(1)

Séance 1
Mise à niveau contours actifs

1) Problème

Dans une image monochrome $I : [0,1]^2 \to [0,1]$
(en pratique de $[0,W]\times[0,H] \to [0,255]$)
les contours d'un objet ont un fort gradient :
$\boxed{\|\nabla I\| \text{ élevé}}$

2) Seuillage par hystérésis

(a) - Débruiter l'image, par exemple en convoluant avec une gaussienne (en pratique, appliquer le noyau $\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$)

(b) - Calculer les dérivées par différence finie (appliquer $(-\frac{1}{2} \; 0 \; \frac{1}{2})$ pour $I_x = \frac{\partial I}{\partial x}, \dots$)

NB : On réalise (a) et (b) en même tps avec par exemple Sobel $\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$. Mieux on utilise des filtres récursifs (Deriche) implémentant rapidement et exactement des filtres "optimaux" c'est-à-dire adaptés à un profil idéal de contour (Canny)

(c) Calculer $\|\nabla I\|$

(d) NMS : ne conserver dans la suite que les $\underbrace{\text{points max.}}_{\text{de } \|\nabla I\|}$ dans la direction du gradient.

(e) Choisir deux seuils $S_1 > S_2$ et :
- extraire les points tq $\|\nabla I\| > S_1$ (graine)
- extraire les points de $\|\nabla I\| > S_2$ reliés aux grains par d'autres points de $\|\nabla I\| > S_2$

3) Snakes

Inconvénient de la méthode précédente : pas d'information globale sur le bord d'un objet (des contours à l'intérieur, pas de contour complet au bord, etc...)

→ on modélise maintenant un contour comme une courbe $C : [0,1] \to [0,1]^2$ dépendant de paramètres $(\alpha_i)_{1 \leq i \leq N}$ (en pratique, B-spline) et on cherche :

$$\max_{(\alpha_i)} E(c) \text{ avec } \boxed{E(c) = \int \|\nabla I(C(p))\| \, dp + \text{lissage}}$$

(en général, lissage $= \int \|c'\|$ ou $\int \|c''\|$)

→ on obtient ce maximum par descente de gradient

→ Problèmes : 1) Non intrinsèque (dépend de la forme de C mais aussi de la paramétrisation p)

2) Minima locaux → commencer la desc. de gradient près de la solution.

4) Contours actifs

$C(p)$ est maintenant fermée ($C(0) = C(1)$) et ne dépend plus de paramètres (α_i). On minimise $E(c)$ avec

$$\boxed{E(c) = \int g(\|\nabla I(C(p))\|) \, d\sigma(p)}$$

avec g fn° positive décroissante (par ex $\frac{1}{1+c_1 x^2}$)

σ = absciss curviligne de C

La descente de gradient devient :

$$\begin{cases} C(p,0) = C^0(p) \\ \frac{\partial C}{\partial t}(p,t) = -\nabla_C E(C)(p,t) \end{cases}$$

4.1) Gradient de forme

Soit $\vec{w}(p)$ un champ de vecteurs défini sur $C(p)$

alors $D_{\vec{w}} = \lim_{\lambda \to 0} \dfrac{E(C(p)+\lambda\vec{w}(p)) - E(C(p))}{\lambda}$

(Dérivée de Gâteaux)

Le gradient de E en C est le champ $\nabla_C E$ tel que :

$$\forall \vec{w}, \quad D_{\vec{w}} = \langle \nabla_C E \mid \vec{w} \rangle$$

avec $\langle \vec{u} \mid \vec{v} \rangle = \int_C \vec{u} \cdot \vec{v} \, dp$

Exemples

- Si $F(c) = \int_{\Omega_i} f$ avec Ω_i intérieur de C alors

$$\nabla_C F = f \vec{N} \quad (\vec{N} \text{ normale } \underbrace{\text{à}}_{\text{extérieure}} C)$$

