Vision par ordinateur: Stéréoscopie 1

Sébastien Roy

Département d'Informatique et de recherche opérationnelle
Université de Montréal

Hiver 2007

Au programme

1. Champs aléatoires de Markov
2. Implantation
3. La stéréo comme un graphe
4. Quelques statistiques
5. Quelques résulats
6. Caméras arbitraires
7. Pyramides

Reconstruction 3D

But

Calculer une représentation 3D dense d'une surface à partir de 2 ou plusieurs images d'une même scène.

- Caméras calibrées
- Modèle plus ou moins complet selon le nombre et la position des caméras

Applications

- Mesures 3D
- Interpolation de vues
- Intégration d'objets synthétiques dans des scènes réelles (réalité augmentée)

Stéréoscopie

Application : Interpolation de vue





Application : Intéraction 2D - 3D

aussi appellé *réalité augmentée*...

But

- Intégration d'objets synthétiques 3D dans du vidéo
- Doit être automatique !

Donc...

- La profondeur doit être récupérée automatiquement

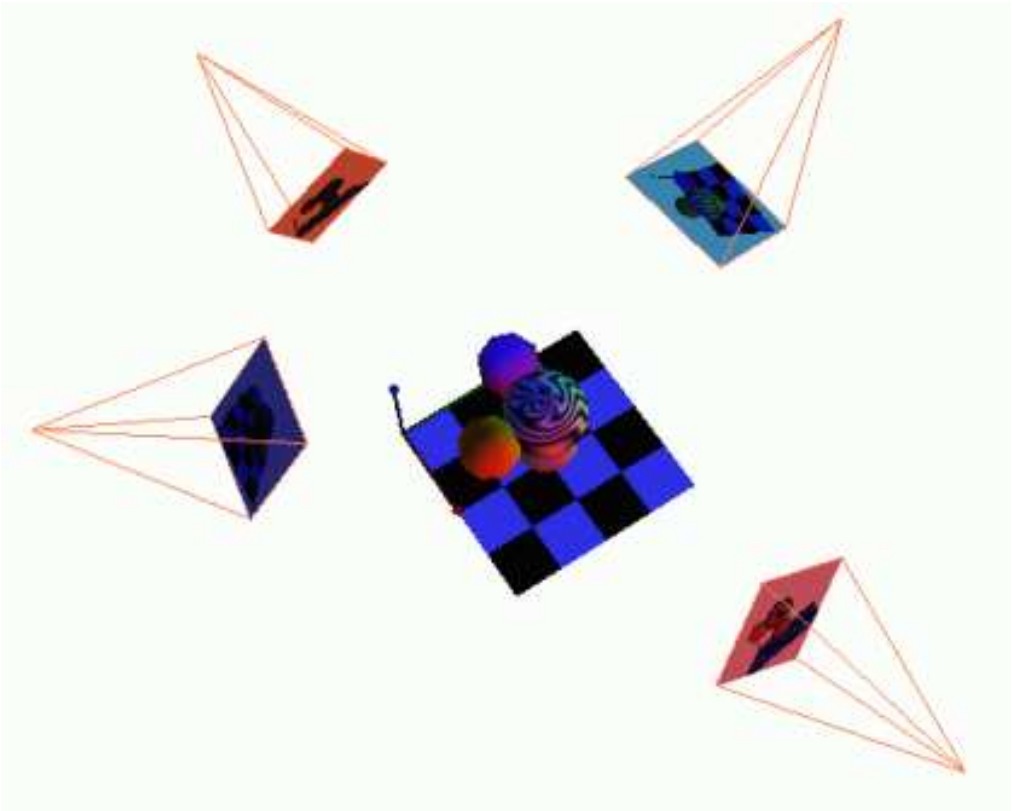
Intéraction 2D - 3D : Nouvel algorithme stéréo

- Traitement particulier des occlusions
- Détermination automatique du niveau de lissage

Reconstruction 3D à partir de vues multiples

Le problème

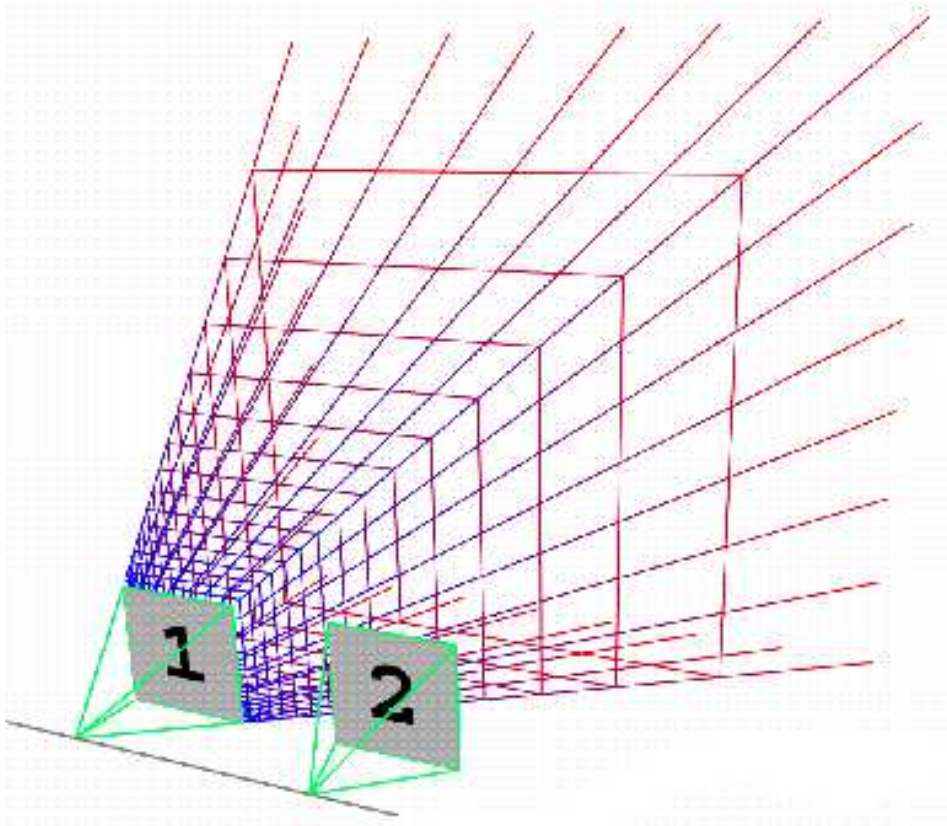
Calculer une représentation 3D dense d'une surface à partir de 2 ou plusieurs images d'une même scène.



2 variantes : occlusions / pas d'occlusion...
ou "Que faire si les caméras se font face ?"

Stéréoscopie traditionnelle

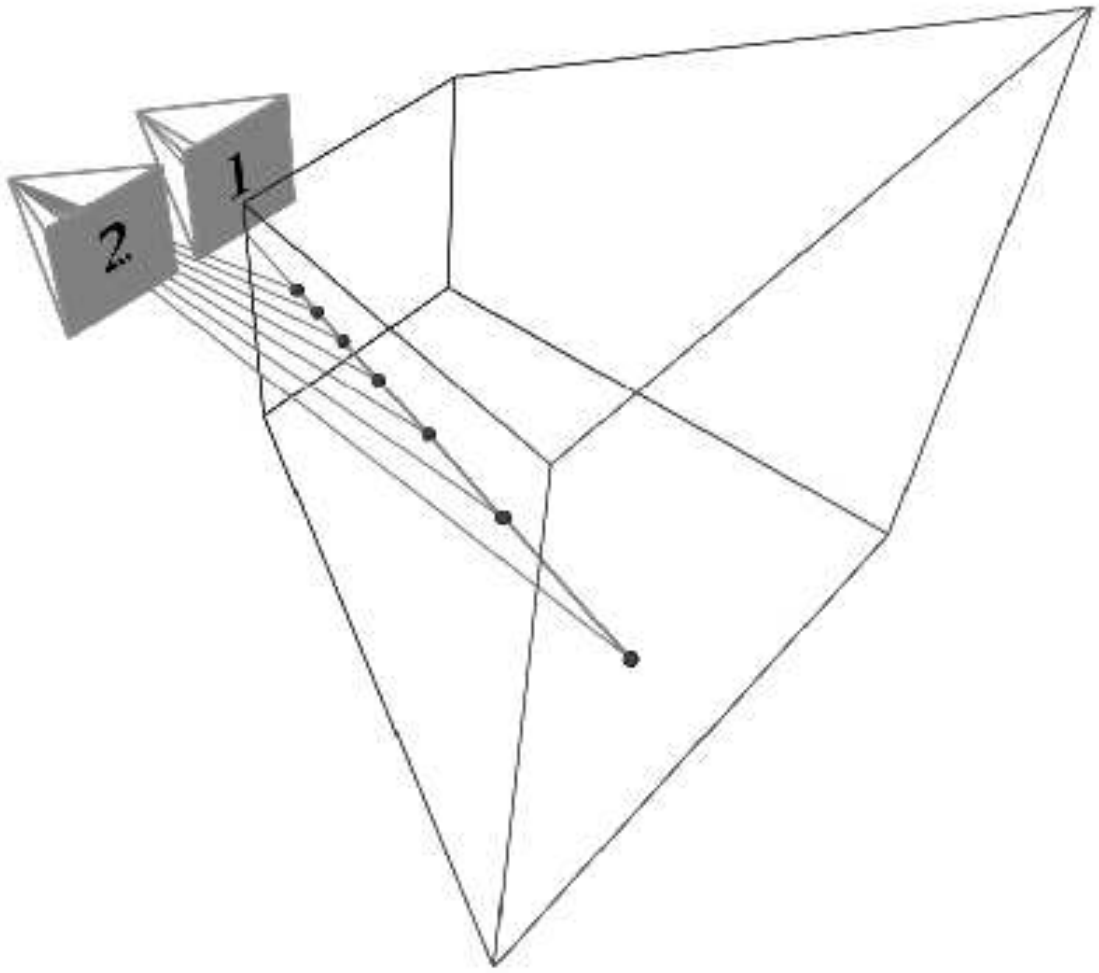
Deux caméras avec axes optiques parallèles et déplacement horizontal.



Important

Points de vue similaires $\longrightarrow$ occlusions négligeables...

Triangulation



- On cherche la correspondance à travers le volume
- On utilise la *triangulation* pour obtenir la profondeur.

Volume de reconstruction

Avec des images multiples, la géométrie épipolaire ne peut pas être utilisée. On utilise plutôt un volume de reconstruction défini directement dans le monde.

La déprojection / reprojection

Chaque caméra est calibrée par la matrice 4×4 $\mathbf{M}_i$ telle que

$$\mathbf{p}'_i = \mathbf{M}_i \mathbf{P}' \quad , \quad \text{typiquement} \quad \mathbf{M}'_i = \begin{pmatrix} & (3 \times 4) & \\ 0 & 0 & 0 & 1 \end{pmatrix}'_i$$

où $\mathbf{P}'$ est un point du monde et $\mathbf{p}'_i$ sa projection dans l'image.

$$\mathbf{P}' = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}' \quad , \quad \mathbf{p}' = \begin{pmatrix} x \\ y \\ w \\ h \end{pmatrix}'_i \equiv \begin{pmatrix} x/w \\ y/w \\ 1 \\ h/w \end{pmatrix}'_i$$

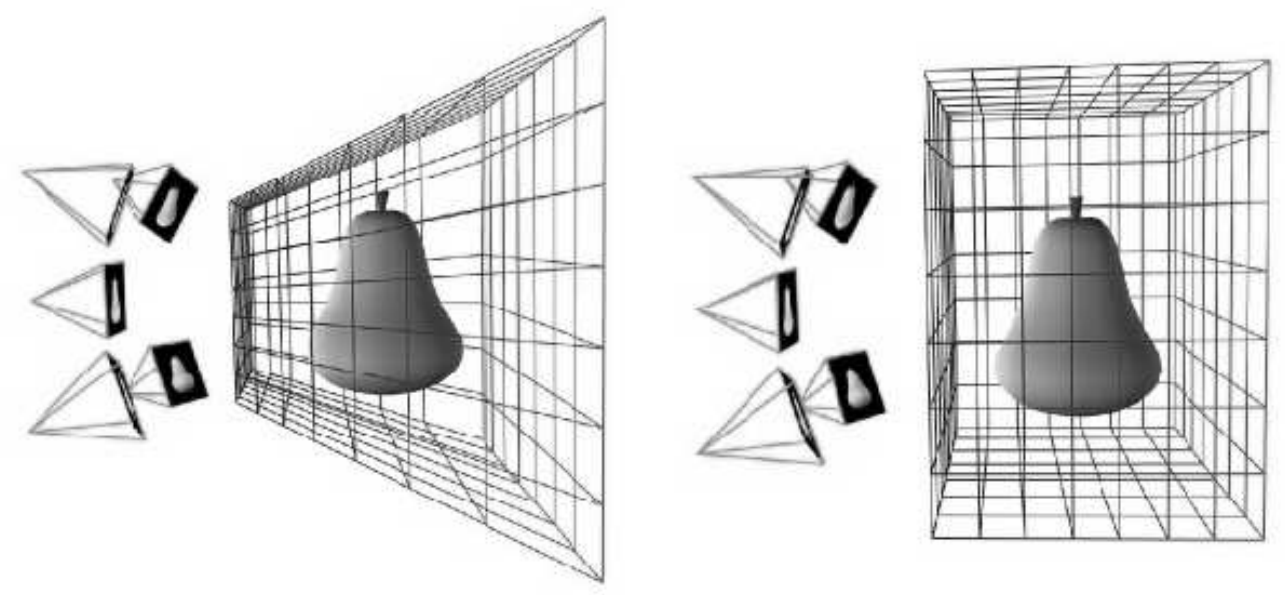
La déprojection / reprojection

Soit un point (x, y) dans la caméra de reférence r, à la profondeur z, on reprojète dans la caméra i par

$$\mathbf{p}'_i = \mathbf{M}_i \, \mathbf{M}_r^{-1} \, \mathbf{p}'_r = \mathbf{M}_i \, \mathbf{M}_r^{-1} \begin{pmatrix} x \\ y \\ 1 \\ 1/z \end{pmatrix}'_r$$

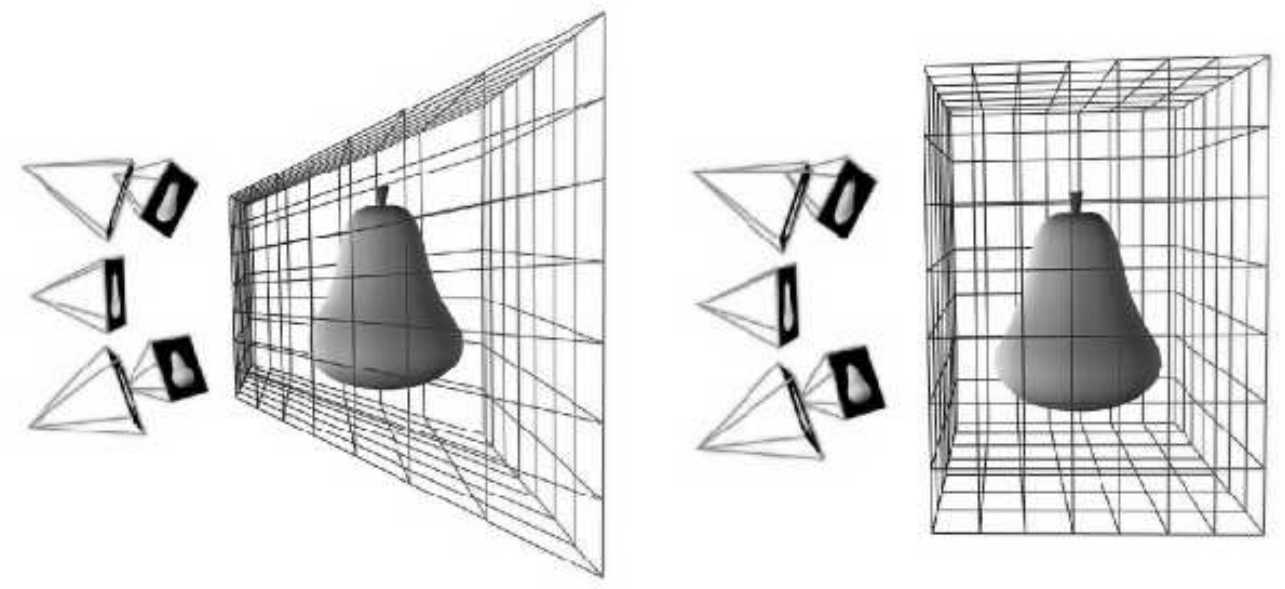
Volume de reconstruction

Volume contenant la surface reconstruite.



