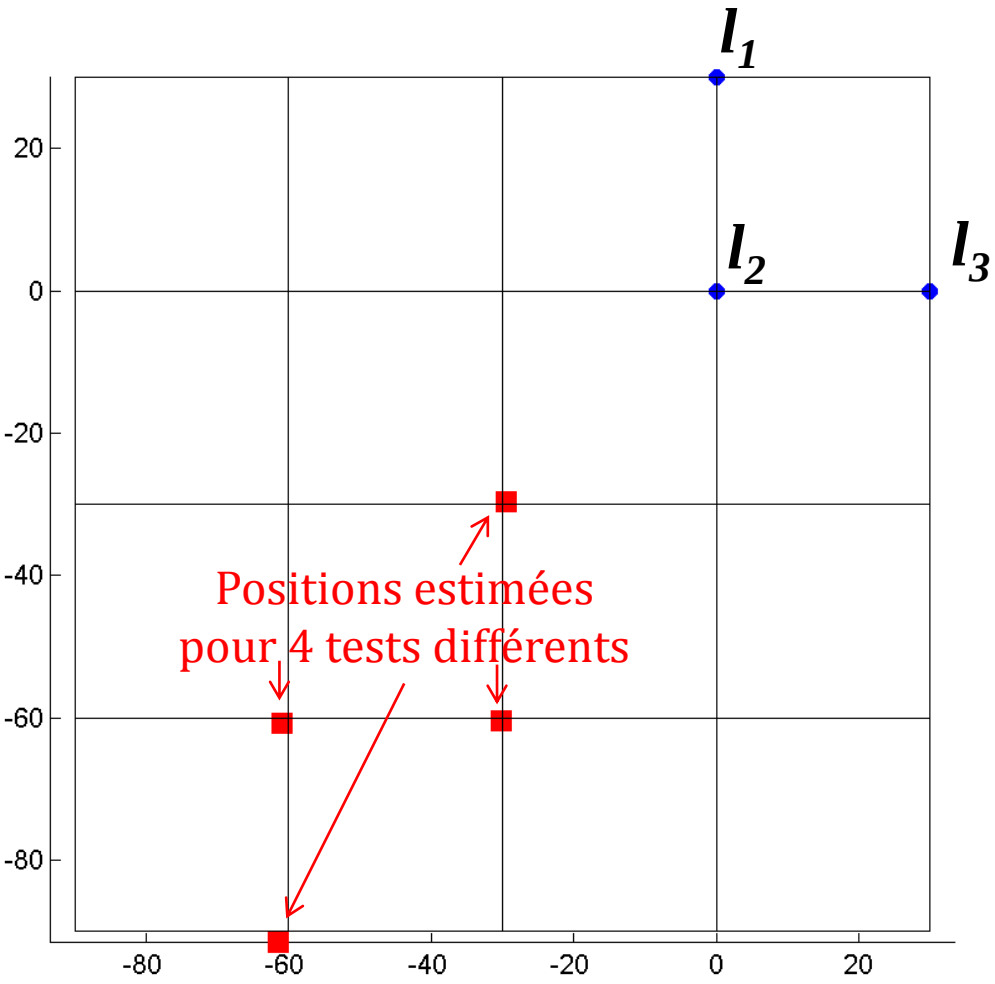
$$\alpha = \cos^{-1}\left(\frac{(f,-696.5)\bullet(f,41.5)}{\|(f,-696.5)\|\|(f,41.5)\|}\right)$$

Similaire pour β

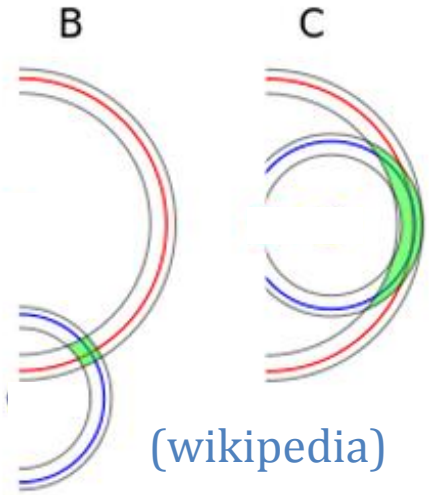
- Résout avec l'une des méthodes mentionnées auparavant

Exemple de localisation : résultats

- Les positions calculées sont très près des positions réelles (coins des tuiles)
- Mais la précision diminue grandement à mesure que l'on s'éloigne des points de repère ou pour certaines positions



Geometric Dilution of Precision



(wikipedia)