- Si $F(C) = \int_C d\sigma$ (= longueur de C !) alors

$$\nabla_C F = \kappa \vec{N} \quad (\kappa \text{ courbure signée})$$

4.2) Cas des contours actifs

Pour $E(c) = \int g(\|\nabla I\|)\, d\sigma$ on trouve

$$\boxed{\nabla_C E = + g(\|\nabla I\|)\, \kappa \vec{N} + (\vec{\nabla} g \cdot \vec{N})\, \vec{N}}$$

En pratique, on accelère la convergence en rajoutant ~~on~~ une force dépendant d'une constante C_2 et en partant d'un C^0 entourant l'objet :

$$\begin{cases} C(t=0) = C^0 \\ \dfrac{\partial C}{\partial t} = (g(\kappa + C_2) - \vec{\nabla} g \cdot \vec{N})\, \vec{N} \end{cases}$$

5) Régions actives

Lorsqu'on dispose d'un modèle de l'intérieur ou de l'extérieur du contour (par ex un à-priori sur les couleurs ou la texture de l'objet à segmenter ou du fond) : $E = \int_C g\, d\sigma + \int_{\Omega_i} g_i\, dx + \int_{\Omega_e} g_e\, dx$

avec Ω_i (resp Ω_e) intérieur (resp. extérieur) de C et g_i g_e fonctions modèles.

Le gradient devient :

$$\boxed{\nabla_C E = g\, \kappa \vec{N} + (\vec{\nabla} g \cdot \vec{N})\, \vec{N} + (g_i - g_e)\, \vec{N}}$$

NB : alors que le minimum global des contours actifs est la courbe vide, celui des régions actives ne l'est plus → importance des termes de région !

6) Simulation

× Le pb se ramène à simuler $\left\{ C(t=0) = C^0 ; \frac{\partial C}{\partial t} = \vec{V} \right\}$ (1)

× Discrétiser $C(p,t)$ par un ensemble de sommets et faire $C[p, t+\Delta t] \Leftarrow C[p,t] + \Delta t\, \vec{V}[p,t]$ n'est ni précis, ni stable.

× Level Sets :

- On représente C par une fonction implicite $\varphi(x,y)$ nulle sur C, négative à l'intérieur et positive à l'ext.

$$\boxed{C(\cdot, t) = \{ (x,y) \mid \varphi(x,y,t) = 0 \}}$$

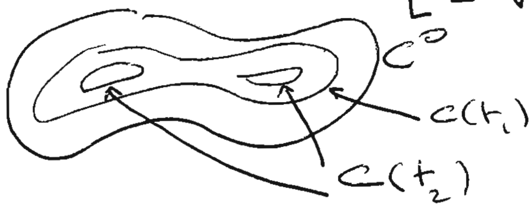
On peut montrer que (1) est équivalent à :

$$\boxed{\begin{cases} \varphi(x,y,0) = \varphi^0(x,y) \\ \frac{\partial \varphi}{\partial t}(x,y,t) = -\vec{V} \cdot \nabla \varphi \end{cases}} \quad (2)$$

(Si φ^0 est une rép. implicite de C^0)

Il suffit donc de discrétiser φ sur une grille régulière et d'implémenter un schéma numérique pour (2) adapté aux $\neq$ts choix de $\vec{V}$.

La méthode est :
- précise et stable
- gère les changements de topologie (i.e. split ou merge du contour)
- valable dans $\mathbb{R}^d$ (en général $\mathbb{R}^3$)



φ^0 $\quad\quad$ $\varphi(t_2)$

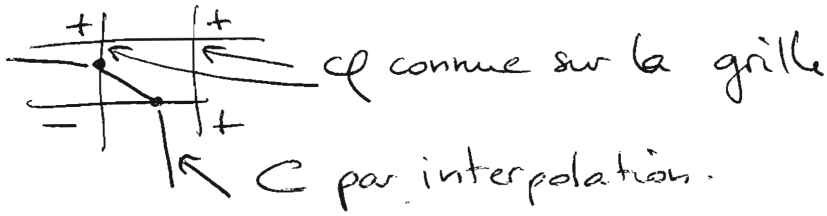
<u>NB</u> : il faut savoir passer de C à φ et réciproq^t