- Il doit contenir un *avant* et un *arrière*.
- Les caméras sont toujours du même coté du volume.
- La forme peut changer....

Reconstruction 3D VS Stéréoscopie



Stéréoscopie

Les caméras sont toutes du même coté de l'*avant* du volume

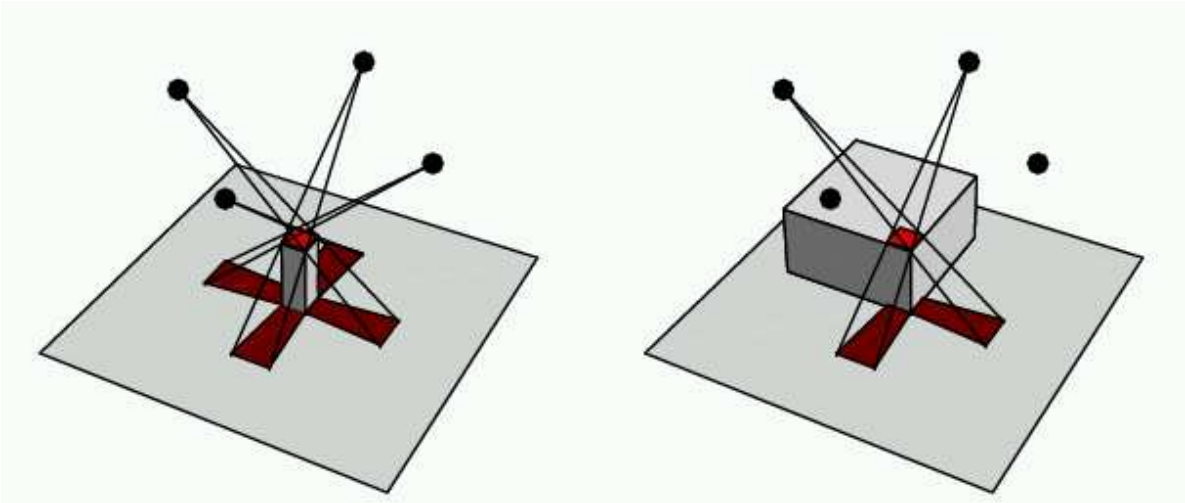
Reconstruction 3D générale (ou volumétrique)

Les caméras sont arbitrairement situés autour du volume

La différence ? $\rightarrow$ **Les occlusions**

Occlusions

Les différences de profondeur font que certains pixels en cachent d'autres...

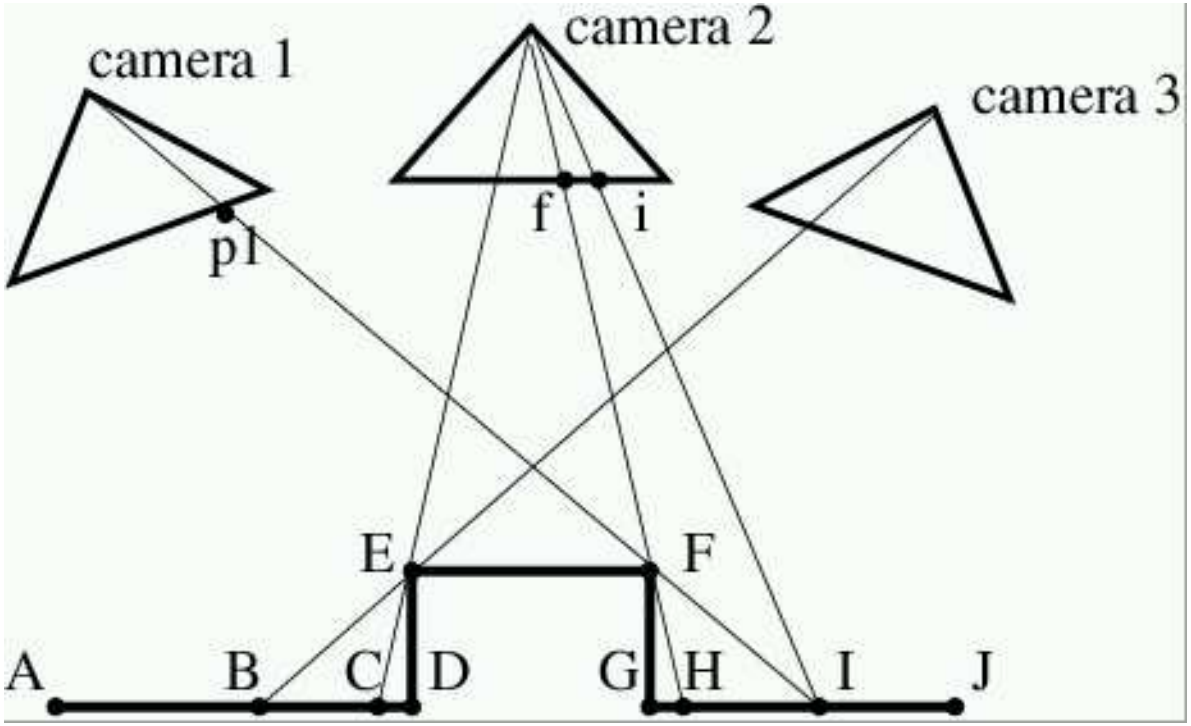


Un pixel caché est dit *en occlusion* pour une caméra donnée.

En stéréo conventionelle (2 caméras)

- Point visible d'une seule caméra : demi-occlusion
- Point visible d'aucune caméra : occlusion complète

Occlusions



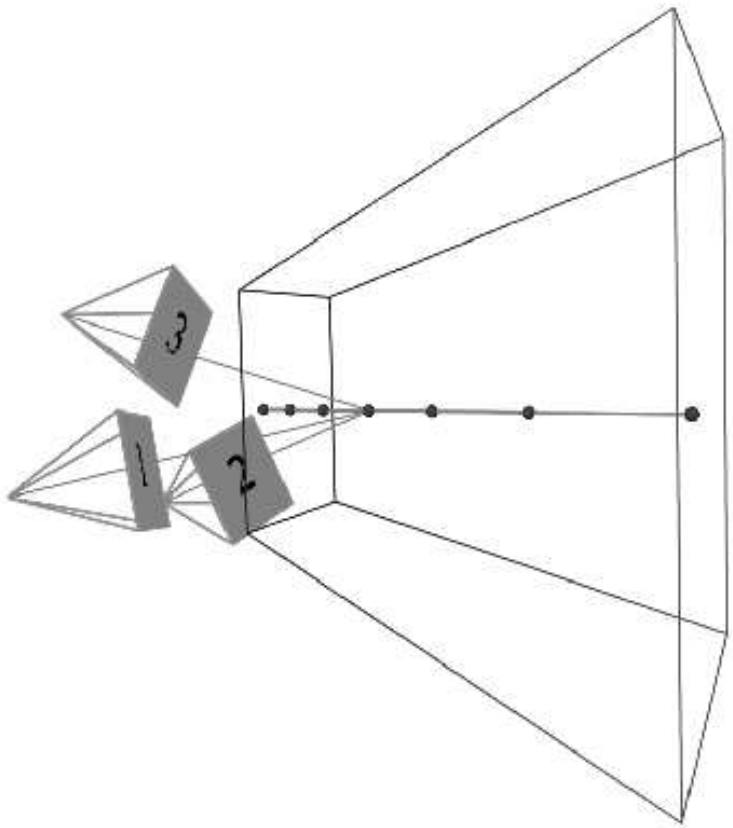
A-B, E-F, I-J : Entièrement visible.

B-C-D, G-H-I : Occlusions partielles.

Idéalement, on ne veut pas utiliser une caméra dans la mise en correspondance si elle ne peut pas voir le point....

Coût de mise en correspondance

On associe une probabilité *d'être à la surface* à chaque point du volume de reconstruction par une fonction de coût.



Comment choisir ce coût ?

Le choix d'un coût de mise en correspondance

Le coût est calculé indépendamment pour chaque pixel !
Soit v_i l'intensité $I_i(\mathbf{p}'_i)$.
Les coûts typiquement utilisés sont

$$c = \sum |v_i - v_r| \quad c = \sum |v_i - v_r|^2$$
$$c = \sum |v_i - \bar{v}| \quad c = \sum |v_i - \bar{v}|^2$$

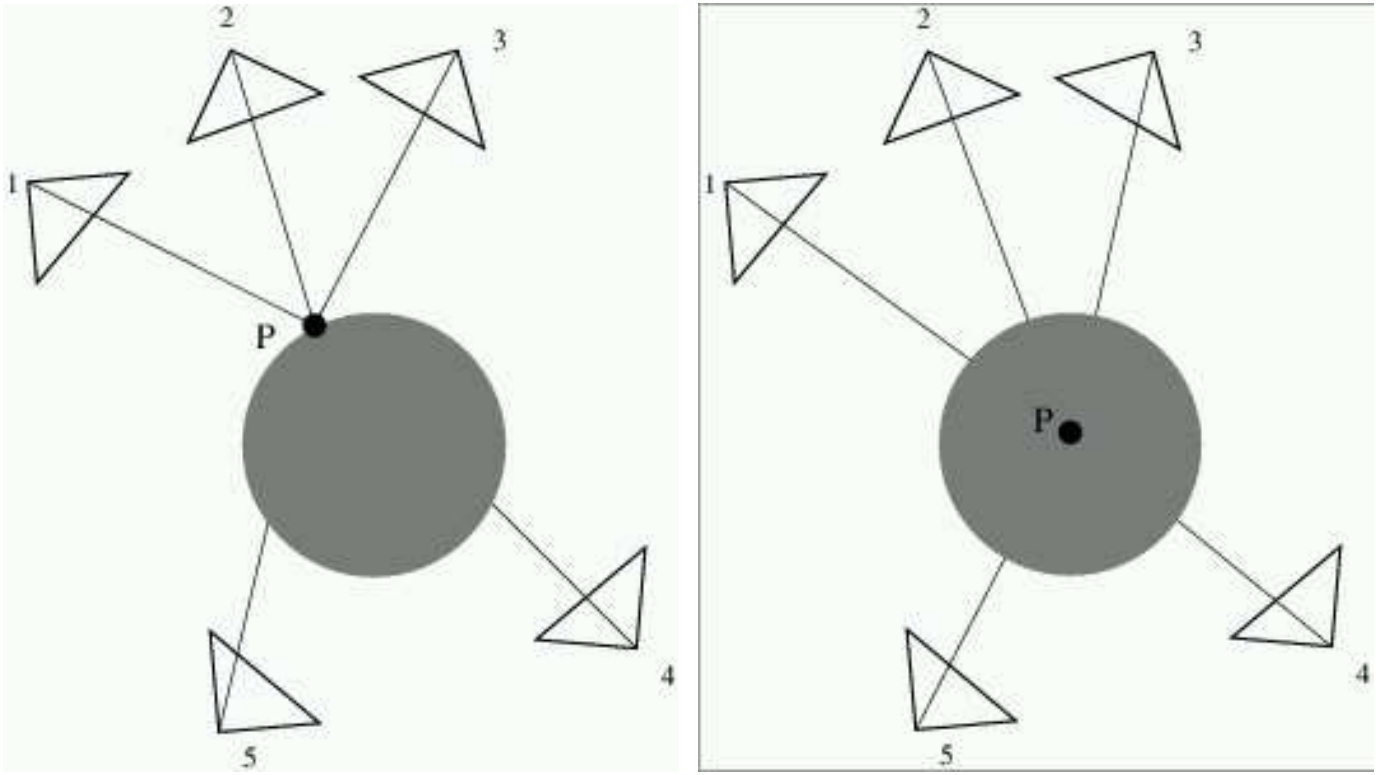
Bref,

$$c = f(\{v_1, v_2, \ldots, v_n\})$$

Ces modèles sont-ils réalistes ?

Coût de mise en correspondance

- Un point sur la surface se reprojète au bon endroit dans les images, $\Rightarrow v_1, v_2, \ldots, v_n$ sont identiques.
- Plus un point est loin de la surface, plus les reprojections sont différentes.



Mise en correspondance : Hypothèses

Opacité

Les objets de la scène doivent être opaques.

Intensité constante

L'intensité observée d'un objet ne change pas lorsque la caméra se déplace.

Pas d'occlusion

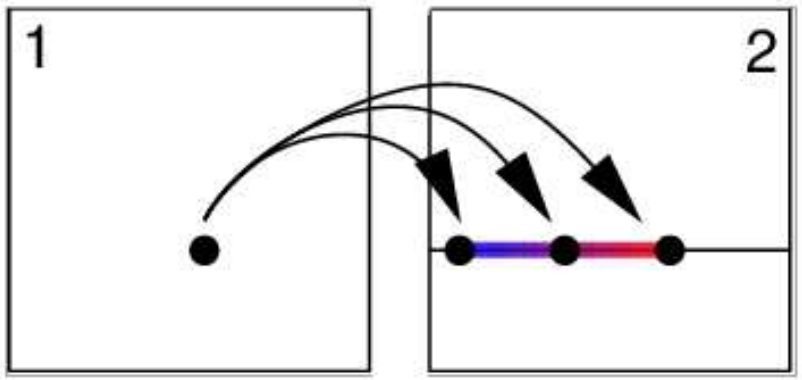
Tous les pixels d'une image sont visibles dans l'autre image.

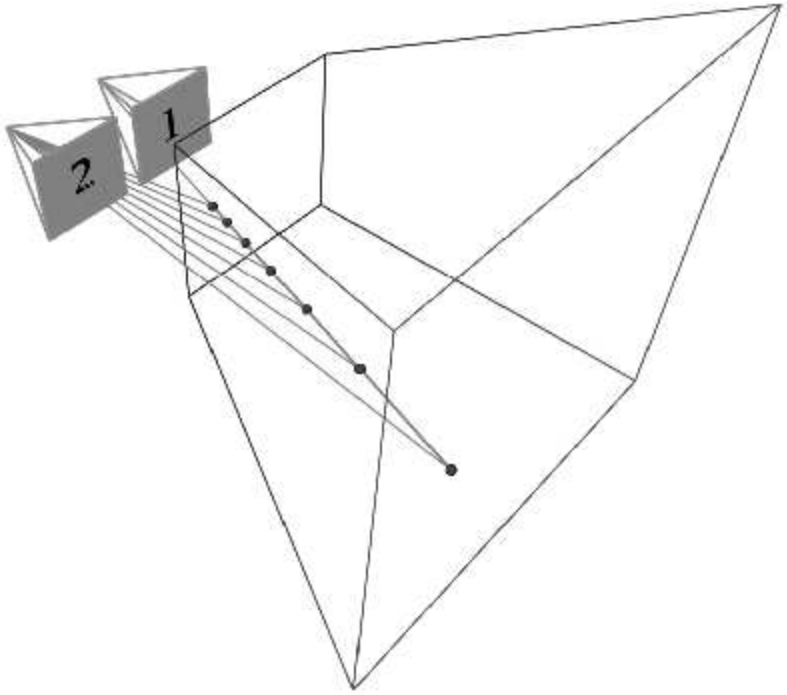
$\longrightarrow$ Le déplacement de la caméra doit être faible.