Imagerie stéréo : retrouver la 3D

Images stéréo

- 2 caméras placés à b=5 cm distance, axes optiques parallèles

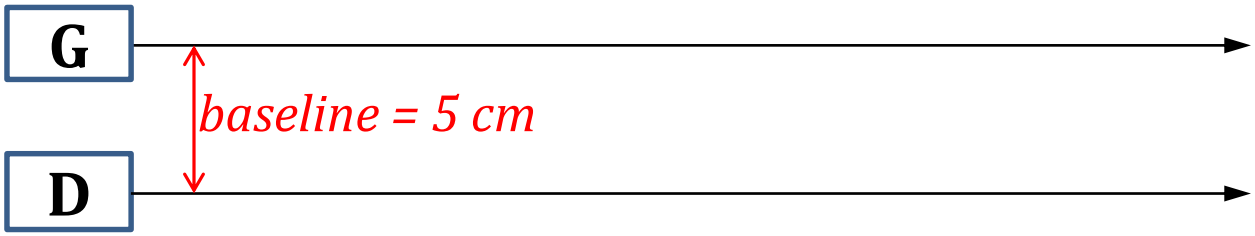


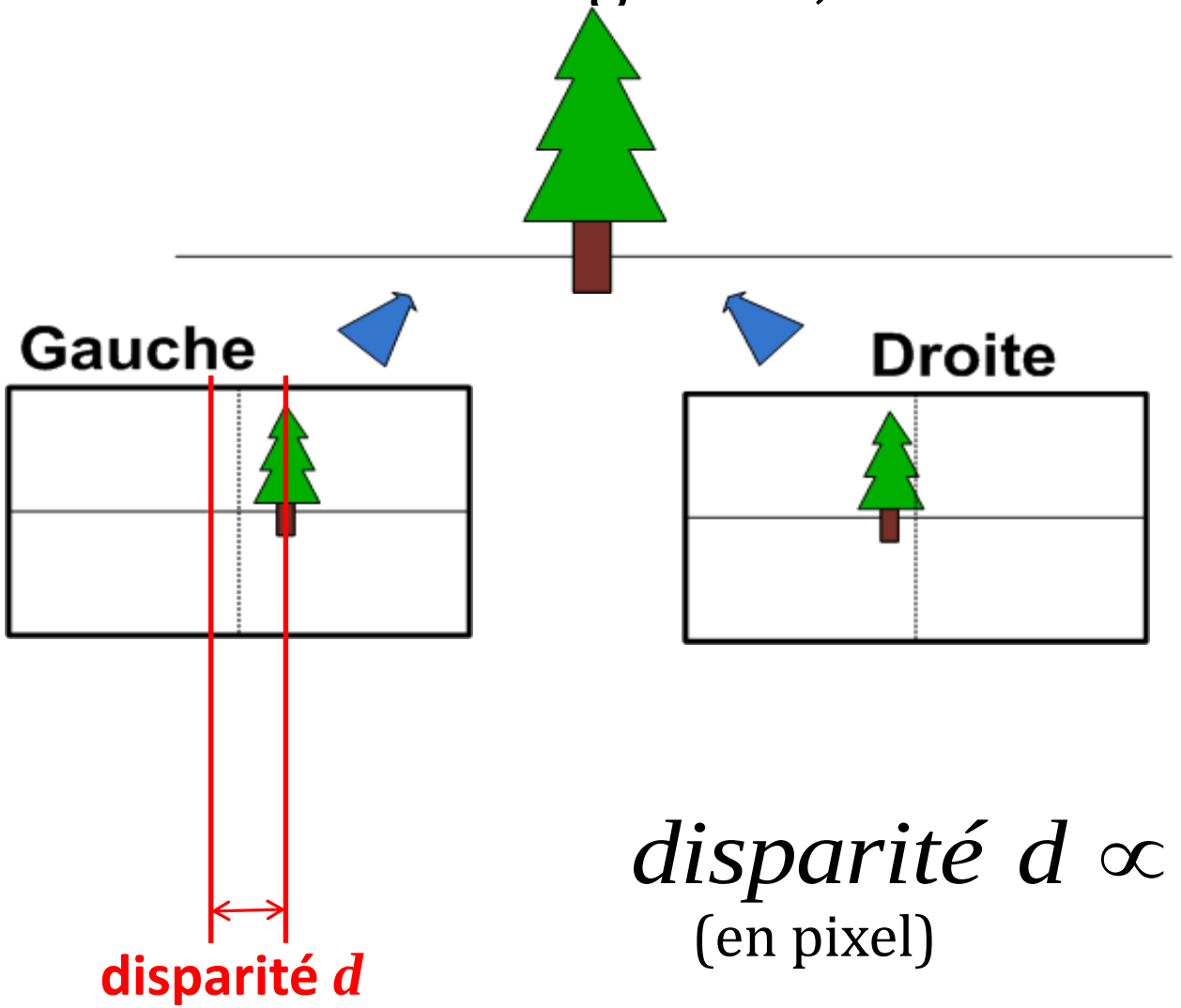
Image de caméra gauche

Image de caméra droite

centre de l'image

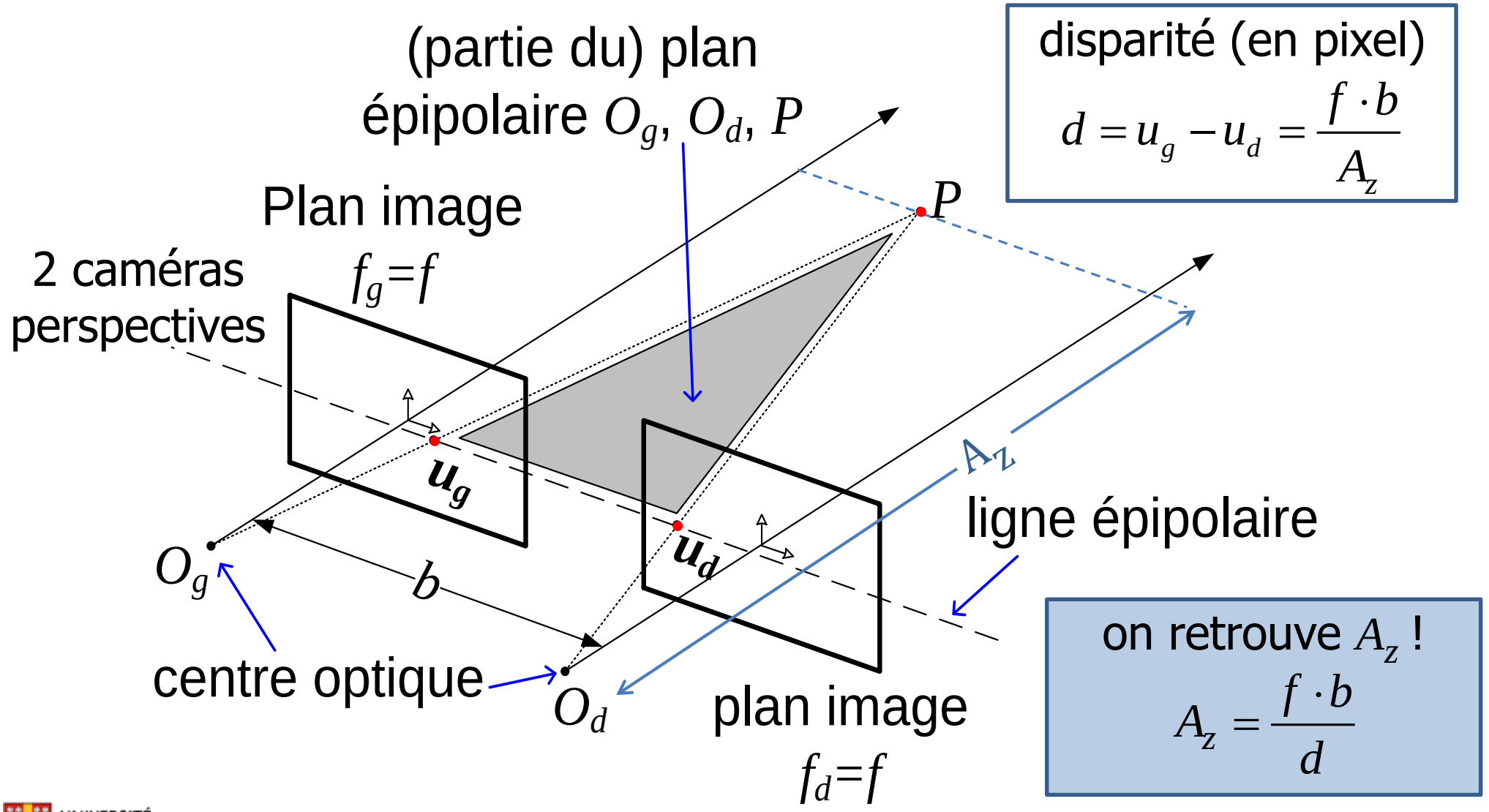
Imagerie stéréo

- À partir de deux images 2D, retrouver 3D



$$disparité\ d \propto {}^{1}\!/_{A_z}$$
(en pixel)