1) $C \to \varphi$ (normalement, seul^t pour C^0)

On choisit φ^0 = <u>distance signée</u> à C^0

2) $\varphi \to C$ (pas à chaque pas de temps, mais quand on veut récupérer C pour l'afficher ou après convergence)

Algorithme des <u>marching cubes</u> qui fournit un maillage du niveau $\emptyset$ de φ avec une précision supérieure à la grille :



2) Graph Cuts

2.1) Coupes minimales

- $G = (V, E)$ graphe orienté pondéré
- s, t deux sommets particuliers de V (source et puits)
- $w(p,q)$ poids de l'arête $(p,q) \in E$ ($w \geq 0$)

– une s,t-coupe C est une partition de V en deux sous ensembles S et T avec $s \in S$, $t \in T$

– coût de la coupe : $\boxed{c(C) = \sum_{\substack{p \in S \\ q \in T \\ (p,q) \in E}} w(p,q)}$

NB : on compte les arêtes de S vers T

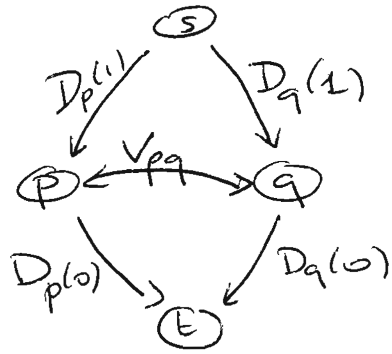
→ Il existe des algorithmes polynomiaux pour calculer la coupe de coût minimal. Ils se ramènent à calculer le flot maximal de s vers t en identifiant $w(p,q)$ à des capacités (Ford-Fulkerson). Leur complexité au pire est en $O(n^3)$ mais linéaire en pratique, en particulier sur les graphes utilisés en computer vision (algorithme des "GraphCut" [Boykov et al.])

2.2) Segmentation binaire

Le problème des régions actives peut se modéliser comme l'étiquetage des pixels p à $f_p = 0$ ou 1 (0 = intérieur, 1 = extérieur) qui minimise

$$\boxed{\begin{array}{l} E(f) = \sum_p D_p(f_p) + \sum_{p,q \text{ voisins}} V_{pq} \mathbb{1}\{f_p \neq f_q\} \\ \text{avec } D_p(0) = g_i(p),\ D_p(1) = g_e(p), \quad V_{pq} = g\left(\frac{p+q}{2}\right) \end{array}}$$

Or, minimiser $E(f)$ revient à trouver la min-cut dans ce graphe :



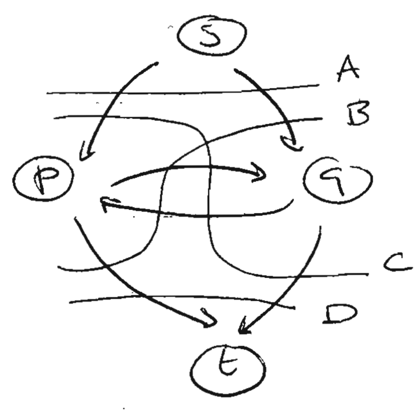
puis à faire : $f_p = 0$ si $p \in S$, $f_p = 1$ si $p \in T$

→ minimum global et rapide pour les régions actives ! (NB : le terme $\int g\,d\sigma$ n'est que grossièrement approché. On peut l'approcher mieux avec des arcs supplémentaires)

2.3) Autres cas

a) Il est possible de construire un graphe pour un cas plus général $\boxed{E(f) = \sum D_p(f_p) + \sum_{pq} V_{pq}(f_p, f_q)}$ à condition que V_{pq} soit submodulaire :