$\longrightarrow$ L'illumination est fixe par rapport à l'objet.

$\longrightarrow$ Les objets de la scène sont lambertiens.

Mise en correspondance : Stéréo traditionnelle





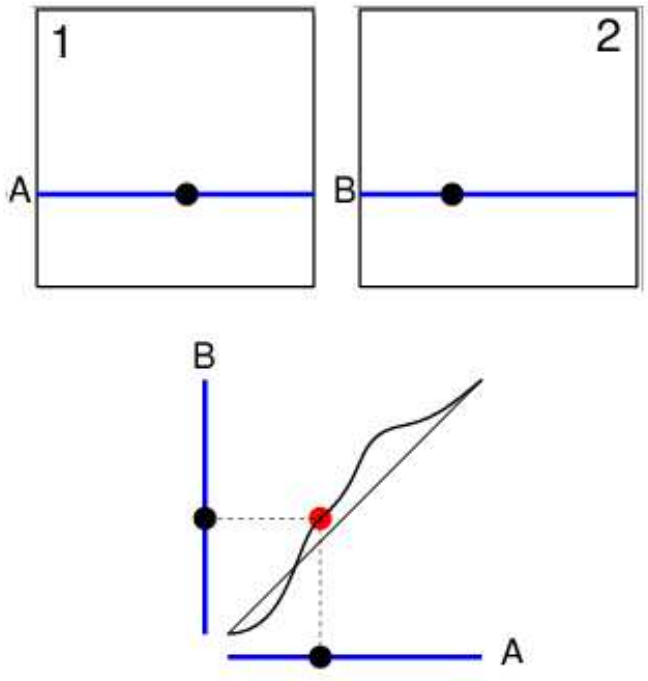
Recherche directe

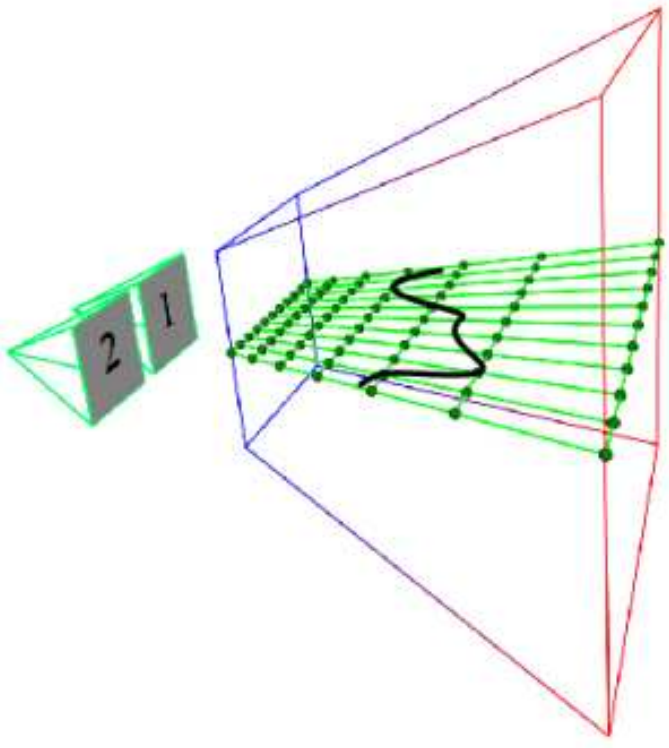
Pour chaque pixel de l'image 1, on cherche le pixel le plus *semblable* dans l'image 2. [Kanade & Okutomi 94]

$\rightarrow$ La solution est très bruitée

$\rightarrow$ Plein d'ambiguités locales !

Programmation dynamique





Programmation dynamique

Mise en correspondance de deux lignes épipolaires.

- Trouve efficacement une solution globale par ligne
- Lissage appliqué le long des lignes, pas entre les lignes

Nouveau ! : Le lissage...

Modèles

La fonction de coût doit tenir compte des propriétés du monde.
Propriétés utilisables (entre autres) en stéréo :

- Pixels correspondants $\equiv$ intensités similaires
- Pixels voisins $\equiv$ disparités similaires

La fonction de coût prendra la forme d'une énergie contenant plusieurs termes à minimiser.

$$E_{totale}() = E_{correspondance}() + E_{lissage} + E_{...}$$

Reste à trouver un minimum global...

Sommaire

Champs aléatoires de Markov

Sites (pixels)

Un ensemble de sites $S = \{1, 2, \ldots, m\}$

Étiquettes (disparités)

Un ensemble d'étiquettes $L = \{1, 2, \ldots, n\}$

Champs aléatoire

Une famille de variable aléatoire $F = \{F_1, F_2, \ldots, F_m\}$ définies sur les sites S. Une configuration f est une réalisation de ce champs.

Voisinage

La relation entre les f_i est déterminé par un voisinage $N = \{N_i : i \in S\}$

Champs aléaoires de Markov

On a une densité sur les configuration $P(F = f)$ qui représente ce qu'on connait *d'avance* (*a priori*) sur le monde.
Comment définir $P(F = f)$?

Propriété markovienne

$$Pr(F_i \mid F_j,\, j \in S \setminus \{i\}) = Pr(F_i = f_i | F_j = f_j,\, j \in N_i)$$

Bref, la probabilité jointe qu'une étiquette f_i soit donnée au site i ne dépend que des voisins de i.
$\rightarrow$ on peut représenter le lissage entre pixels voisins.

Champs de Markov, Champs de Gibbs

Théorème de Hammersley-Clifford

Champs de Markov $\equiv$ Champs de Gibbs.

Dans un champs de Gibbs, on a

$$p(F = f) = \frac{e^{-U(f)}}{Z}$$

où $U(f)$ est somme des potentiels de clique

$$U(f) = \sum_{c \in \mathcal{C}} V_c(f)$$

Cliques

Une clique est un sous-ensemble de sites. Dans notre cas, on s'intérèsse aux cliques de 2 sites.
Rappel : c'est ce qu'on connait *a priori* sur la scène qui est représenté par les potentiels de cliques.

Clique de 1 site

Rien de connu *a priori* pour un seul pixel.

Clique de 2 sites

Deux sites voisins ont des étiquettes similaires (lissage).

Clique de 3 sites

Hummm...

Étiquettage bayesien

On souhaite trouver la configuration *a posteriori* (MAP) la plus probable.
Soit les observations X (les images), on a

$$P(F = f \mid X = x) \quad \propto \quad P(X = x \mid F = f)\, P(F = f)$$

où $P(X = x \mid F = f)$ est la vraisemblance et $P(F = f)$ est la distribution *a priori*.
Une hypothèse d'indépendance nous permet de poser

$$P(X = x \mid F = f) \quad = \quad \prod_{i \in S} P(X_i = x_i \mid F_i = f_i)$$

Vers une minimisation d'énergie

$$
\begin{aligned}
& \max\left[\, P(F = f \mid X = x) \,\right] \\
= \quad & \max\left[\, P(X = x \mid F = f)\, P(F = f) \,\right] \\
= \quad & \max\left[\, P(X = x \mid F = f)\, e^{-U(f)} \,\right] \\
= \quad & \max\left[\, \prod_{i \in \mathcal{S}} P(X_i = x_i \mid F_i = f_i)\, e^{-U(f)} \,\right] \\
= \quad & \max\left[\, \ln\left(\prod_{i \in \mathcal{S}} P(X_i = x_i \mid F_i = f_i)\, e^{-U(f)} \right) \,\right] \\
= \quad & \min\left[\, U(f) - \sum_{i \in \mathcal{S}} \ln\left(P(X_i = x_i \mid F_i = f_i)\right) \,\right] \\
= \quad & \min\left[\, \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{N}_i} V_{\{i,j\}}(f_i, f_j) - \sum_{i \in \mathcal{S}} \ln\left(P(X_i = x_i \mid F_i = f_i)\right) \,\right] \\
= \quad & \min\left[\, E(f) \,\right]
\end{aligned}
$$

Vers une minimisation d'énergie

Dans notre fonction d'énergie

$$E(f) = \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{N}_i} V_{\{i,j\}}(f_i, f_j) - \sum_{i \in \mathcal{S}} ln(Pr(X_i = x_i \mid F_i = f_i))$$

Le coût de lissage est typiquement

$$V_{\{i,j\}}(f_i, f_j) = v(i,j)\ |f_i - f_j|$$

Le coût de mise en correspondance est typiquement

$$ln(P(X_i = x_i \mid F_i = f_i)) = c(i, f_i)$$

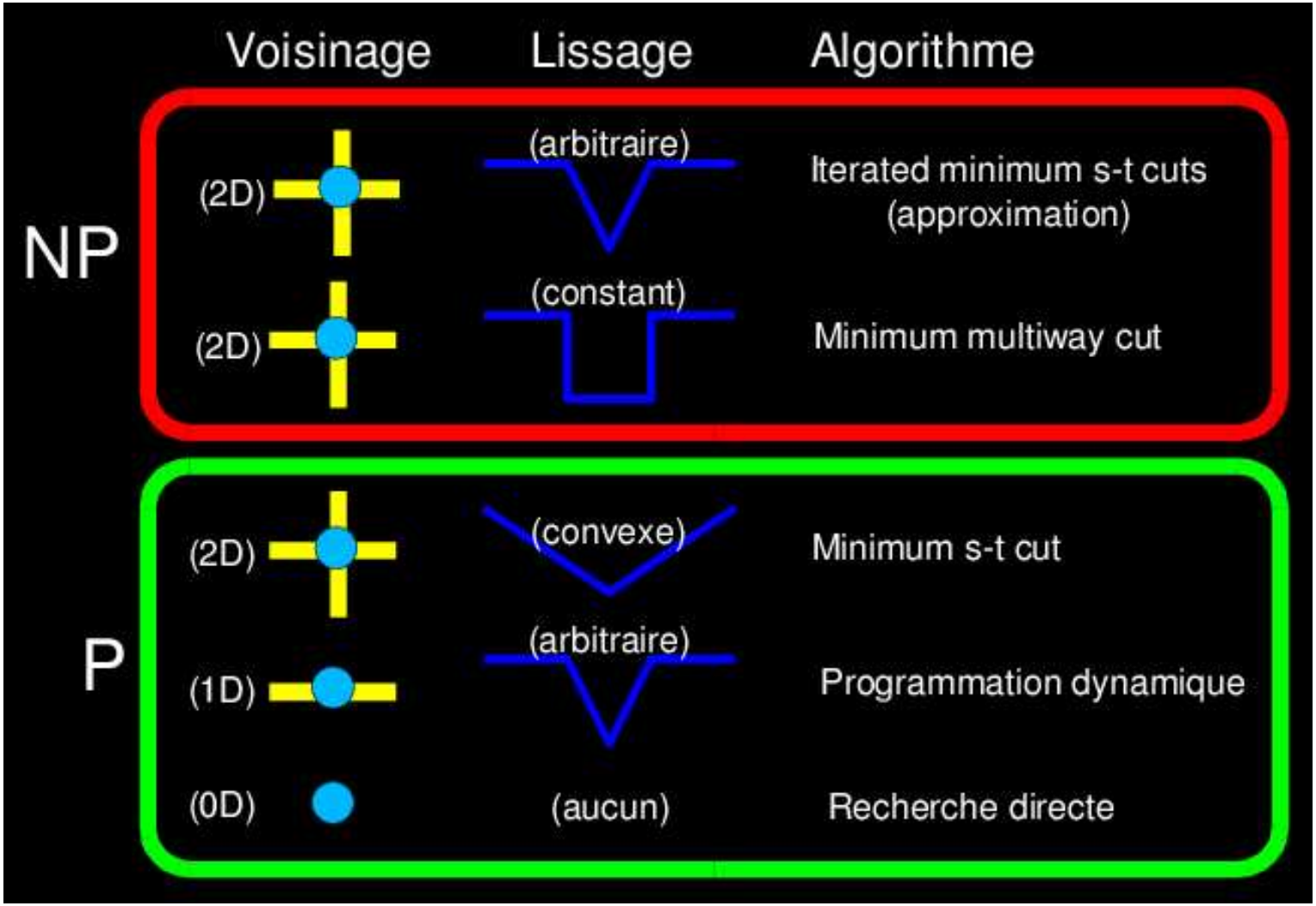
donc le coût d'associer la disparité f_i au pixel i, en tenant compte du contenu des images (X).

Résultats important

Si le lissage est de la forme $v(i,j)|f_i - f_j|$, alors on peut trouver la configuration la plus probable **globalement** et **efficacement** en utilisant le flot dans les graphes.
Si le lissage est d'une autre forme, probablement NP-complet.

Minimisation d'énergie

Complexité du modèle VS Complexité de l'algorithme



Sommaire

Discrétisation de la disparité

On doit choisir parmis des profondeurs discrètes. Comment bien choisir les étiquettes ?

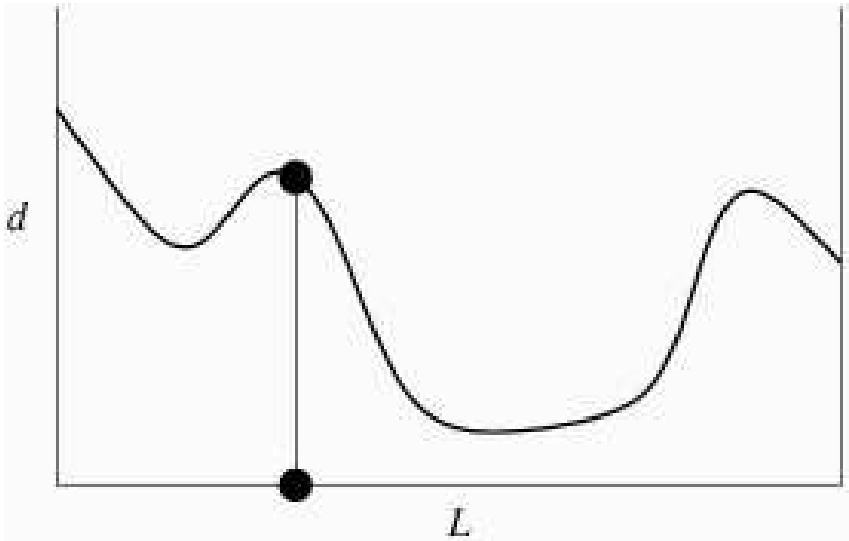
$\rightarrow$ Uniforme en Z, uniforme en $1/Z$, non-uniforme...

Uniforme : Pour un nombre d_s de pas désirés, on associera les valeurs discrètes $d_i = [(0, 1, \ldots, d_s - 1]$ aux étiquettes de l'interval $d = [d_1, d_2]$.
Conversion entre une disparité discrète d_i et une disparité réelle d :

$$d = d_i \frac{d_2 - d_1}{d_s - 1} + d_1 \quad d_i = \text{round}\left((d - d_1)\frac{d_s - 1}{d_2 - d_1}\right)$$

Programmation dynamique

À cause de la géométrie épipolaire commune à l'ensemble des caméras, on peut résoudre indépendement la mise en correspondance sur chaque ligne L (horizontale) de l'images.