Caméra stéréo : pour retrouver la 3D



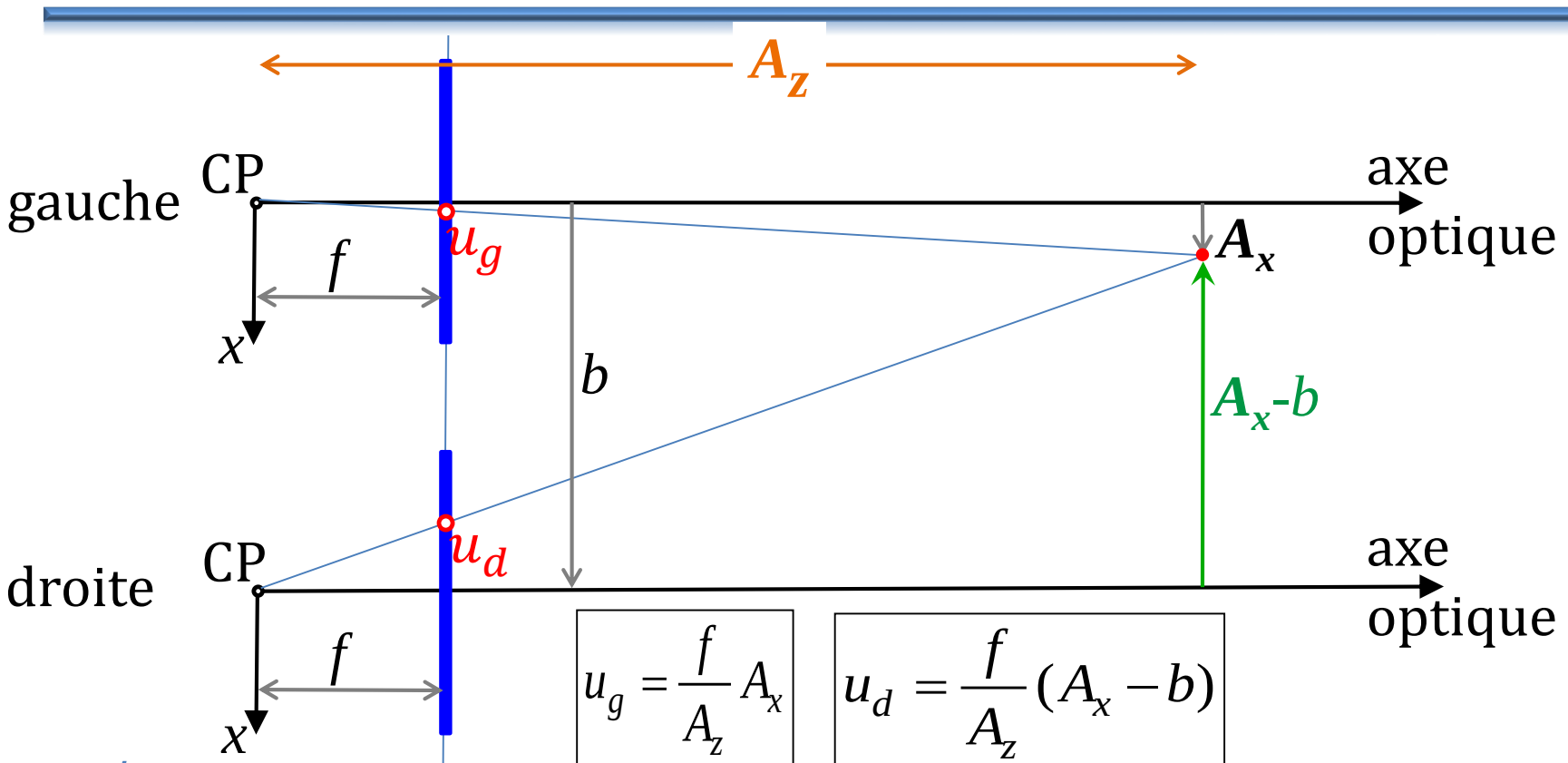
disparité (en pixel)

$$d = u_g - u_d = \frac{f \cdot b}{A_z}$$

on retrouve A_z !

$$A_z = \frac{f \cdot b}{d}$$

Disparité



$$u_g = \frac{f}{A_z} A_x \qquad u_d = \frac{f}{A_z}(A_x - b)$$

$$d = u_g - u_d = \frac{f}{A_z} A_x - \frac{f}{A_z}(A_x - b) - = \frac{f}{A_z} b$$

note: on assume la même focale f pour g et d

On retrouve donc la profondeur du point : $A_z = \frac{f\ b}{d}$

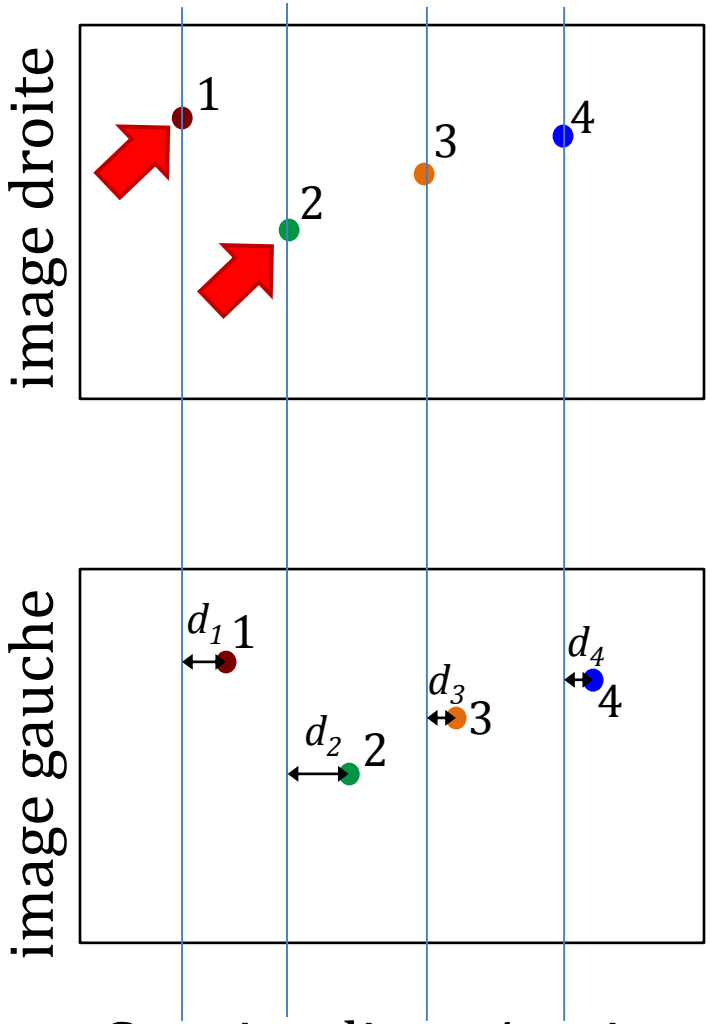
Sensibilité stéréo : *baseline b*

Télémètre Allemand
Entfernungsmesser-1m-R-36
Tir anti-aérien
Baseline $\boldsymbol{b}$= 1 *m*

$$A_z = \frac{f\ b}{d}$$

Disparité à chaque point visible (pixel?)