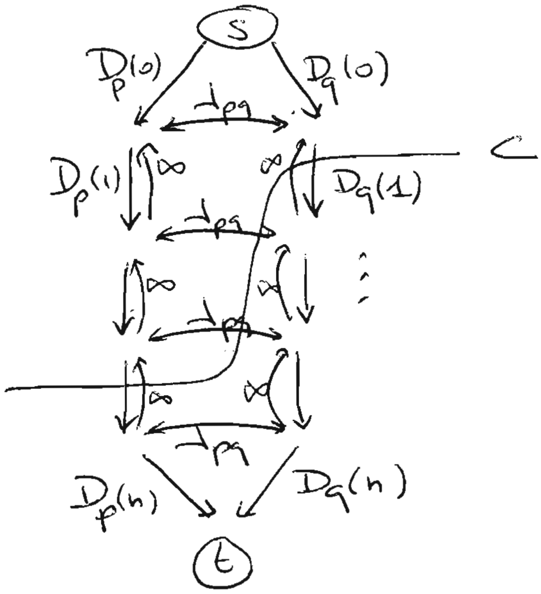
$$\boxed{V_{pq}(0,0) + V_{pq}(1,1) \leq V_{pq}(1,0) + V_{pq}(0,1)} \quad (1)$$



Il faut trouver les bons poids pour que les coupes A, B, C et D coûtent les valeurs d'énergie adéquates → ça n'est possible avec des poids ≥ 0 que si on a (1)

b) Cas où les étiquettes sont multiples ($f_p \in \{0,1,\dots,n\}$)
et où $V_{pq}(f_p, f_q) = \lambda_{pq} |f_p - f_q|$

Il est encore possible, en rajoutant une dimension, de construire un graphe donnant l'optimum global



- f_p est attribué à la "hauteur" de la coupe dans la "colonne" de p
- Pour éviter qu'une colonne soit coupée plusieurs fois, on ajoute des arcs de poids ∞

NB : Le terme de régularisation en $|f_p - f_q|$ est essentiel. En particulier, il n'y a pas de solution pour $\mathbb{1}_{\{f_p \neq f_q\}}$

c) Cas général : $f_p \in \mathcal{L} = \{\alpha, \beta, \dots\}$ fini

Si $V_{pq}(\cdot,\cdot)$ est une métrique alors il est possible d'utiliser la méthode de l'α-expansion. Pour $\neq$ tes valeurs de $\alpha \in \mathcal{L}$, on optimise progressivement f_p en un nouveau f'_p tel que $\boxed{f'_p = f_p \text{ ou } f'_p = \alpha}$. Le choix du f'_p optimum est un choix binaire effectué par graph-cuts. Après convergence, on obtient un minimum local.

(NB : chaque itération est résolue de façon globalement optimale)

Séance 2
Géométrie Epipolaire

(NB : - on note A^T la transposée
- $\wedge$ est le produit vect.
- par rapport au cours, les ~ ont disparu)

1) Géométrie projective

1.1) $\mathbb{P}^2$ (plan projectif)

Point : $(u,v) \longrightarrow m=(u,v,1)^T$ à $\lambda \neq 0$ près (coord. homogènes)

Direction : $u\vec{i}+v\vec{j} \longrightarrow m=(u,v,0)^T$ (point à l'∞)

Droite $l=(a,b,c)^T$ avec $\boxed{m \in l \Leftrightarrow m^T l = 0}$

Propriétés :

i) m_1 et m_2 donnés, $m_1 \wedge m_2$ est la droite (m_1, m_2)

ii) l_1 et l_2 données, $l_1 \wedge l_2$ est leur intersection même si elles sont //.