- Il faut choisir une ligne (horizontale) L
- On travaille maintenant avec l'espace de mise en correspondance $L \times d$.
- On doit trouver le chemin de coût minimum reliant la gauche et la droite de l'espace $L \times d$.

Programmation dynamique

Pourquoi la ligne L doit-elle être horizontale ?

$\Rightarrow$ Pour être alignée avec la géométrie épipolaire ! (en général horizontal, mais pas toujours)

On peut choisir une ligne L quelquonque :

- Si L est la ligne épipolaire, elle impose la contrainte d'ordre
- Si L est différente, elle impose la contrainte de lissage

Bref, ca revient presqu'au même....

Programmation dynamique

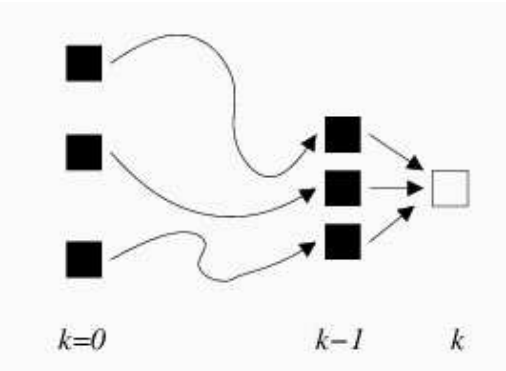
Pourquoi la ligne L doit-elle être horizontale ?
$\Rightarrow$ Pour être alignée avec la géométrie épipolaire ! (en général horizontal, mais pas toujours)
On peut choisir une ligne L quelquonque :

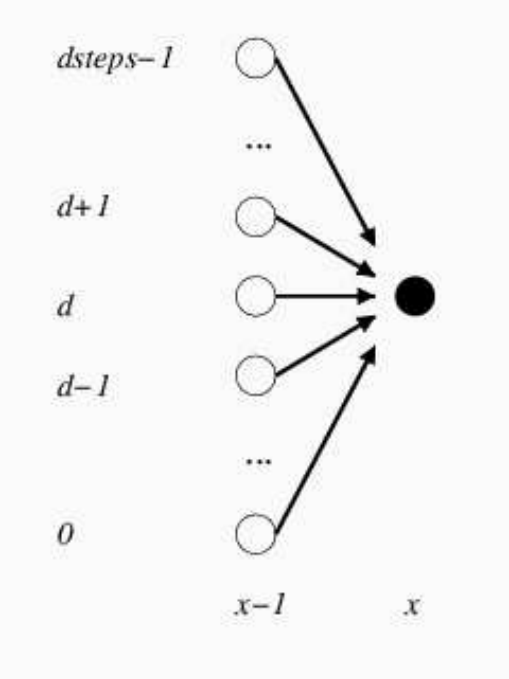
- Si L est la ligne épipolaire, elle impose la contrainte d'ordre
- Si L est différente, elle impose la contrainte de lissage

Bref, ca revient presqu'au même....

Programmation dynamique

Hypothèse : On peut choisir d à la position k seulement à partir des positions précedentes ($< k$).





On détermine récursivement le chemin optimal vers (x, d) en complétant les chemins optimaux rejoingnant les noeuds $(x-1, 0), \ldots, (x-1, d), \ldots, (x-1, d_s-1)$.

Chemin optimal

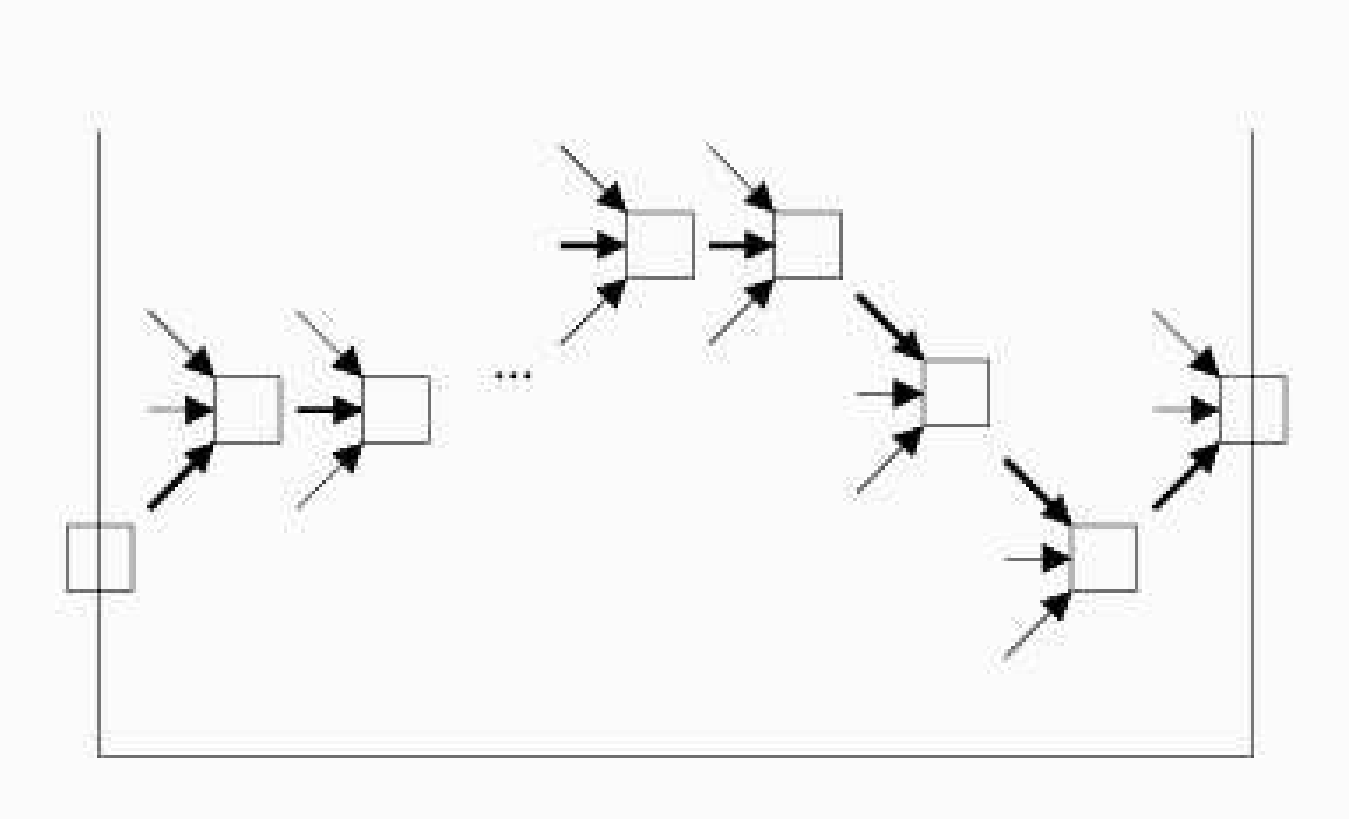
Le coût $opti(x, y)$ du chemin optimal qui rejoint (x, d) est

$$\begin{aligned} opti(0, d) &= c(0, d) \\ opti(x, d) &= \min_{d'}[c(x, d) + opti(x - 1, d') + s(d, d')] \end{aligned}$$

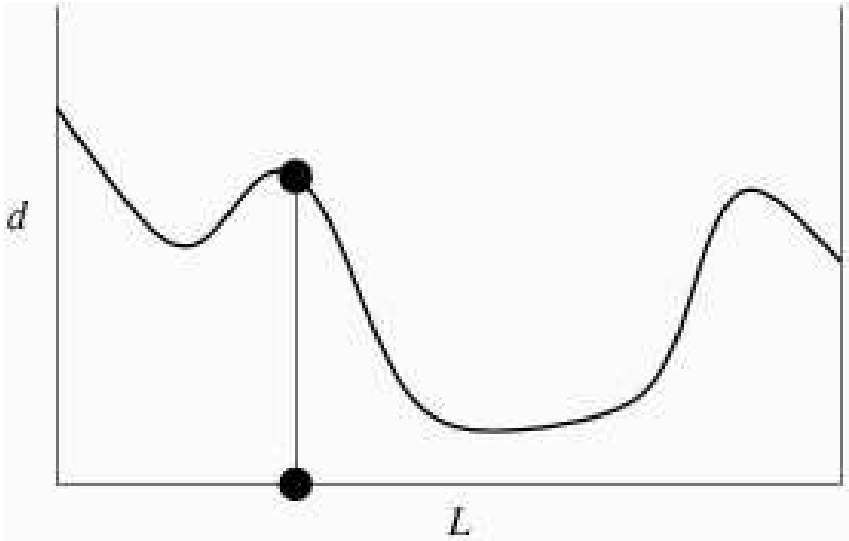
où $c(x, y)$ est le coût de correspondance et $s(d, d')$ est le coût de lissage.
Cette fonction peut être évaluée par programation dynamique en commencant en $x = 0$ et en conservant les valeurs calculées de $opti()$ dans un tableau.
Un fois le tableau rempli, le chemin optimal est celui qui termine en $(X_{size} - 1, d)$ tel que $opti(X_{size} - 1, d) \leq opti(X_{size} - 1, d'), \forall d'$.

Chemin optimal

Pour extraire le chemin optimal, il faut remonter le chemin à partir du point d'arrivée $(d_s - 1, d)$ jusqu'en $x = 0$.



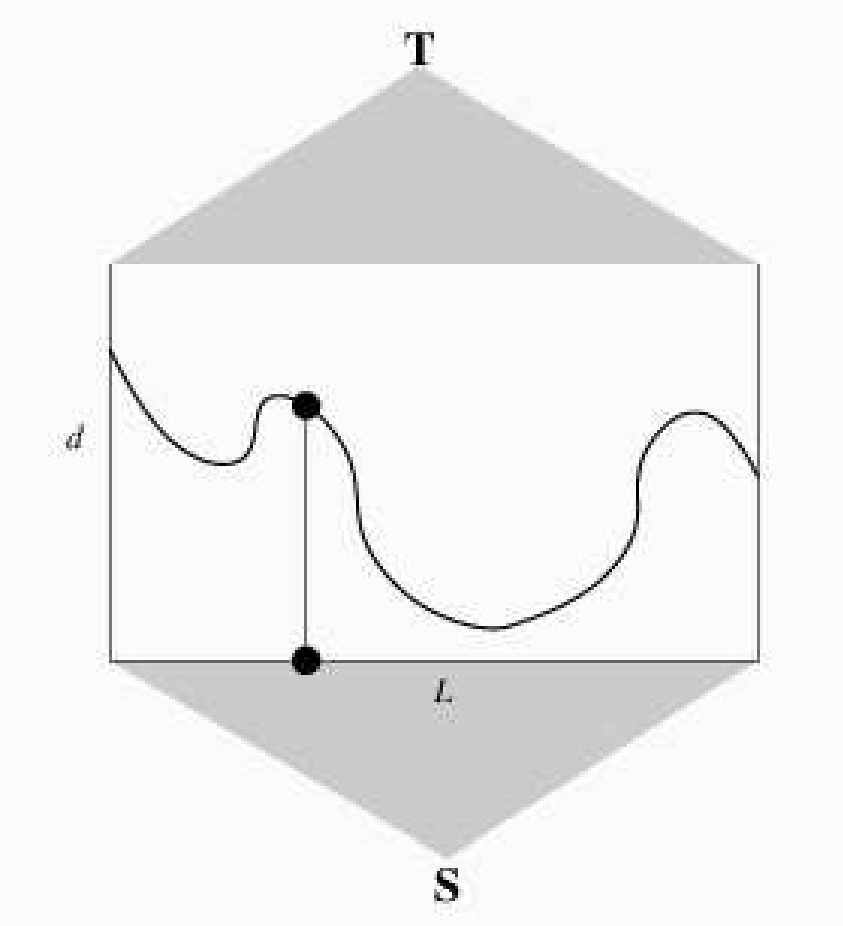
Chemin le plus court



Formulation $\approx$ équivalente à la programmation dynamique.

- Créer un ensemble d'arcs reliant les noeuds du volume V.
- Trier les arcs par coût.
- Créer un ensemble connexe de noeuds $\mathbf{S} = \{(0, d)\}$.
- Ajouter un par un les arcs à $\mathbf{S}$ en conservant la connexité.
- Ajouter jusqu'à ce que $\mathbf{S}$ contienne un noeud (k_{max}, d).

Minimisation du coût : Coupe minimum



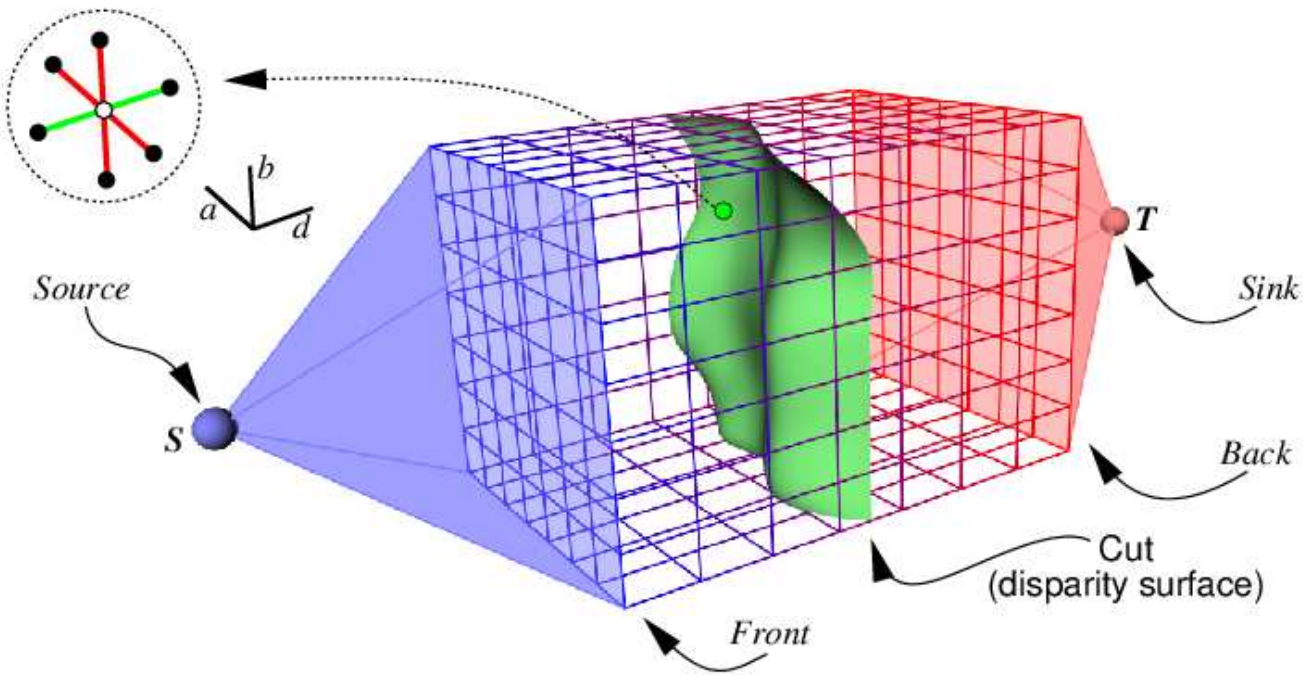
- Créer un ensemble d'arcs reliant les noeuds du volume V.
- Calculer le flot maximum entre la source S et le drain T.
- Extraire du graphe de flot résiduel la coupe de flot minimum.