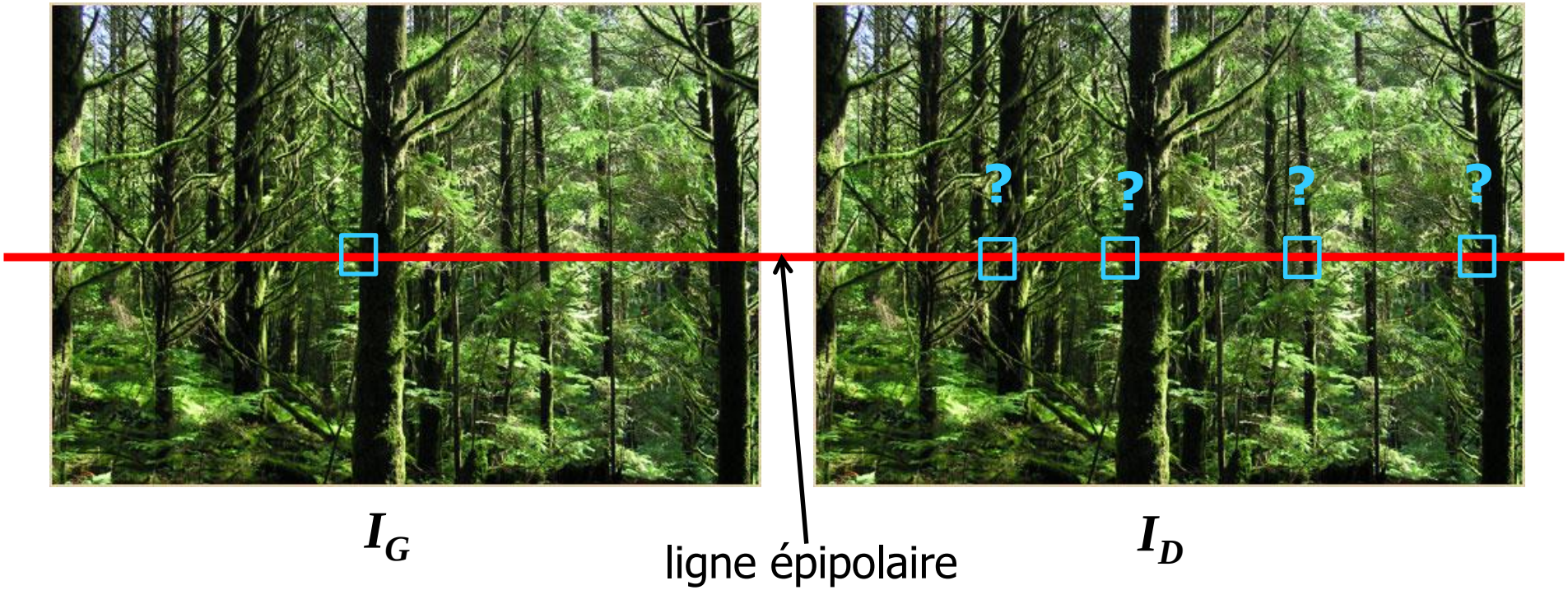
- Pour chaque point visible i, on doit :
  - faire la correspondance entre les deux images *(basé sur l'apparence)* *data association*
  - estimer la disparité d_i
- On pourra ainsi retrouver la profondeur A_{zi} pour chaque point i
- Si on calcule la disparité d_i pour chaque pixel, beaucoup de calcul!



Cas simpliste, 4 points visibles devant la caméra

Problème de correspondance… *data association*

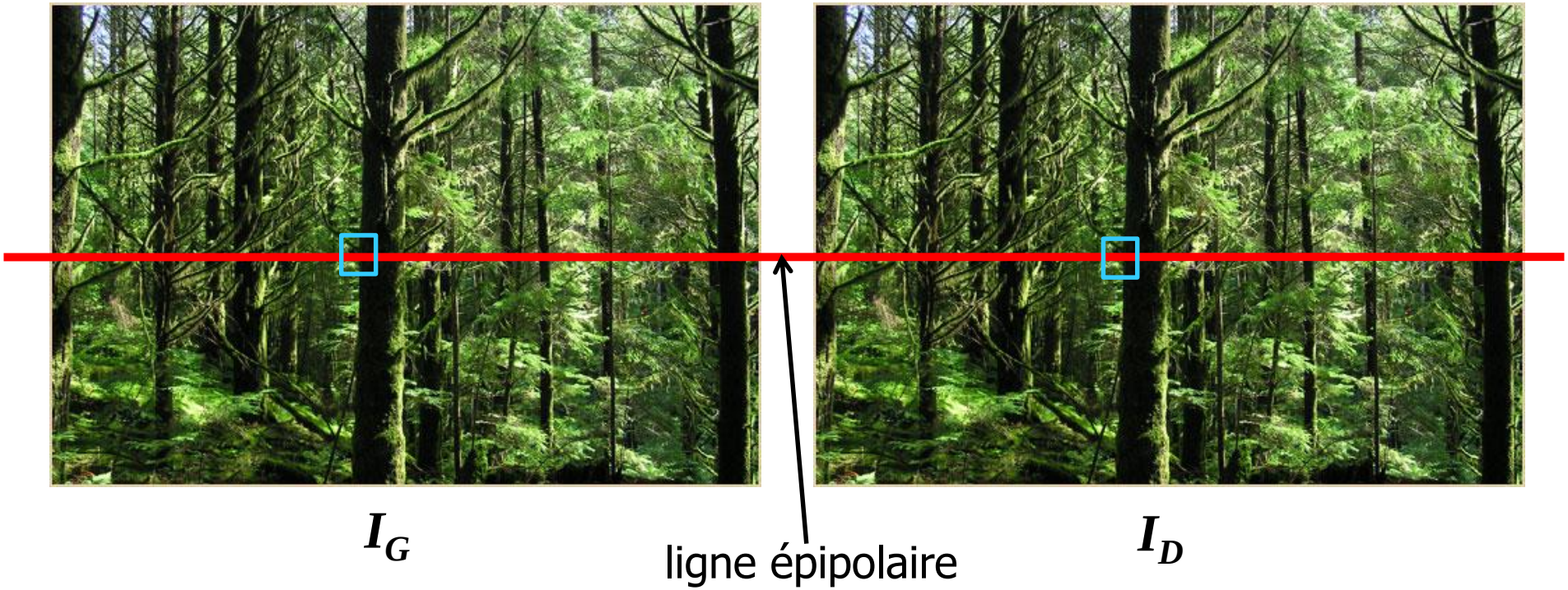
- Quel pixel dans l'image I_D correspond au pixel du même point physique dans image I_G sur la **ligne épipolaire***?