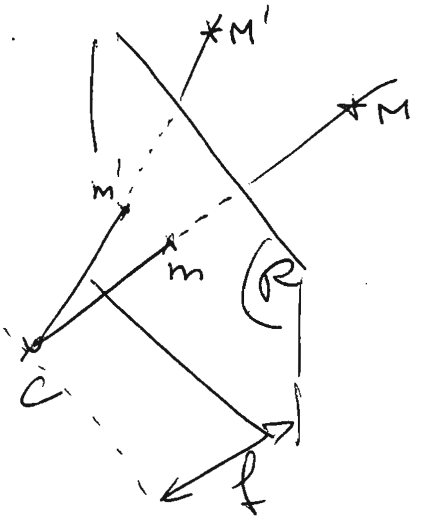
iii) $l_\infty = (0,0,1)$ est la droite à l'∞ $((u,v,0)^T \in l_\infty)$

1.2) $\mathbb{P}^3$ espace projectif

- Points $M=(X,Y,Z,T)^T$ à $\lambda \neq 0$ près
- $T=0 \rightarrow$ points à l'∞ (directions)
- $\pi=(a,b,c,d)^T$ plan $(M \in \pi \Leftrightarrow M^T\pi = 0)$
- Droites : coordonnées de Plücker (pas vu en cours)

2) Caméra

Le modèle le + simple (pinhole camera)



R : rétine

C : centre optique

M : point 3D

m : image 2D

f = distance focale

(i) Cas où $C=0$, $\mathcal{R}=\{z=1\}$ et le point $(X,Y,1)$ de $\mathcal{R}$ est numérisé en (X,Y) par le capteur.

Alors : $u = X/Z$, $v = Y/Z$ ce qu'on peut écrire

$$\underbrace{m}_{3\times 1} = \underbrace{\begin{pmatrix} I \,|\, 0 \end{pmatrix}}_{3\times 4} \underbrace{M}_{4\times 1}$$ en coordonnées homogènes

(ii) Si maintenant $\mathcal{R}=\{z=f\}$ (focale $\neq 1$) alors

$m = PM$ avec $P=\left(\begin{matrix} f & & \\ & f & \\ & & 1 \end{matrix} \middle| 0\right)$ car $u = f\,X/Z$, $v = f\,Y/Z$

(iii) Si on tient compte maintenant du changement de repère du capteur : $u = \alpha_u X/Z + u_0$, $v = \alpha_v Y/Z + v_0$

soit $P=\left(\begin{matrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{matrix} \middle| 0\right)$ (NB : $\alpha_u = f/\text{larg. pixel}$, $\alpha_v = f/\text{haut. pixel}$)

(iv) Maintenant l'observateur se déplace $\rightarrow$ R rotation, t translation

$$\boxed{\begin{array}{l} m = PM \text{ avec } P = K[R|t] \rightarrow 6 \text{ param. extrinsèques} \\ K = \begin{pmatrix} \alpha_u & & u_0 \\ & \alpha_v & v_0 \\ & & 1 \end{pmatrix} \rightarrow 4 \text{ param. intrinsèques} \end{array}}$$

(NB : si pixels non rectangles (ie. ▱) alors un 5° param. intrins.)
(NB : On a négligé la distortion)

3) Relations utiles

i). Rayon de m :

– en coord. non homogènes pour M et avec $P=(A|b)$

$$\lambda m = AM + b$$

et donc $\boxed{M = \lambda A^{-1} m - A^{-1} b}$

– en particulier, on a (avec $\lambda=0$ et $\lambda=\infty$) :

$C = -A^{-1}b$ (coord. non homog.)

$A^{-1}m$ est la direction du rayon

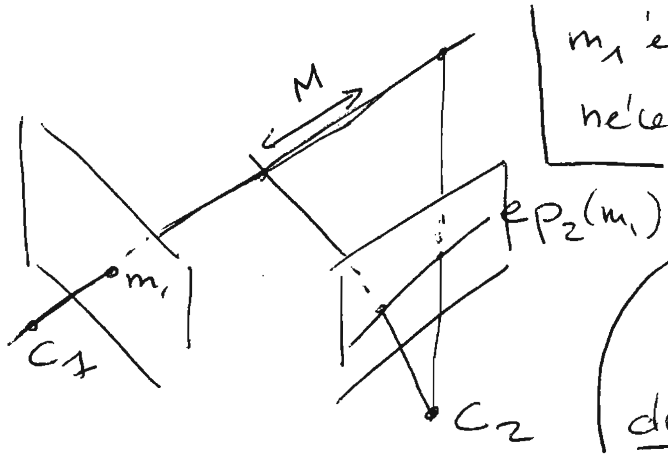
ii) Les points du plan focal (plan // à $\mathcal{R}$ passant par C) se projettent à l'∞, donc en $(u,v,0)^T$