Sommaire

Stéréo avec flot maximum

Le volume de reconstruction est représenté par un graphe $G = (V, E)$.

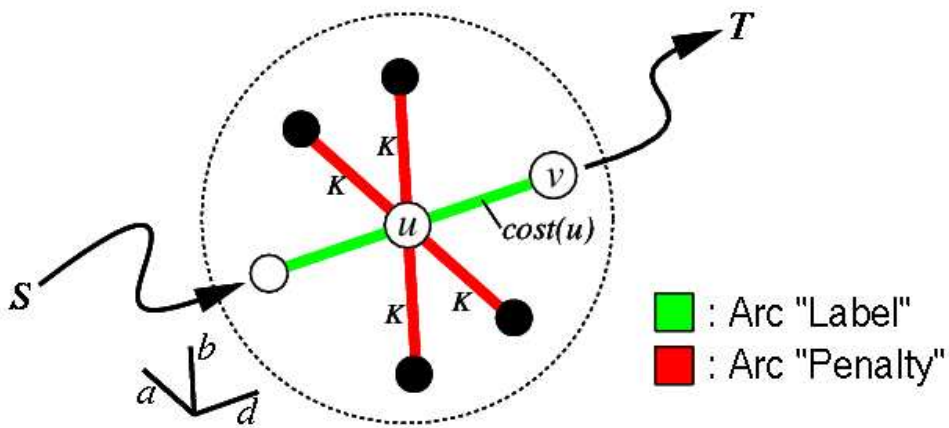
- Une *coupe* du graphe correspond à une surface valide.



Fonction de Mise en correspondance et graphe

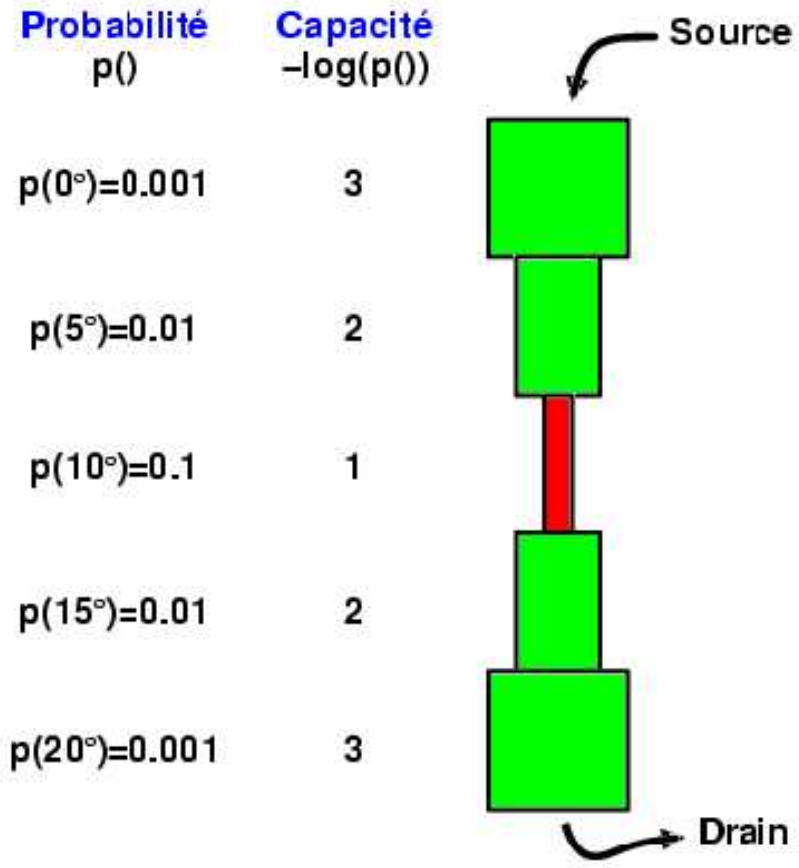
On peut représenter la fonction de mise en correspondance par la capacité des arcs.

- Les arcs selon d (*arcs Label*) contiennent le coût de mise en correspondance.
- Les autres arcs (*arcs Penalty*) représentent le lissage.

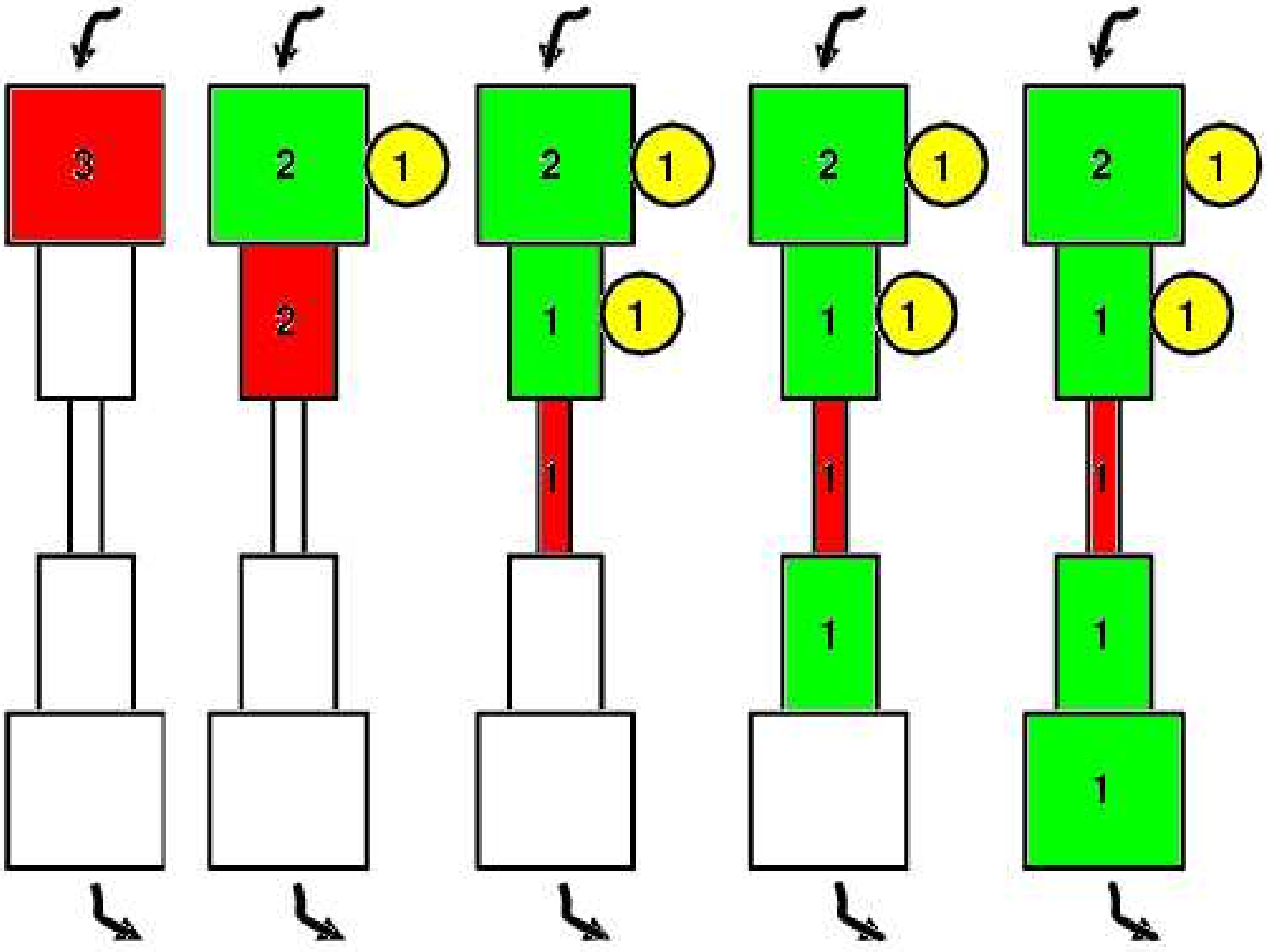


Probabilité et Plomberie

Théorème *Max-flow min-cut*, version simplifiée....

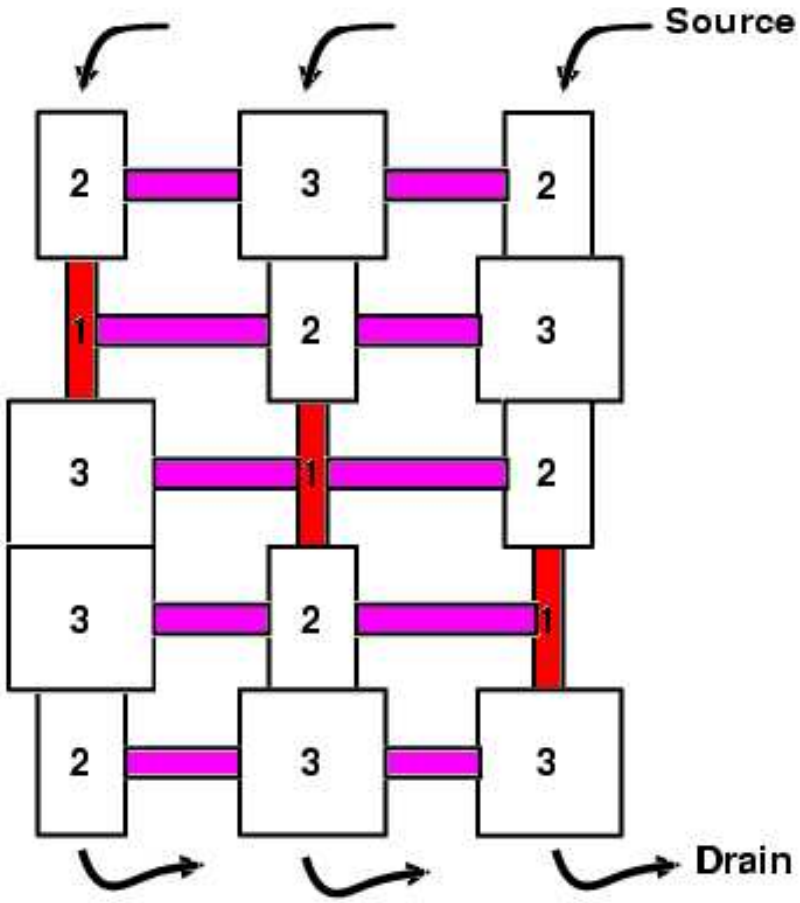


Preflow-push lift-to-front



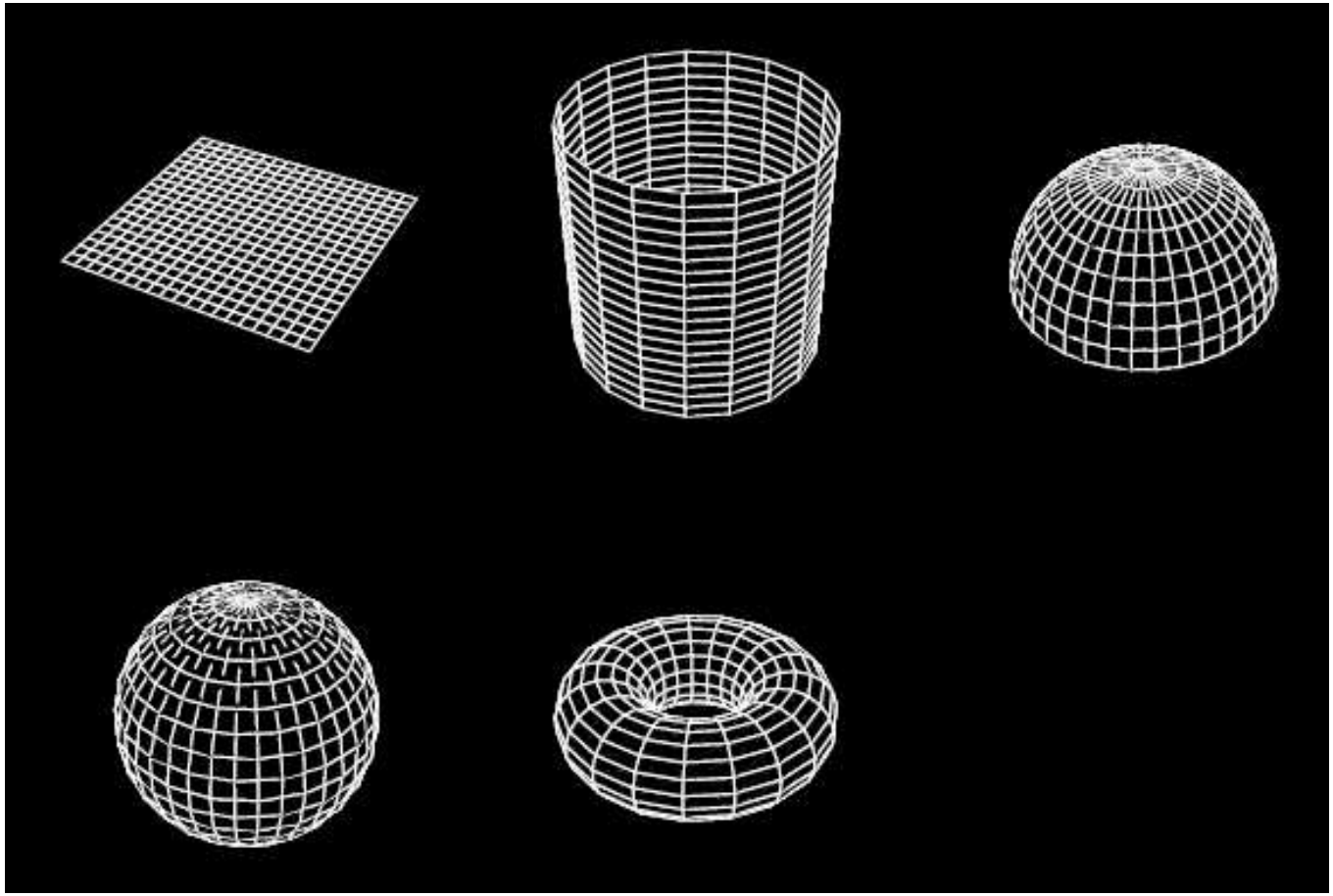
Et le lissage ?

Il suffit d'ajouter quelques tuyaux....



Autres volumes de reconstruction

Il est facile de *connecter* les cotés du volume de reconstruction en ajoutant des arcs au graphe.
On peut donc résoudre pour des images aux formes diverses...