*On assume que les caméras sont décalées strictement horizontalement, d'où la ligne épipolaire horizontale

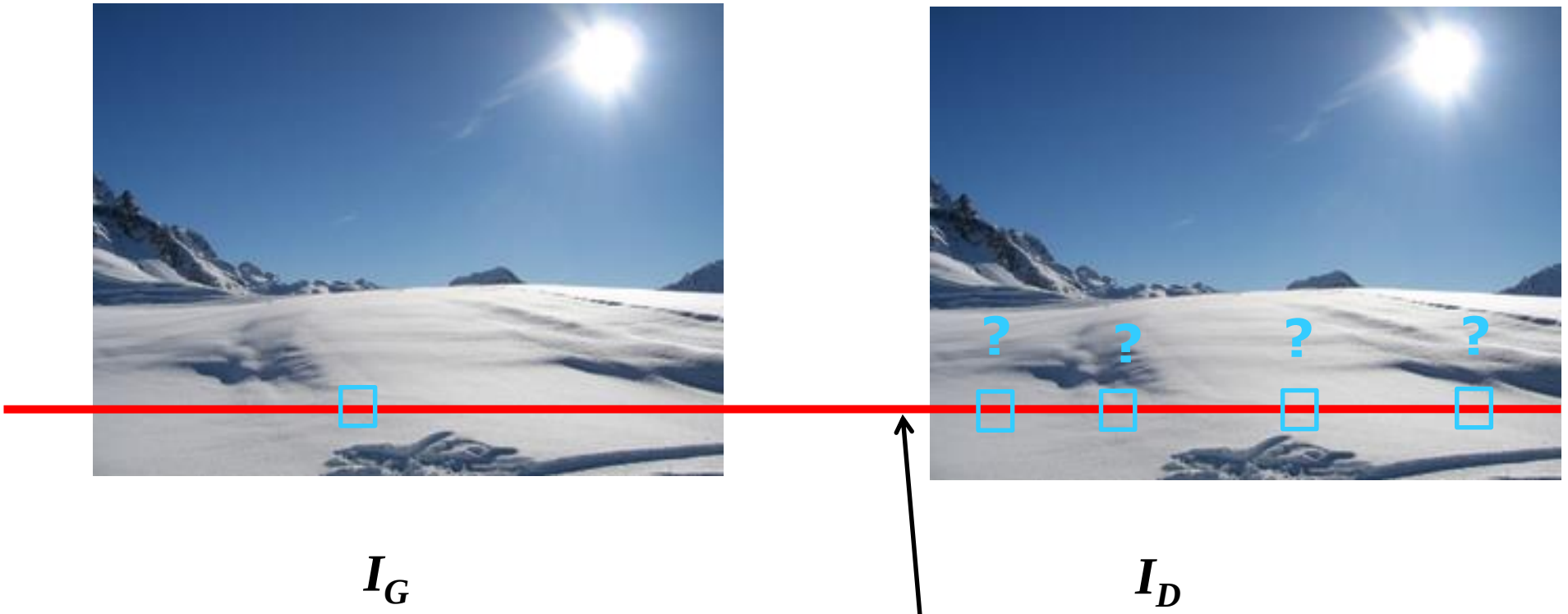
Problème de correspondance… *data association*

- Quel pixel dans l'image $\boldsymbol{I_D}$ correspond au pixel du même point physique dans image $\boldsymbol{I_G}$ sur la **ligne épipolaire**?



Problème de correspondance…

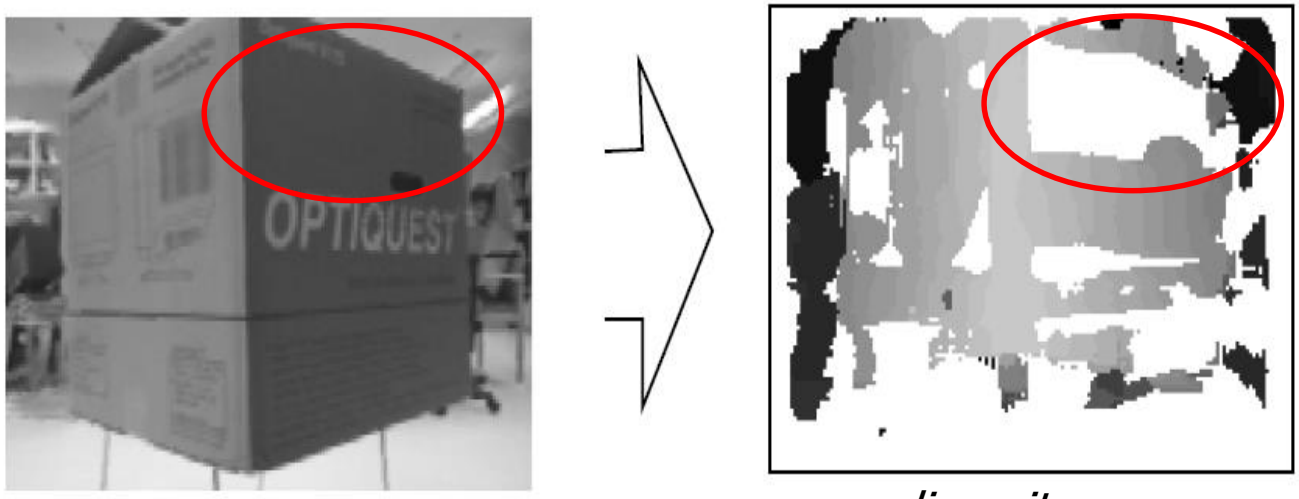
- Quel point dans image I_D correspond au point dans l'image I_G?