→ la troisième ligne de P est l'équation du plan focal. En particulier :

$\boxed{\text{la troisième ligne de A est la normale à } \mathcal{R}}$

4) Géométrie épipolaire

On a maintenant 2 caméras P_1 et P_2



m_1 étant donné, m_2 est nécessairement sur la projection du rayon C_1m_1 dans l'image 2 : c'est la droite épipolaire de m_1 dans 2

$$\boxed{m_1 \text{ connu} \longrightarrow m_2 \in ep_2(m_1) \text{ "=" } P_2(C_1m_1)\text{"}}$$

Si $\boxed{P_i = (A_i | b_i)}$ alors :

- on cherche deux points du rayon C_1m_1, que l'on projette

$$\begin{cases} \times\ C_1 = \begin{pmatrix} -A_1^{-1}b_1 \\ 1 \end{pmatrix} \rightarrow e_2 = P_2C_1 = \boxed{-A_2A_1^{-1}b_1 + b_2 = e_2} \\ \qquad e_2 \text{ est } \underline{\text{l'épipôle}} \ (\forall m_1,\ e_2 \in ep_2(m_1)) \\ \times\ M_\infty = \begin{pmatrix} A_1^{-1}m_1 \\ 0 \end{pmatrix} \rightarrow m_{2\infty} = P_2M_\infty = A_2A_1^{-1}m_1 \end{cases}$$

- on a alors $ep_2(m_2) = e_2 \wedge m_{2\infty} = [e_2]_\wedge A_2A_1^{-1}m_1$

en posant $\boxed{\begin{bmatrix} a \\ b \\ c \end{bmatrix}_\wedge = \begin{pmatrix} 0 & b & -c \\ -b & 0 & a \\ c & -a & 0 \end{pmatrix}}$

D'où finalement la relation symétrique :

$$\boxed{\begin{array}{l} m_2 \in ep_2(m_1) \Leftrightarrow m_1 \in ep_1(m_2) \Leftrightarrow m_2^T F_{21} m_1 = 0 \\ \text{avec } F_{21} = [e_2]_\wedge A_2 A_1^{-1} \text{ (matrice fondamentale)} \end{array}}$$

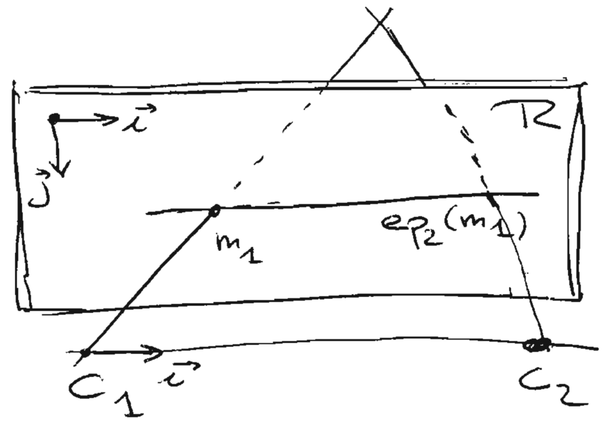
Séance ~~1~~ 3
Rectification / Stéréo binoculaire / Graph Cuts

1) Rectification

On se place ici dans le cas de deux caméras de matrices $P_i = (A_i | b_i)$ connues

1.1) Configuration canonique

Supposons que les deux caméras partagent la même rétine, le même repère sur la rétine, et la direction $\overrightarrow{C_1C_2}$ comme horizontale de ce repère ($\overrightarrow{C_1C_2}$ est donc une des directions de la rétine).
Les épipolaires sont alors les horizontales !