Sommaire

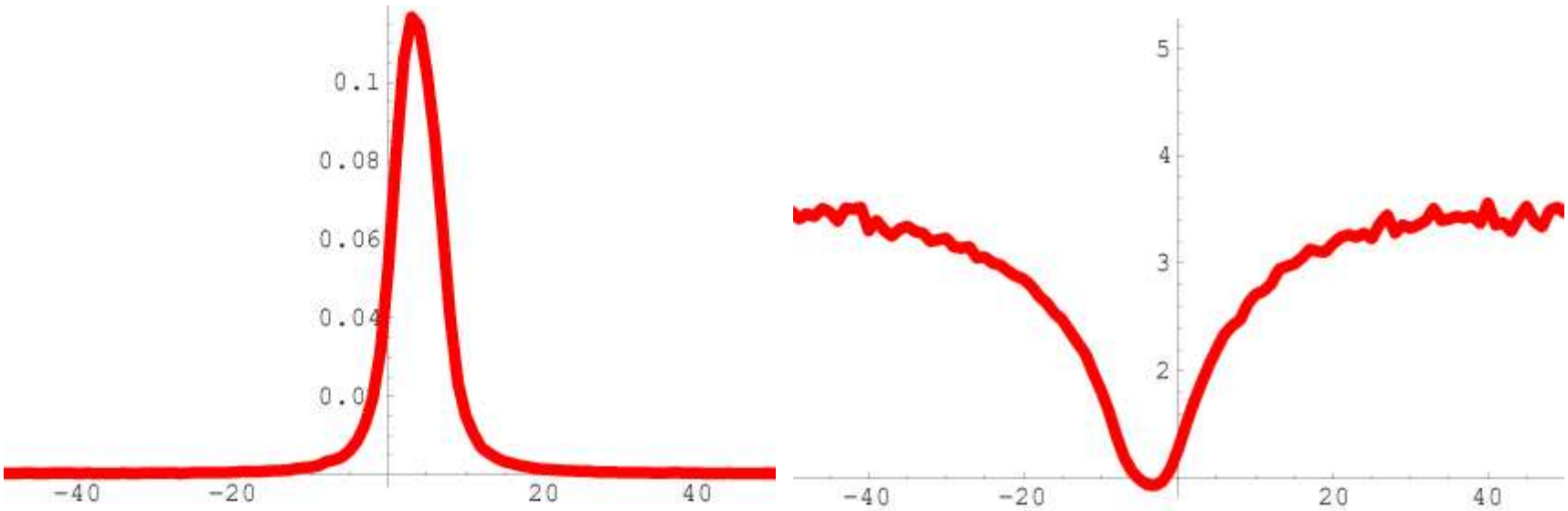
Modèles

On peut vérifier expérimentallement nos modèles sur une séquence avec *ground truth* disponible.

(Scharstein & Szeliski, 2003)

Modèles : correspondance

Cohérence des pixels correspondants
$I_1(p) - I_2(p + d)$

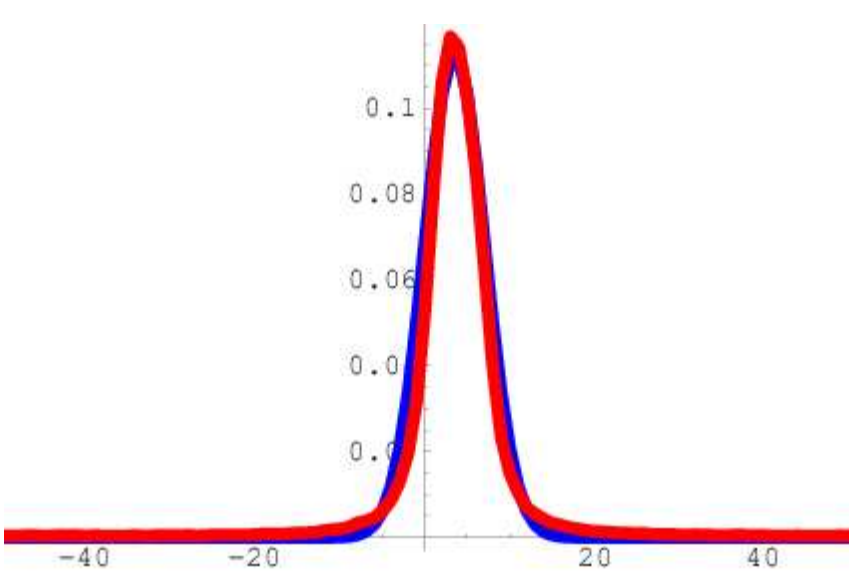


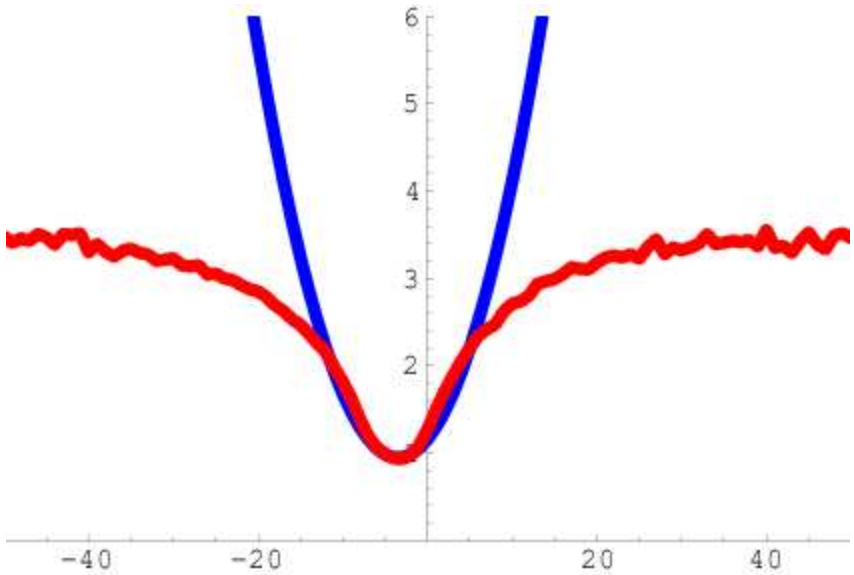
Distribution des différences entre pixels correspondants (à gauche) et -Log de la distribution (à droite).

Modèles : correspondance

Si on utilise une probabilité gaussienne

$$c(\Delta_I) = \left(\frac{\Delta_I - 3}{5}\right)^2 \qquad p(\Delta_I) = e^{-\left(\frac{\Delta_I - 3}{5}\right)^2}$$

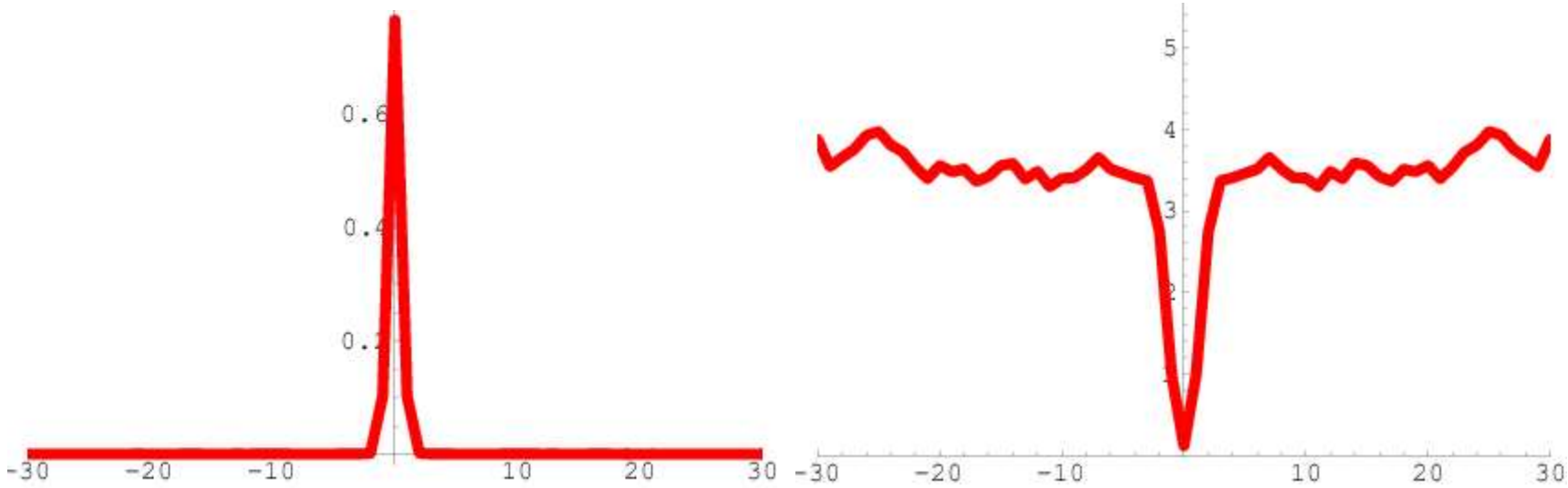




La distribution semble correspondre, mais pas partout.

Modèles : lissage

Cohérence locale des disparité (lissage)
$D(x, y) - D(x \pm 1, y \pm 1)$

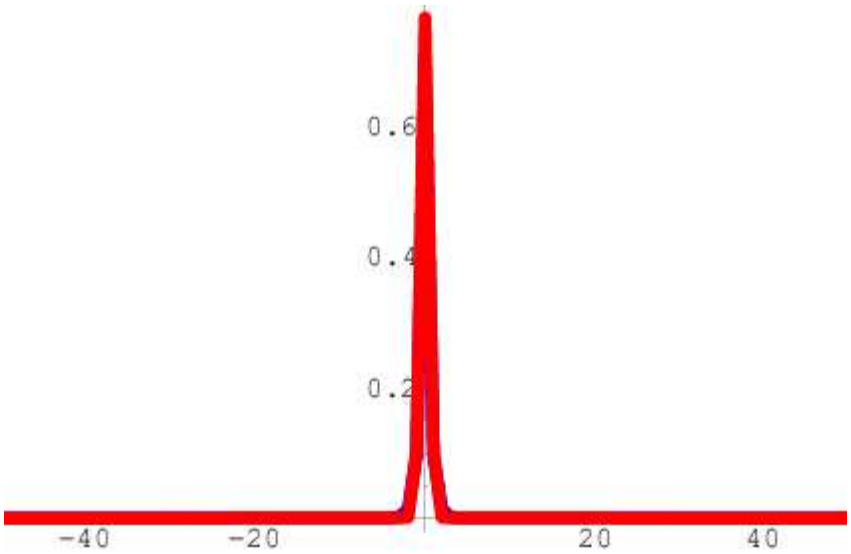


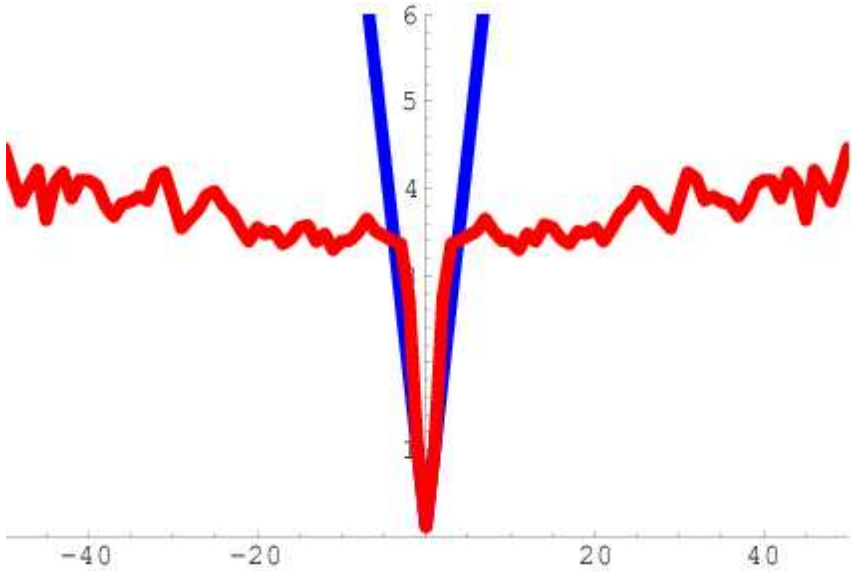
Distribution des différences entre la disparité de deux pixels voisins (à gauche) et -Log de la distribution (à droite).

Modèles : lissage

Si on utilise une probabilité exponentielle

$$c(\Delta_d) = |2\Delta_d| \qquad p(\Delta_d) = e^{-|2\Delta_d|}$$





Hummm.... non convexe...

Performance

Taille du problème

n : taille de l'image en pixels
d : nombre de disparités

Taille du graphe

nombre de noeuds $v = nd$
nombre d'arcs $e = O(v) = nd$

Preflow-push lift-to-front (Goldberg & Tarjan)

$O(ve \log(v^2/e)) \rightarrow O(n^2 d^2 \log(nd))$

Meilleure complexité (Goldberg & Rao 97)

$O(e^{\frac{3}{2}} \log(v^2/e) \log U) \rightarrow O(n^{1.5} d^{1.5} \log(nd))$

Temps moyen observé

$O(n^{1.1} d^{1.3})$
Programmation dynamique : $O(nd)$

Sommaire

Résultats : parcomètre

Résultats : Visage (6 images, 384x288x100)

Impact du lissage

Malgré l'augmentation du degré de lissage, les discontinuités restent nettes.

Lissage : 0,1,2,8,16,32

- Résultat de la nature globale du flot maximum.
- Suggère qu'une pénalité *non robuste* peut préserver les discontinuités.

Le futur : Flot maximum incrémental

Idéalement on voudrait pouvoir modifier le graphe pendant le calcul.

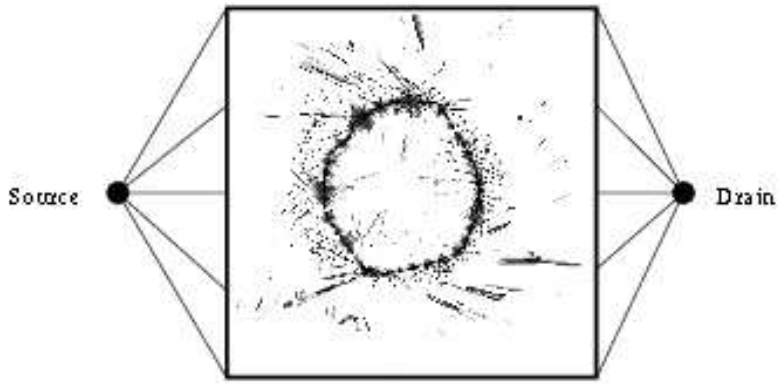
- Pyramides
- Modélisation des occlusions
- Lissage non convexe
- Voisinages variables
- Séquences vidéo
- ...