Exemples *disparity map*

- *Disparity map* : valeur de A_z pour chaque pixel

manque *features* visuels



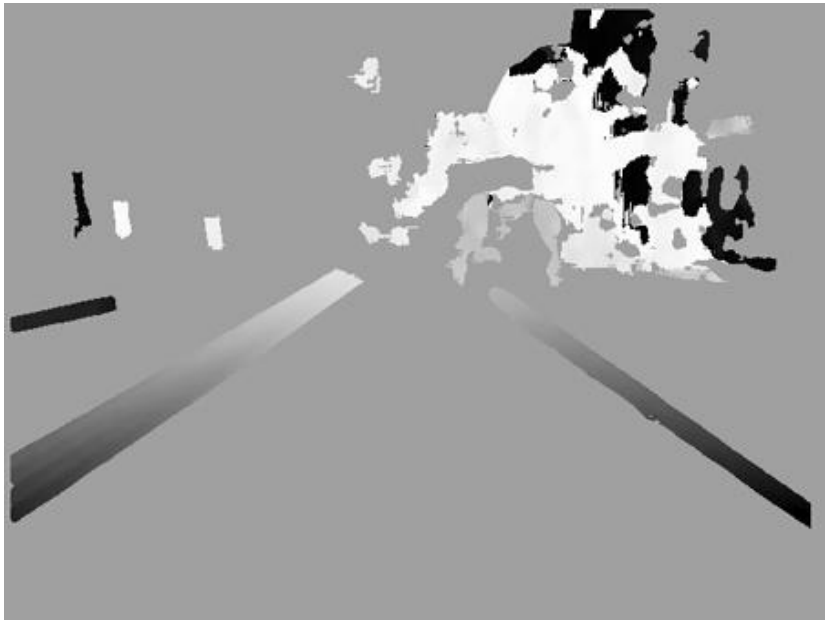
disparity map

Using real-time stereo vision for mobile robot navigation

Don Murray Jim Little

Computer Science Dept.
University of British Columbia
Vancouver, BC, Canada V6T 1Z4

Conduite en ville par stéréo ?



Caméras actives

Kinect 1 : stéréo active

- Caméra « 3D »
- Lumière structurée

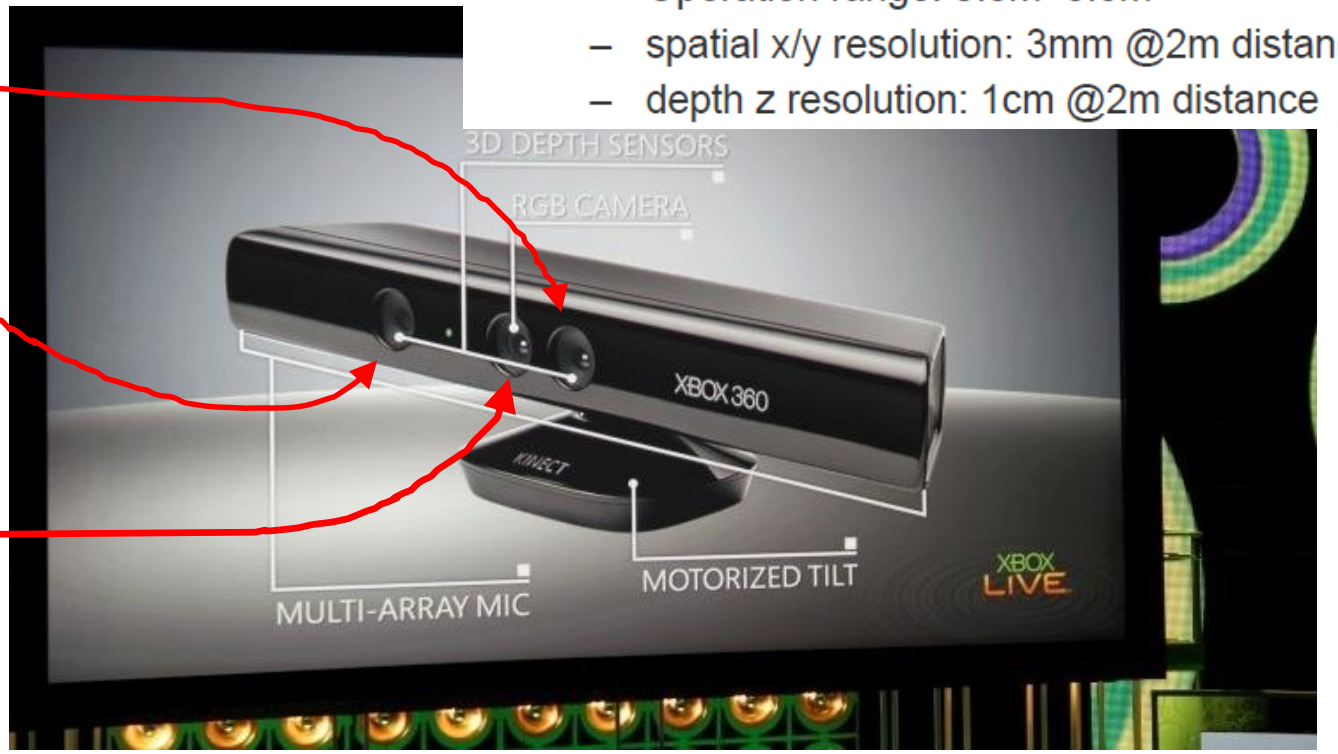
Microsoft Kinect

- Depth resolution: 640x480 px
- RGB resolution: 1600x1200 px
- 60 FPS
- Operation range: 0.8m~3.5m
- spatial x/y resolution: 3mm @2m distance
- depth z resolution: 1cm @2m distance

