Shape from shading

Projet SIMNUM

Mots-clés : Traitement d'images, Problème inverse, Optimisation numérique.

1 Problème

Le "shape from shading" (sfs) est un problème de vision par ordinateur qui consiste à reconstruire le relief d'une scène à partir d'une seule photo de celle-ci. C'est le problème inverse de formation des images : comment retrouver la forme 3D initiale d'une surface à partir de la variation de l'intensité lumineuse de sa photo ?

Le sfs a été initié dans les années 60 par l'astronome Rindfleish, avec la parution d'un article sur la tomographie de la lune [4]. Cependant, ce n'est que dans les années 70 avec les travaux de Horn et Brooks que le sfs est devenu un domaine d'étude à part entière. Les auteurs se sont attachés à la fois à la modélisation physique du problème, à l'unicité de la solution et aux méthodes numériques de résolution. Leur livre fait encore office de référence [2].

Modélisation. Dans ce projet, on considère que le niveau de gris de l'image ne dépend que de la direction de la lumière incidente et de la normale à la surface (modèle d'ombrage lambertien) :

$$I(x,y) = \boldsymbol{L}.\boldsymbol{n}(x,y) \tag{1}$$

avec I intensité de l'ombrage (généralement $I \in [0;255]$), $\boldsymbol{L}$ la direction de la lumière incidente et $\boldsymbol{n}$ la normale **unitaire** sortante. Le problème principal pour la résolution de (1) est sa non linéarité qui provient simplement de la normalisation de $\boldsymbol{n}$.

En effet, si on représente notre surface par la donnée de $h : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ qui à (x,y) associe sa hauteur $z = h(x,y)$, alors les vecteurs (en supposant h assez régulière C^1) $(1,0,\frac{\partial h}{\partial x})^T$ et $(0,1,\frac{\partial h}{\partial y})^T$ forment une base du plan tangent à la surface en un point (x,y) (non singulier). On a alors une expression simple de la normale $\boldsymbol{n} = (1,0,\frac{\partial h}{\partial x})^T \times (0,1,\frac{\partial h}{\partial y})^T$. Le report de cette dernière équation dans (1), en supposant que la source est en $(0,0,1)$ (éclairage "frontal"), on trouve après normalisation :

$$I(x,y) = \frac{1}{\sqrt{1 + \frac{\partial h}{\partial x}(x,y)^2 + \frac{\partial h}{\partial y}(x,y)^2}} \tag{2}$$

Dans la suite, on adopte les notations courantes $p(x,y) = \frac{\partial h}{\partial x}(x,y)$ et $q(x,y) = \frac{\partial h}{\partial y}(x,y)$. L'intensité étant maximale lorsque $\boldsymbol{n}$ et $\boldsymbol{L}$ sont parallèles, on peut réécrire (2) sous la forme suivante, appelée équation eikonale :

$$E(x,y) = \frac{E_{Max}}{\sqrt{1 + p(x,y)^2 + q(x,y)^2}} \tag{3}$$

Résolution. Les méthodes d'optimisation (déterministe) pour le sfs consistent à minimiser des fonctionnelles représentant la vraisemblance entre les dérivées directionnelles $p = \frac{\partial h}{\partial x}$ et $q = \frac{\partial h}{\partial y}$ d'une surface pour l'équation de luminance. Une première erreur naturelle à minimiser est l'écart entre la forme définie par (p, q) et l'éikonale (3) soit :

$$e_1(p,q) = \iint_{(x,y)\in D} \left(E(x,y) - \frac{E_{Max}}{\sqrt{1 + p(x,y)^2 + q(x,y)^2}} \right)^2 dxdy,$$

où D est le domaine de reconstruction de l'image. Même si la minimisation seule de e_1 est possible, deux termes de pénalisation sont généralement ajoutés. Le premier permet d'assurer l'intégrabilité de la solution, hypothèse de base de notre modélisation (condition de Schwartz sur les dérivées) :

$$e_2(p,q) = \lambda_{int} \iint_{(x,y)\in D} \left(\frac{\partial p}{\partial y} - \frac{\partial q}{\partial x} \right)^2 dxdy.$$

Les solutions non lisses sont également pénalisées :

$$e_3(p,q) = \lambda_{csmo} \iint_{(x,y)\in D} \left(\frac{\partial p}{\partial x} \right)^2 + \left(\frac{\partial p}{\partial y} \right)^2 + \left(\frac{\partial q}{\partial x} \right)^2 + \left(\frac{\partial q}{\partial y} \right)^2 dxdy.$$

On préfère pénaliser notre problème d'optimisation plutôt que de le contraindre pour la simplicité de mise en œuvre de l'algorithme. On transforme donc le problème initial avec contraintes :

$$\begin{cases} \min\limits_{(p,q)} e_1(p,q), \\ \\ \dfrac{\partial p}{\partial y}(x,y) = \dfrac{\partial q}{\partial x}(x,y),\ (x,y) \in D \end{cases}$$

par le problème pénalisé suivant :

$$\min_{(p,q)} e_1(p,q) + \lambda_{int} e_2(p,q) + \lambda_{csmo} e_3(p,q). \tag{4}$$