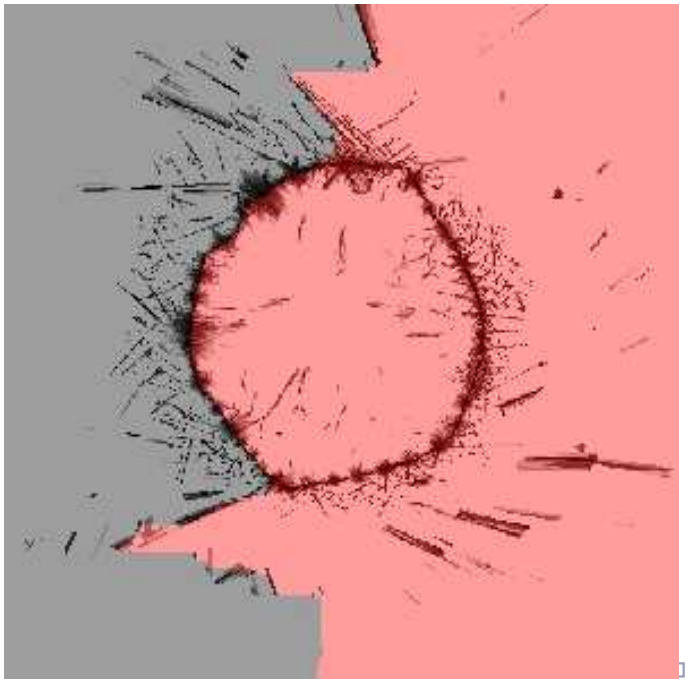
Une possibilité : l'anti-preflow

Sommaire

Flot maximum

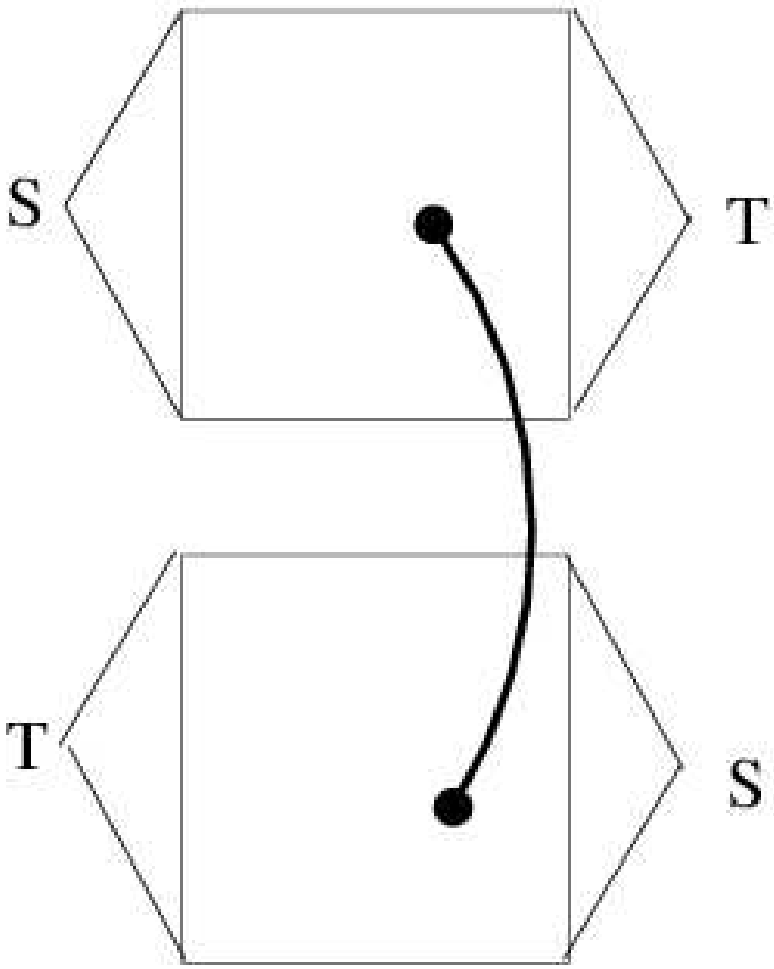
Choisir une direction (*avant* & *arrière*) et résoudre avec le flot maximum

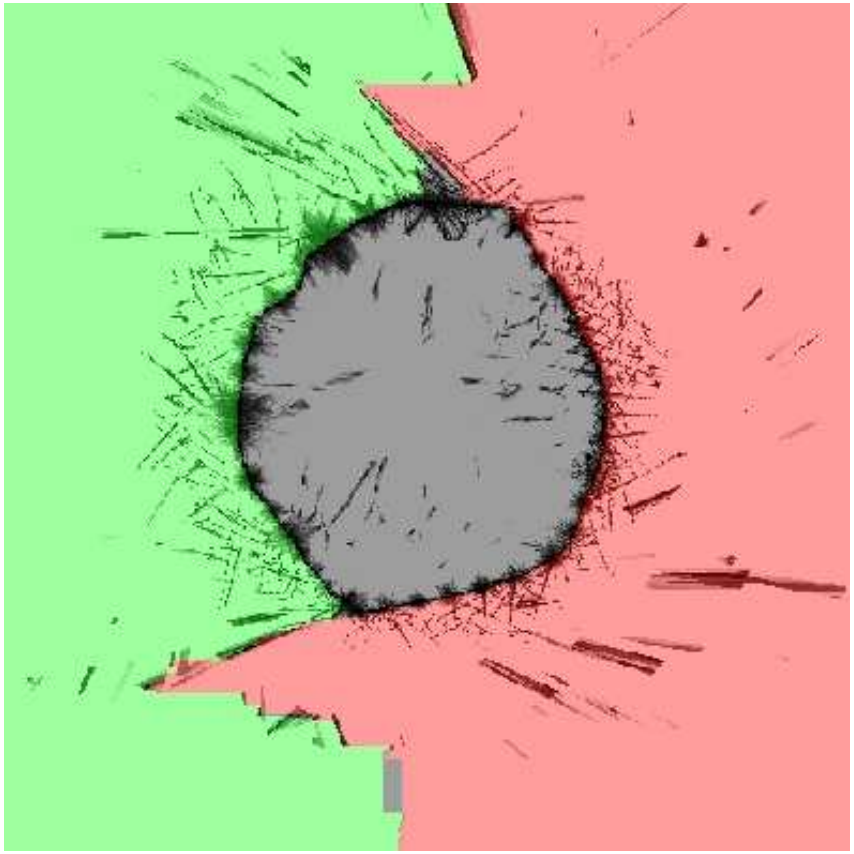




Flot maximum

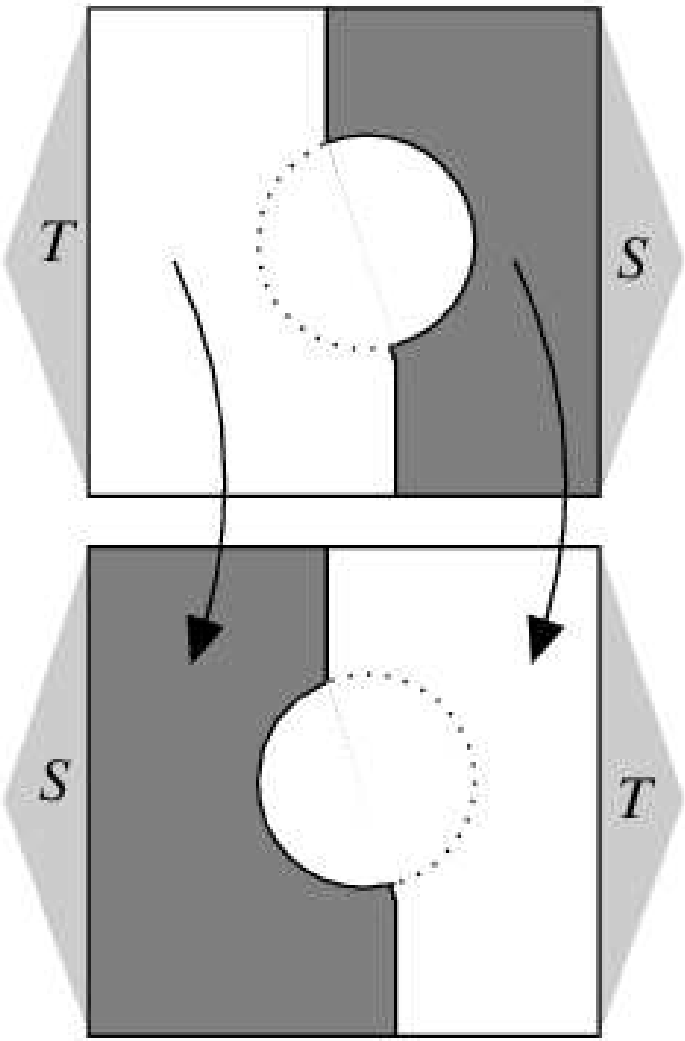
- Copier le problème, mais en échangeant la source et le drain
- Ajouter des arcs de lissage entre les deux copies du problème.

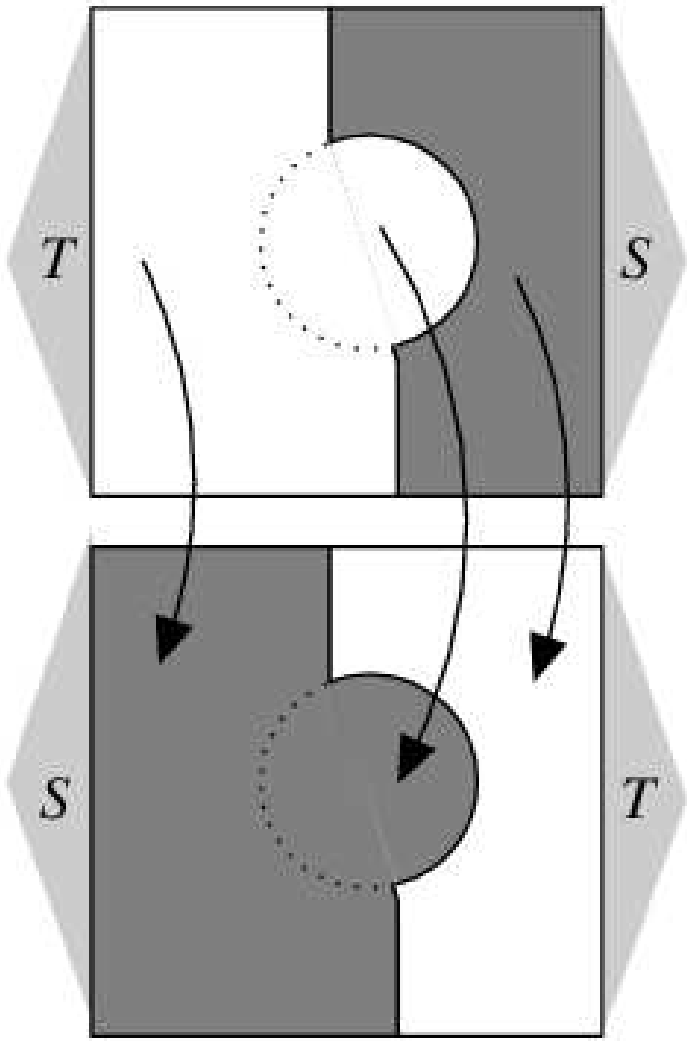




Flot maximum

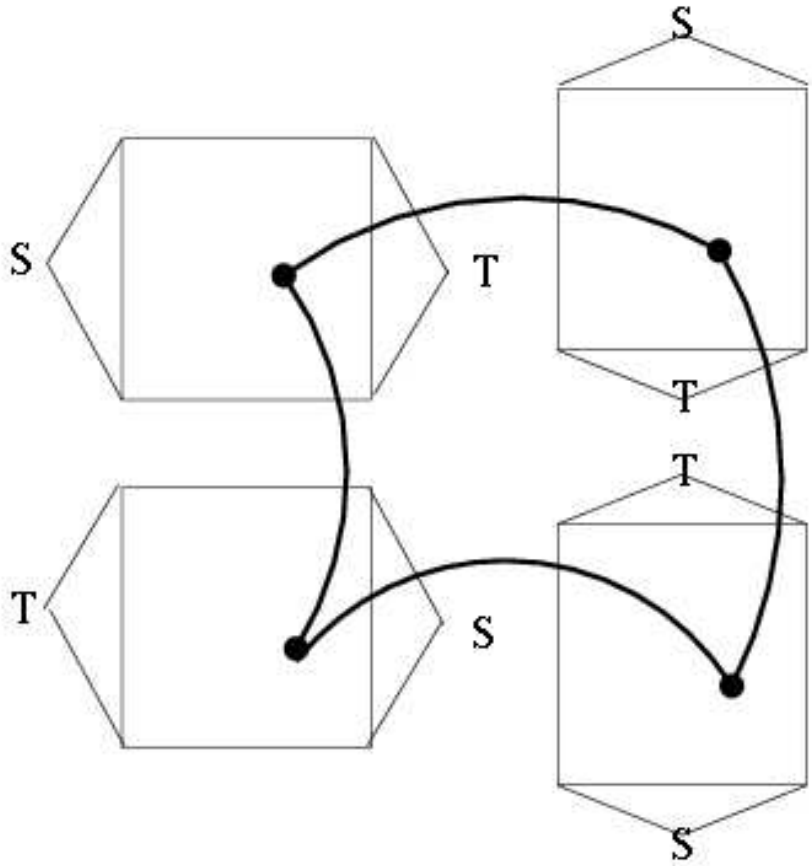
Pourquoi ça fonctionne ?

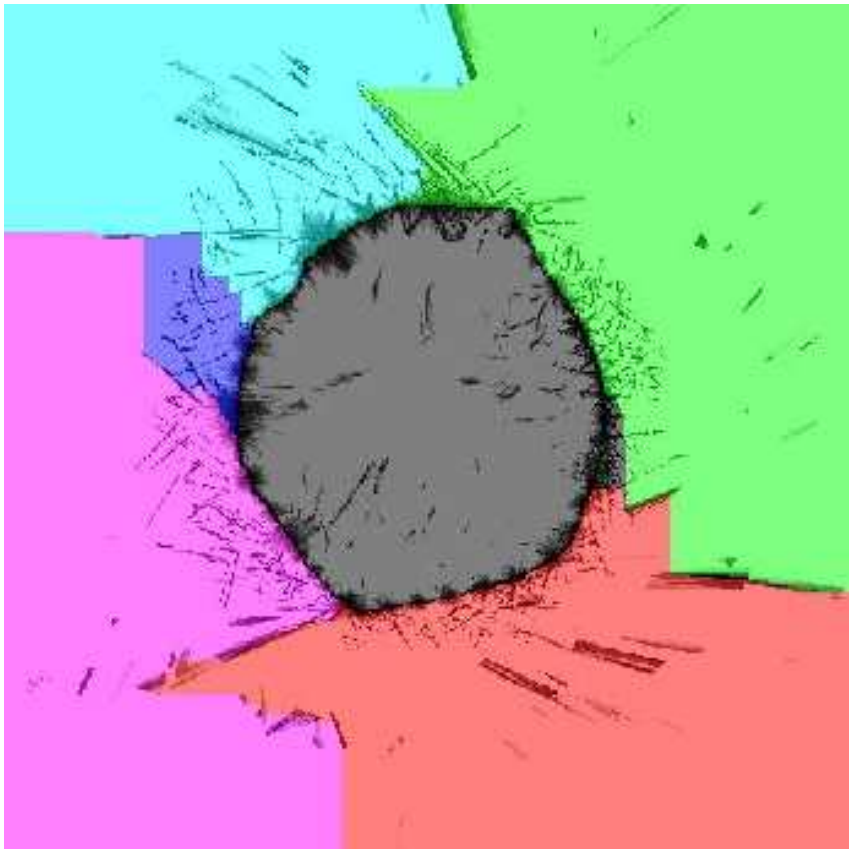




Flot maximum

Pourquoi se limiter à seulement deux directions ?



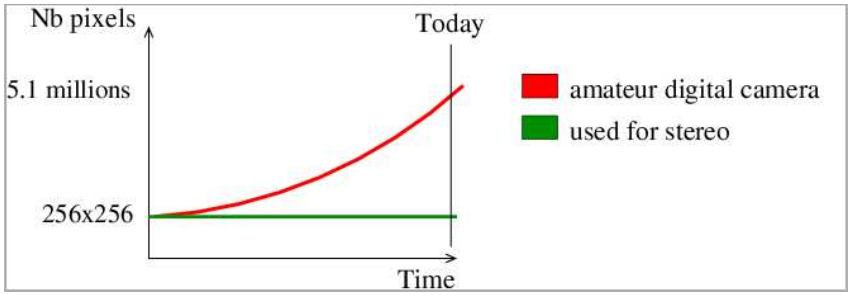


Inconvénients

- Coûteux en mémoire
- Degré de lissage dépendant de la taille de l'objet
- Limité aux objets à peu près convexes

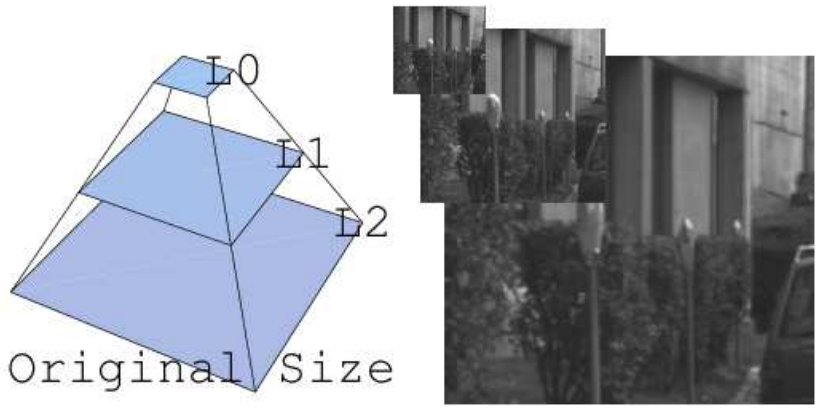
Sommaire

Des images de plus en plus grandes...