caméra infrarouge

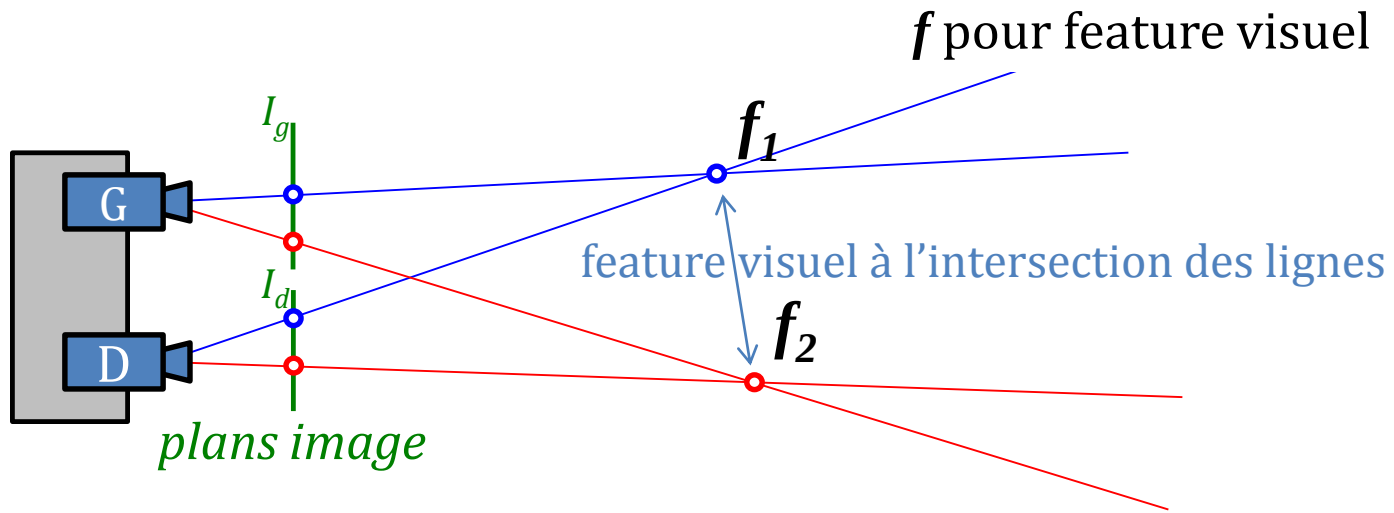
projecteur infrarouge

caméra RGB

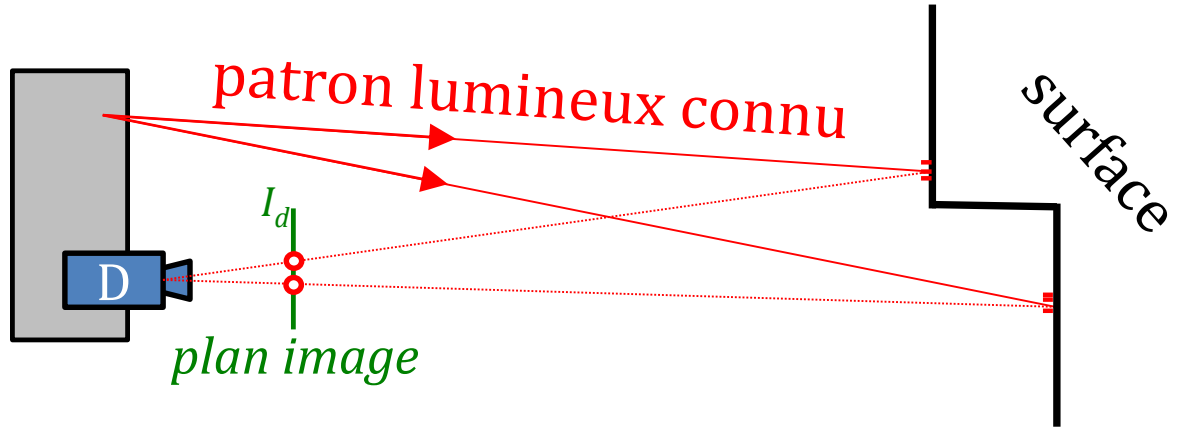


Kinect 1 : stéréo active

Stéréo normale



Stéréo active



Kinect 1 : stéréo active

- Projette patron infrarouge <u>localement distinct</u>
  (pensez code barre)

  Moins de problème de correspondance!
- Observe la « disparité » *d* avec la caméra infrarouge

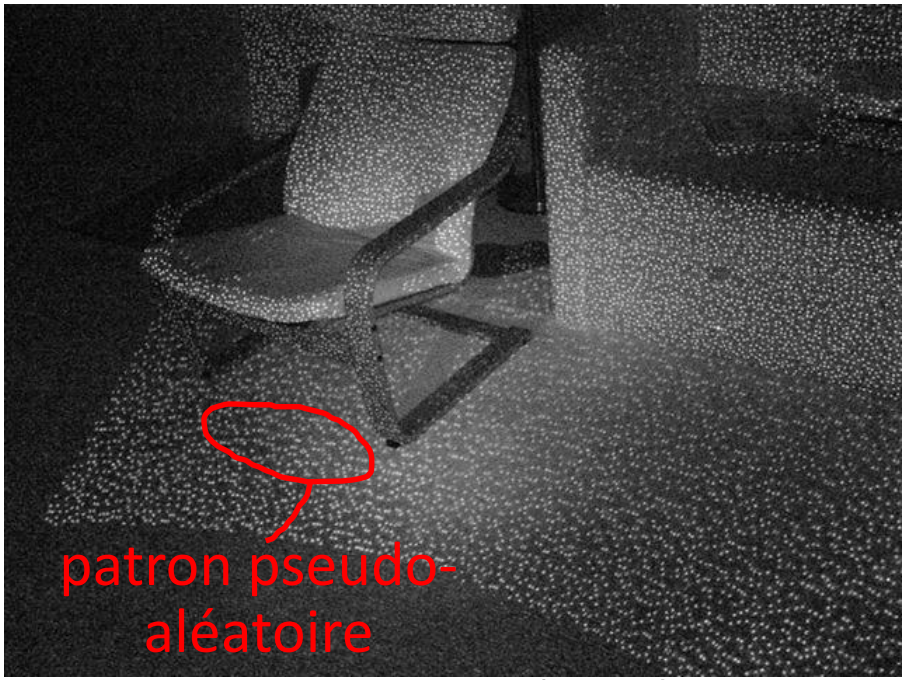
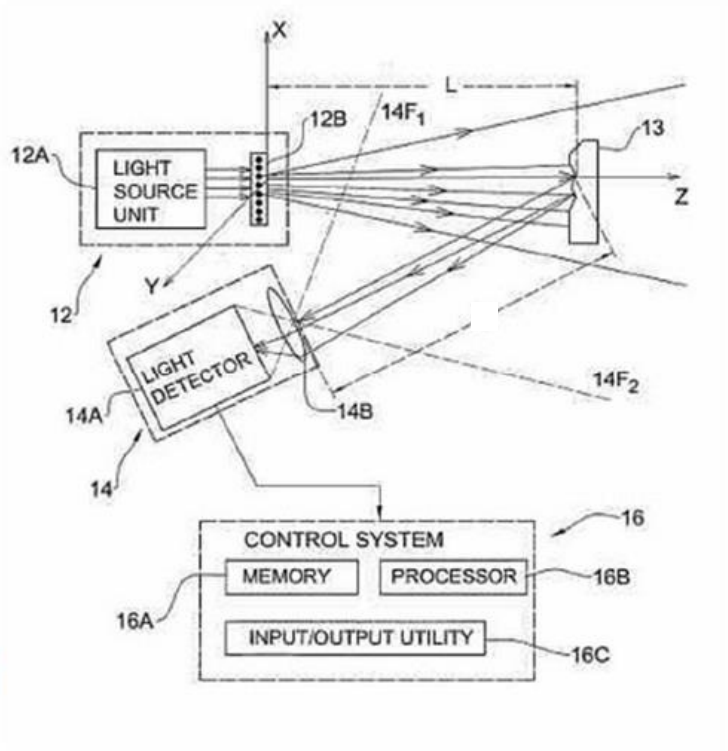