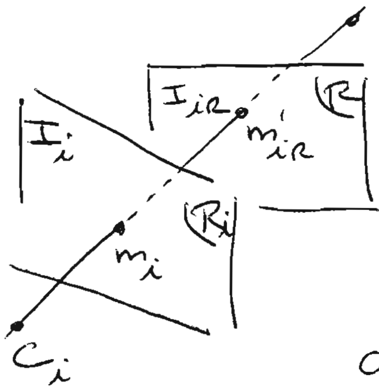
$\mathcal{R} = \mathcal{R}_1 = \mathcal{R}_2$

Une telle configuration est très confortable :
- (u_1, v_1) est associé à (u_2, v_1)
- on appelle $\boxed{d = u_2 - u_1 \text{ la disparité}}$
- apparier les images revient à estimer $d(u, v_1)$
- les fenêtres de corrélation se correspondent davantage.

On va voir qu'il est toujours possible (sauf cas dégénérés) de se ramener a postériori à ce cas canonique → c'est la rectification

1.2) Principe

Il suffit de remarquer qu'on peut à postériori calculer l'image qui aurait été obtenue en changeant la rétine d'un appareil, et ce, sans connaître la position 3D des points



Le point M a donné naissance à $I_i(m_i)$ dans R_i. Avec R au lieu de R_i il aurait généré $\boxed{I_{iR}(m_{iR}) = I_i(m_i)}$

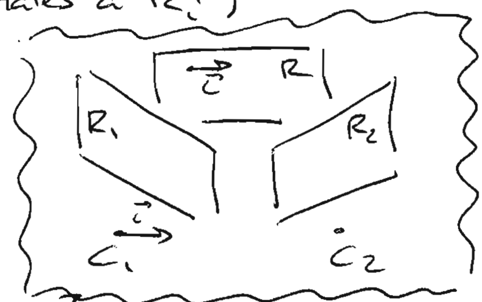
avec $\boxed{m_{iR} = (C_i m_i) \cap R}$

1.3) Calcul (NB : les points 3D sont ici en coord. non homogènes)

a) Plan R

On choisit pour R le plan (i) contenant $\overrightarrow{C_1C_2}$ et la direction $R_1 \cap R_2$ et (ii) passant par l'origine :

- $\vec{i} = \dfrac{\overrightarrow{C_1C_2}}{\|\overrightarrow{C_1C_2}\|}$ (Rappel : $C_i = -A_i^{-1} b_i$)
- $\vec{n}_i =$ 3ème ligne de A_i (normales à R_i)
- $\vec{j}' = \vec{n}_1 \wedge \vec{n}_2$
- $\vec{k} = \vec{i} \wedge \vec{j}' / \|\vec{i} \wedge \vec{j}'\|$
- $\vec{j} = \vec{k} \wedge \vec{i}$



R est alors le plan $\{M \cdot \vec{k} = 0\}$

b) Passage de $m_i = (u_i\ v_i\ 1)^T$ à $m_{iR} = (\alpha, \beta)$ dans R

Il suffit d'intersecter $(C_i m_i)$ avec R :

résoudre $\boxed{\begin{cases} M = \lambda A_i^{-1} m_i + C_i \\ M \cdot \vec{k} = 0 \end{cases}}$ en λ, puis en M

Alors $\boxed{\alpha = M \cdot \vec{i},\ \beta = M \cdot \vec{j}}$

c) Projections

En réalité : (i) pour chaque image, on ramène (α, β) à des intervalles $[0, w] \times [0, h]$ (si les images initiales sont de dimension $w \times h$)

(ii) on remplit I_{iR} à partir de I_i. Il faut savoir passer de m_{iR} à m_i

Au total :

× Pour $i = 1, 2$

- calculer $m_{iR} = (\alpha, \beta)$ pour $m_i = (0,0)\ (0,h)\ (w,0)$ et (w,h)
- mémoriser les valeurs extrêmes $\alpha^i_{min}\ \alpha^i_{max}\ \beta^i_{min}\ \beta^i_{max}$