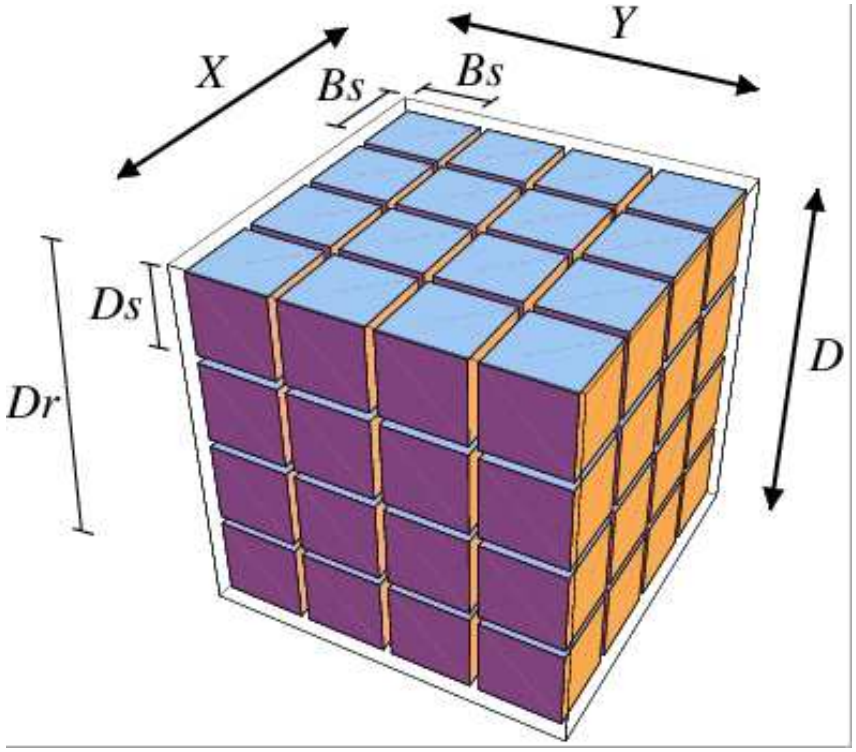
- Des approches pyramidales ont été introduites pour traiter les grandes images.
- Les pyramides souffrent de problèmes de propagation d'erreur.
- Comment minimiser cette propagation entre les niveaux ?

Pyramid



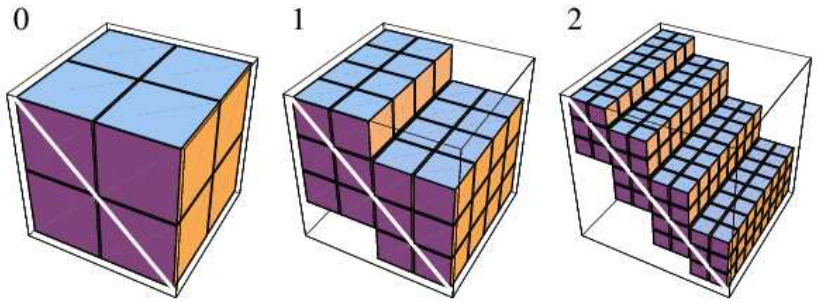
- $L0$ résolu en premier.
- $L1$ résolu ensuite, en utilisant le résultat de $L0$ comme initialisation.
- $L2$ résolu ensuite, en utilisant le résultat de $L1$ comme initialisation.
- On peut avoir plus de 3 niveaux.

Le volume de reconstruction



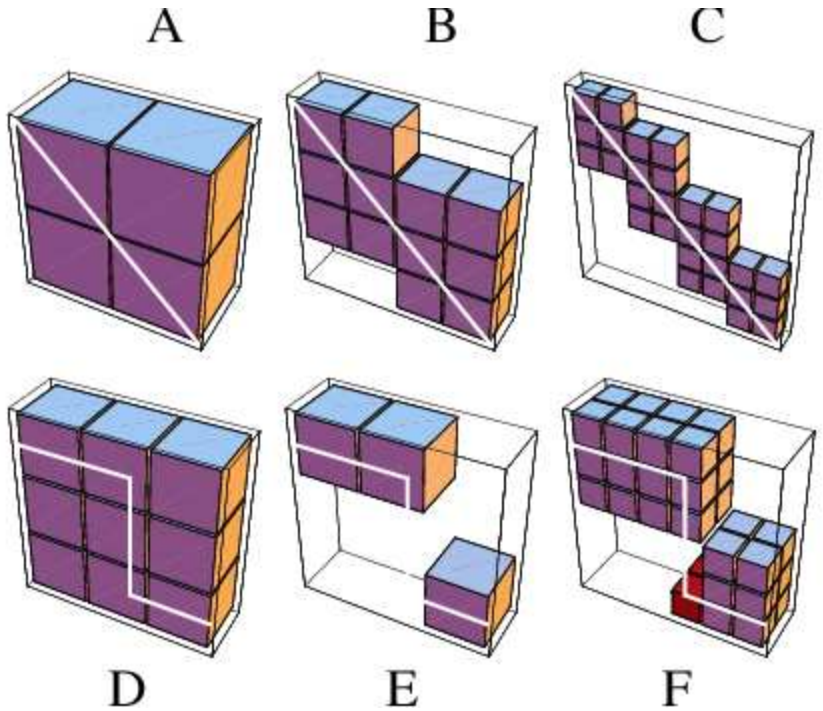
- Le plan (X, Y) est une carte de disparité avec des pixels de taille B_s.
- L'axe D est la disparité à assigner sur un intervale D_r et une taille D_s.

La pyramide classique



- Augmenter simultanément la résolution spatiale et des disparités tout en réduisant l'intervale de disparité.
- Niveau 0 couvre l'intervale complet des disparités (mode absolu).
- Niveaux 1,2 utilise un intervale relative à la solution précédente (mode relatif).

Classification des erreurs

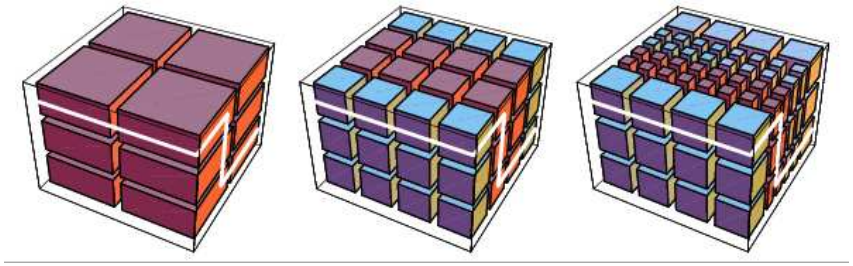


- Deux types d'erreur :

(A-C) Erreur de surface lisse (ok avec les pyramides classiques)

(D-F) Erreur de localisation des discontinuités (pas ok avec les pyramides classiques)

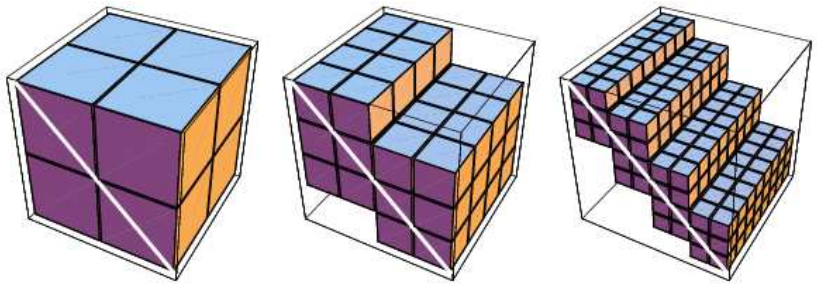
Erreur de localisation des discontinuités



Comment résoudre ?

- En augmentant la résolution spatiale aux grandes discontinuités.
  - Une discontinuité est grande si elle dépasse un seuil.
  - Augmenter la résolution jusqu'à une erreur de localisation de 1 pixel.
  - L'augmentation globale du volume de reconstruction est faible.

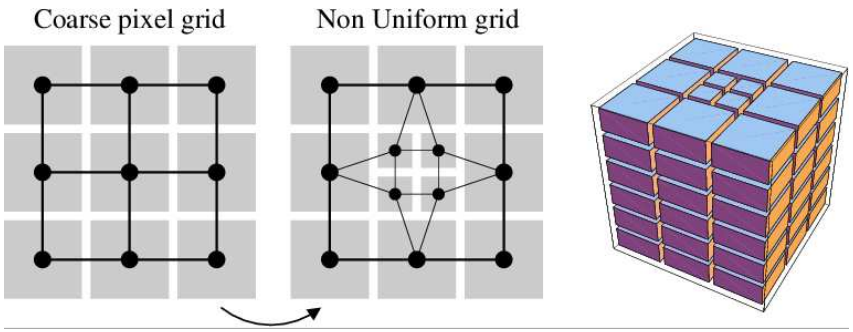
Erreur de surface lisse



Comment résoudre ?

- Augmenter la résolution spatiale et des disparités simultanément.
- Utiliser le mode relatif pour garder la taille du problème sous contrôle.

Formulation sous forme de graphe

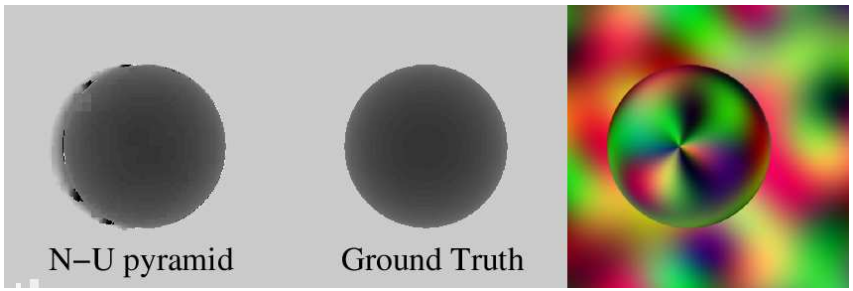


- Les arcs de lissage ont un coût proportionnel à la surface de contact.
- La régularité du graphe permet une implantation efficace.

Design & Limitations

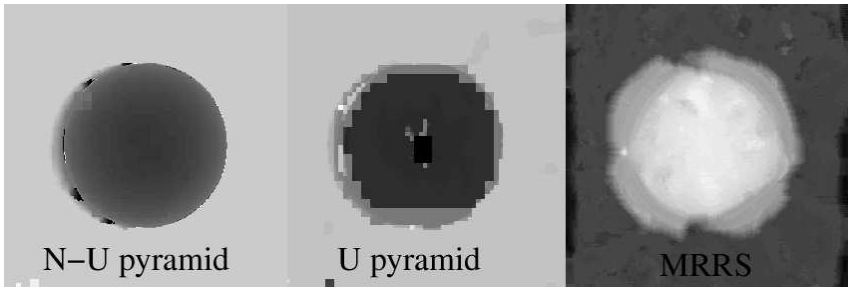
- Aucune modélisation des occlusions
- Aucun lissage adaptatif (variant spatialement)
- La détermination du seuil de grande discontinuité est délicate

Résultats : images synthétiques



- Résolution 512×512 et 64 disparités (17 millions de noeuds).
- Temps de calcul environ 10 sec sur un AMD Athlon 1.4Ghz.
- 200,000 opérations élémentaires (100 million pour le problème original).

Résultats : images synthétiques

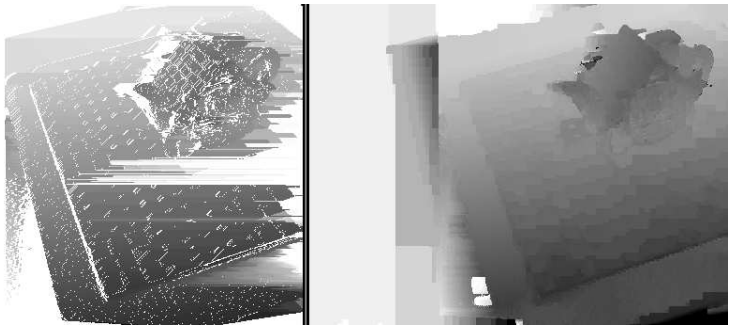


- Non-Uniforme and Uniforme ont le même temps d'éxécution.
- MRSS [C.Sun, 99] utilise une décomposition non-uniforme, mais ne modélise pas les grandes discontinuités.

Tea pot

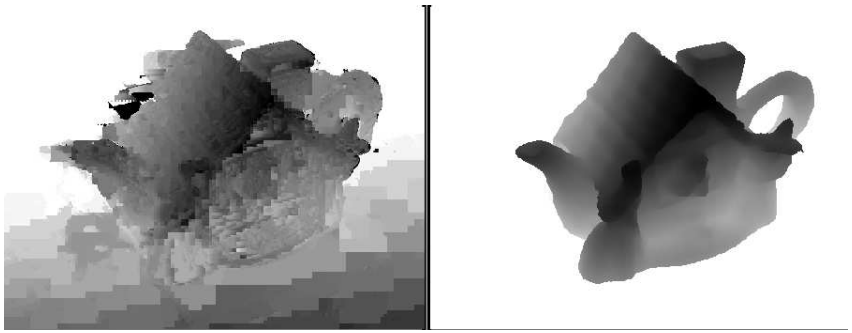
- Courtoisie of Jean-Yves Bouguet.
- Résolution 2048×1536, 400 disparités.
- Graphe d'une taille de plus d'un milliard de noeuds.
- Beaucoup d'occlusion, spécularités et surfaces à faible texture.

Tea pot



- L'algorithme MLMH [Cox] est utilisé pour la programmation dynamique.

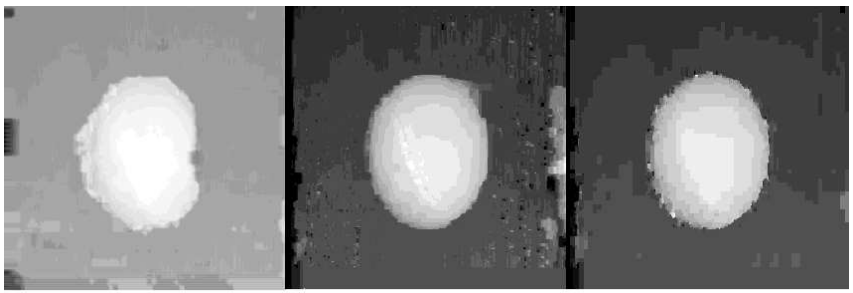
tea pot



- *Ground Truth* obtenue par lumière structurée (Jean-Yves Bouguet).

Baseball

baseball



- 9.4 million d'opérations élémentaires pour Max-flow.
- .57 million d'opérations élémentaires pour pyramide Non-Uniforme.

Conclusion

La stéréoscopie est une technique efficace de reconstruction 3D.

- Méthode passive (simple à mettre en oeuvre)
- Permet les caméras multiples
- Peut être résolue efficacement par flot maximum
- Peut être accélérée par une approche pyramidale
- Sensible à la texture des images
- Sensible aux occlusions