Étant donnée la solution (p^*, q^*) de (4), on résout le problème :

$$\min_h \iint_D \left(\frac{\partial h}{\partial x} - p^*(x,y) \right)^2 dxdy + \iint_D \left(\frac{\partial h}{\partial y} - q^*(x,y) \right)^2 dxdy.$$

Fonctionnelles discrétisées. On discrétise les fonctionnelles précédentes par des différences finies décentrées à droite et on note ε_i la version discrète des e_i pour $i = 1, 2, 3$, et $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3$. Les fonctionnelles discrètes s'écrivent :

$$\begin{cases} \varepsilon_1 &= \delta^2 \displaystyle\sum_{(i,j)\in D} \left(E_{i,j} - \frac{E_{Max}}{\sqrt{1 + p_{i,j}^2 + q_{i,j}^2}} \right)^2, \\ \varepsilon_2 &= \lambda_{int} \displaystyle\sum_{(i,j)\in D_1} \left((p_{i,j+1} - p_{i,j}) - (q_{i+1,j} - q_{i,j}) \right)^2, \\ \varepsilon_3 &= \lambda_{csmo} \displaystyle\sum_{(i,j)\in D_1} (p_{i+1,j} - p_{i,j})^2 + (p_{i,j+1} - p_{i,j})^2 \\ & \qquad\qquad + (q_{i,j+1} - q_{i,j})^2 + (q_{i+1,j} - q_{i,j})^2, \end{cases} \tag{5}$$

où δ est le pas de discrétisation et $D_1 = \{(i,j)|(i+1,j+1) \in D\}$. La fonctionnelle discrétisée pour retrouver la hauteur est :

$$\min_{(h_{i,j})_{i,j}} \sum_{(i,j)\in D_1} (h_{i+1,j} - h_{i,j} - \delta p^*_{i,j})^2 + (h_{i,j+1} - h_{i,j} - \delta q^*_{i,j})^2, \tag{6}$$

Pour une image $m \times n$, on remarque que le point de coordonnées (m,n) n'est pas traité en approximant les dérivées par différences finies à droite. On lui affecte donc la moyenne de ses voisins pour le gradient $\frac{\partial \varepsilon_h}{\partial h_{m,n}}$. Pour un traitement exhaustif, il est possible de considérer selon le nombre de voisins d'un pixel donné les différences finies centrées, décentrées à droite ou à gauche. On est alors amené à minimiser une forme quadratique qui est convexe, par conséquent la condition initiale n'influe pas sur la solution finale de (6). En pratique, on peut partir d'un plan pour reconstruire h.

2 Travail demandé

L'enjeu de ce projet est tester la résolution des problèmes d'optimisation (5) et (6) liés au sfs en utilisant l'algorithme L-BFGS. On écrira également une classe *ImageFactory* permettant de créer des images tests à partir de surfaces 3D analytiques ou discrètes.

Rappel sur L-BFGS. On reprend l'algorithme de descente simple qui correspond au schéma itératif suivant :

$$\begin{cases} x_0 \text{ donné}, \\ x_{k+1} = x_k + \alpha_k d_k, \\ d_k = -\nabla(e(x_k)), \end{cases}$$

où e est la fonctionnelle d'erreur, x_k est la solution à l'étape k, *i.e.*, $x_k = (p_k, q_k)$ (resp. $x_k = h_k$) et α_k est la longueur du pas. Le meilleur pas est théoriquement solution de :

$$\min_{\alpha>0} \Phi(\alpha) = e(x_k + \alpha p_k). \tag{7}$$

Le pas optimal n'est calculable que pour certains cas simples, les formes quadratiques par exemple. En pratique, on réalise une recherche linéaire inexacte pour trouver un pas satisfaisant les conditions fortes de Wolfe :

$$\begin{cases} e(x_k + \alpha_k p_k) & \leq & e(x_k) + c_1 \alpha_k \nabla(e(x_k))^T p_k, \\ |\nabla(e(x_k + \alpha_k))^T| & \leq & c_2 |\nabla(e(x_k))^T p_k|, \end{cases}$$

avec $0 < c1 < 1/2 < c2 < 1$. La première condition assure une décroissance suffisante de la fonctionnelle d'erreur. La deuxième tente d'approcher les points stationnaires. Or ces conditions, munies de quelques hypothèses supplémentaires, assurent la convergence superlinéaire des méthodes de quasi-newton dont se déduit la méthode L-BFGS présentée ci-dessous. Cependant le gradient est une direction de descente "vraiment pauvre" qui est le plus souvent à éviter [1]. Son utilisation n'est réellement justifiée que lorsqu'on est proche de la solution, en fait, là où la variation du gradient est grande. Si ce n'est pas le cas alors cet algorithme est souvent très lent. Au lieu de considérer le modèle linéaire de e, on écrit son modèle quadratique autour du point x_k, soit :