× $\beta_{min} = \min(\beta^1_{min}\ \beta^2_{min})$, $\beta_{max} = \max(\beta^1_{max}\ \beta^2_{max})$

(NB : intervalle commun pour β)

× Pour $i = 1, 2$

Pour $u_R = 0 \nearrow w$ et $v_R = 0 \nearrow h$

- $\begin{cases} \alpha = \alpha^i_{min} + (\alpha^i_{max} - \alpha^i_{min}) \dfrac{u_R}{w} \\ \beta = \beta_{min} + (\beta_{max} - \beta_{min}) \dfrac{v_R}{h} \end{cases}$
- $M = \alpha \vec{i} + \beta \vec{j}$ (NB : point 3D !)
- $m_i = A_i M + b_i = (u_i v_i w_i)^T$
- $I_{iR}(u_R, v_R) \leftarrow I_i\left(\dfrac{u_i}{w_i}, \dfrac{v_i}{w_i}\right)$

(NB : - interpoler dans I_i
- attention : on peut tomber en dehors de l'image)

I | Mesure de photo consistance

- La plus courante est la corrélation croisée normalisée (NCC)

Soit W un voisinage autour des pixels :

$$\begin{cases} \langle I_i, m_i, I_j, m_j \rangle = \dfrac{1}{|W|} \displaystyle\sum_{w \in W} \Big[\big(I_i(m_i + w) - \overline{I}_i(m_i) \big) \times \big(I_j(m_j + w) - \overline{I}_j(m_j) \big) \Big] \\ \text{avec } \overline{I}_i(m_i) = \dfrac{1}{|W|} \displaystyle\sum_{w \in W} I_i(m_i + w) \end{cases}$$

puis

$$\boxed{NCC(I_i, m_i, I_j, m_j) = \frac{\langle I_i, m_i, I_j, m_j \rangle}{\big(\langle I_i m_i I_i m_i \rangle \langle I_j m_j I_j m_j \rangle \big)^{1/2}}}$$

On a $-1 \leq NCC \leq 1$. Plus NCC est grande, plus les "textures" autour de m_i et m_j sont "similaires".

- Si deux images sont reliées par une transformation affine, $NCC = 1$
- Limitations :
  - invariance seulement par translation
  - pb de l'ouverture : ambiguïtés, par ex. le long d'un contour

II | Points "saillants"

1- Points de Harris :

- G_σ gaussienne
- $H = \begin{pmatrix} (I_x)^2 * G_\sigma & (I_x I_y) * G_\sigma \\ (I_x I_y) * G_\sigma & (I_y)^2 * G_\sigma \end{pmatrix}$

tenseur de structure. (I_x, I_y dérivées)

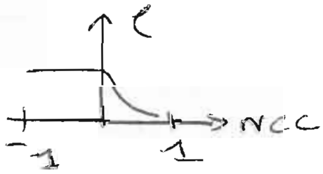
~~(NB : si $V_{pq}(\cdot,\cdot)$ est une semi-métrique (i.e. ne vérifie pas l'inégalité triangulaire), il est possible de procéder par α-β swap (échanger itérativement 2 étiquettes))~~

3) ~~Application à la~~ stéréovision et G Cuts

On suppose que les images sont rectifiées. Il n'y a plus qu'à estimer une disparité $d(u,v)$ optimale.

- $p = (u,v)$
- $d_p \in \mathcal{D} = \{d_{min}, \dots, d_{max}\}$
- p, q voisins $\Leftrightarrow$ voisins dans l'image
- $D_p(d_p) = \rho(NCC(I_1, p, I_2, p + (d_p, 0)))$

  avec ρ fn ≥ 0 décroissante de la corrélation NCC



- $V_{pq}(d_p, d_q) = \lambda |d_p - d_q|$

$\rightarrow$ application directe de 2.3.b (ou de 2.3.c si $V_{pq} = \lambda \mathbb{1}_{\{d_p \neq d_q\}}$)