photo prise avec caméra infra-rouge



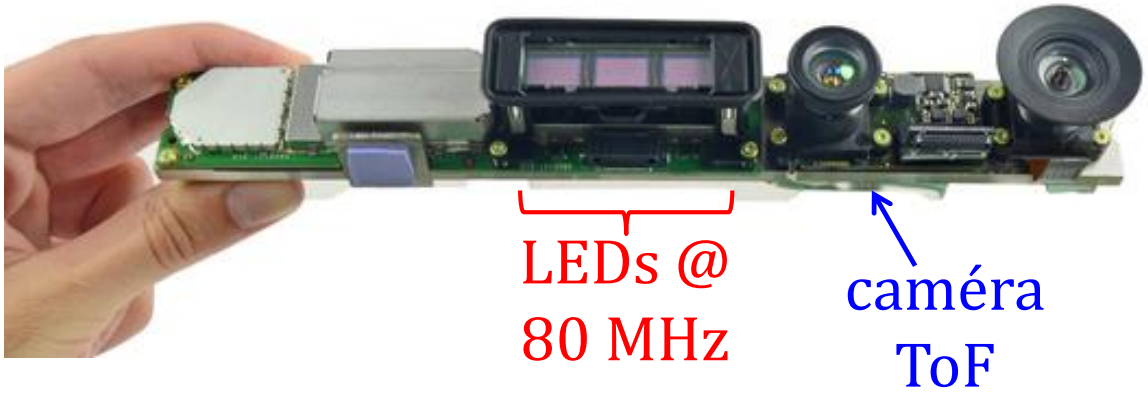
Kinect 1 : depth image

- Retourne une image de profondeur (*depth image*)
- Pour chaque pixel, on aura la distance en *Z* (ou rien)
- 640x480 pixels, 30 *Hz*
- Précision dépend de la distance

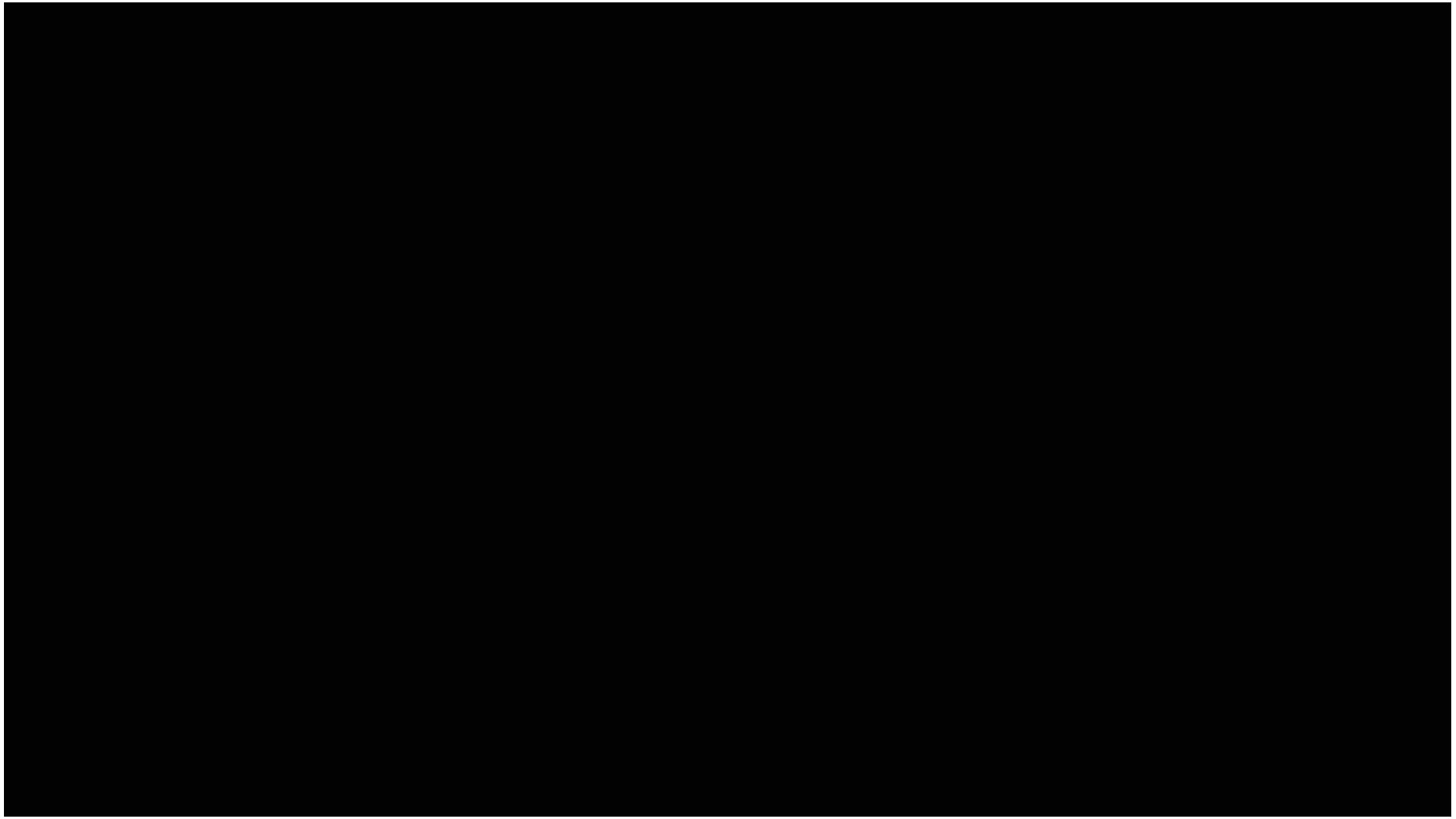
depth image *image RGB*

Kinect v2

- Technologie différente : basée sur temps de vol *(time of flight : ToF)*
- Mesure la phase entre émission lumineuse des LEDs et chaque pixel de la caméra ToF



Kinect v2



Autres types de caméras

Caméra omnidirectionnelles

- Caméra standard + miroir convexe = vue 360°

Caméra omnidirectionnelles

- *Ladybug* de *Pointgrey* combine 6 caméras ensemble: vue 360°

même voiture

Event camera : DVS128 de iniLabs

- Envoie des messages de changement d'intensité **ON/OFF**
- Temps de réponse autour de 15 microseconde