$$m(x) = e(x) + \nabla e_k^T x + \frac{1}{2} x^T B_k x,$$

où B_k est une matrice symétrique définie positive qui est une approximation du "véritable" hessien de e en x_k. L'information supplémentaire du second ordre représente la courbure. On mettra en œuvre

la méthode L-BFGS, 'low memory' BFGS, inspirée de la méthode de type quasi-newton BFGS. On a choisit cette méthode pour sa simplicité. L'idée consiste à approximer l'inverse du hessien, noté H_k, à partir des m itérations précédentes. Considérons un pas de la méthode BFGS :

$$x_{k+1} = x_x - \alpha_k H_k \nabla(e_k),$$

où α_k est un pas satisfaisant les conditions fortes de Wolfe. La mise à jour de l'inverse du hessien se fait par :

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T,$$

avec,

$$\begin{cases} \rho_k = \dfrac{1}{y_k^T s_k}, \\ V_k = Id - \rho_k y_k s_k^T, \\ s_k = x_{k+1} - x_k, \\ y_k = \nabla e_{k+1} - \nabla e_k, \end{cases}$$

où Id est la matrice identité.

Ainsi, à l'étape k, l'approximation de l'inverse du hessien à partir des m étapes précédentes nous donne :

$$\begin{aligned} H_k \quad = \quad & (V_{k-1}^T ... V_{k-m}^T) H_k^0 (V_{k-m} ... V_{k-1}) \\ & + \rho_{k-m} (V_{k-1}^T ... V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1} ... V_{k-1}) \\ & + \rho_{k-m+1} (V_{k-1}^T ... V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2} ... V_{k-1}) \\ & ... \\ & + \rho_{k-1} s_{k-1} s_{k-1}^T. \end{aligned}$$

Pour calculer H_k, il suffit donc de mémoriser l'ensemble suivant :

$$\{s_i, y_i\}_{k-1,...,k-m}.$$

Finalement, la direction de descente est :

$$d_k = -H_k \nabla(e(x_k)),$$

avec H_k calculé suivant l'équation précédente (2). À l'inverse de la méthode BFGS, l'approximation initiale H_k^0 peut varier à chaque itération. On utilise une approximation diagonale de la forme $H_k^0 = \gamma Id$ avec Id matrice identité de $\mathbf{R}^{2n \times m}$. Le facteur γ est un terme de mise à l'échelle. Il permet d'approximer la taille réelle de l'inverse du hessien à partir de la dernière itération. La recherche linéaire a donc de forte chance d'accepter le premier pas essayé $\alpha = 1$.

Pour la mise en œuvre des algorithmes d'optimisation, on pourra compléter les classes vecteur et matrices vues en TP.

3 Fichiers fournis

3.1 Visualisation des surfaces reconstruites

On pourra utiliser le logiciel medit (éxécutable fourni) pour visualiser les maillages représentant la surface reconstruite. Ce logiciel est également téléchargeable sur `http://www.ann.jussieu.fr/~frey/software.html`. On donne en exemple les fichiers calotte.Href.mesh/face.mesh. Pour visualiser le maillage, taper la commande suivante dans un terminal : `medit calotte.Href.mesh`. Ce logiciel utilise le format mesh décrit ci-dessous. Pour zoomer/dé-zommer, il suffit de taper `z` ou `Z` dans la fenêtre graphique.

Le format mesh permet de décrire des maillages non structurés, on utilise dans le projet le cas des maillages 3D composés de quadrangles. Pour définir un maillage 3D, il suffit de définir l'entête composée des deux mot-clés `MeshVersionFormatted 2` et `Dimension 3` qu'on laisse inchangé. Ensuite, on définit la liste de sommets `Vertices` et la liste des quadrangles `Quadrilaterals`. La référence `ref` pour les quadrangles et sommets n'est pas utilisé ici, mettre 0. Le fichier doit se terminer par le mot-clé `End`.

```
MeshVersionFormatted 2
Dimension 3

# Set of mesh vertices (x,y,h(x,y),ref)
Vertices
581
0.1 1. 0
0.333 12.125 50.0 0
.........

# Set of mesh quads (v1,v2,v3,v4,ref)
Quadrilaterals
1162
1 28 521 10 0
23 45 77 35 0
.........
End
```

Une description complète de ce format est disponible dans le rapport de recherche de medit donnée dans l'archive.

Références

[1] J. F. Bonnans, J. Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization*. Springer Verlag, 2003.

[2] M. J. Brooks and B. K. P. Horn. Shape and source from shading. In B. K. P. Horn and M. J. Brooks, editors, *Shape from Shading*. MIT Press, Cambridge, MA, 1989.

[3] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[4] T. Rindfleisch. Photometric Method for Lunar Topography. *Photometric Engineering*, 32(2) :262–277, 1966.