Calcul de carte de disparité par programmation dynamique

Yonatan DELORO - Clément RIU

Projet MOPSI 2016-2017 encadré par M. Pascal MONASSE (Laboratoire IMAGINE)

École des Ponts ParisTech

Le besoin :

La perception de la profondeur repose sur la vision stéréo : deux photos légèrement décalées d'une même scène devraient permettre de trouver la profondeur d'un pixel.

FIGURE 1: Images stéréo d'une même scène.

Le modèle physique :

Pour ce faire, nous allons calculer la carte de disparité qui témoignera du déplacement de chaque objet d'une image à l'autre. La disparité est une grandeur qui se ramène au décalage d'un pixel d'une image par rapport au pixel de l'autre image correspondant au même élément de la scène (voir Figure 2). Si on appelle z la profondeur, et d la disparité, on a ainsi :

$$z \propto \frac{1}{d}$$

Il est toujours possible de se ramener à ce cas coplanaire par techniques de stéréorectification (une transformation qui ramène une image dans le plan optique de l'autre). Nous admettrons ce point pour nous concentrer sur la partie mathématique et algorithmique du problème (et non sur sa partie physique).

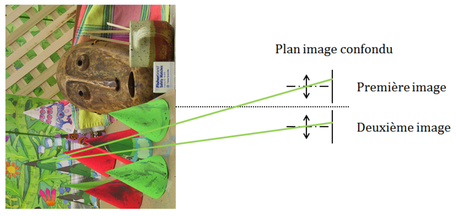

FIGURE 2: Schéma d'une prise de vue stéréo. Un même pixel va se retrouver décalé. En fonction de la profondeur du pixel, le décalage sera différent.

Hypothèses du modèle sur un exemple

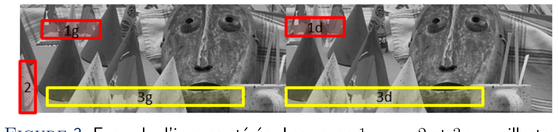

FIGURE 3: Exemple d'images stéréo. Les zones 1_d ou g, 2 et 3_d ou g illustrent les hypothèses faites dans la partie *Le modèle mathématique*.

Bibliographie

[1] Stephen S. Intille Aaron F. Bobick.
Large occlusion stereo.
The International Journal of Computer Vision, Juin 2005.

[2] Daniel Scharstein et Richard Szeliski.
A taxonomy and evaluation of dense two-frame stereo correspondence algorithms.
The International Journal of Computer Vision, Mai 2002.

[3] Litrico Arnaud.
L'appariement stéréoscopique.
Conservatoire National des Arts et Métiers. Rapport de stage., Octobre 2006.

[4] Jean-Hugh Thomas ENSIM – Université du Maine.
Imagerie - détection de contours.
Cours ressource numérique "Optique pour l'ingénieur", 2007.

[5] 2003 Middlebury Stereo Datasets.

Le modèle mathématique

Pour calculer la carte de disparité, il faut donc associer les pixels de l'image de droite (I_d) aux pixels de l'image de gauche (I_g) correspondant aux mêmes éléments de scène. Notre première hypothèse est : $\mathbf{H_1}$ **: deux pixels d'intensité suffisamment proche peuvent être associés.** La justification de cette hypothèse sera complétée par une autre hypothèse que nous ferons sur l'ordre des pixels (voir zone 3 dans la Figure 3).

De plus comme les deux images résultent d'un décalage latéral, on fait une seconde hypothèse : $\mathbf{H_2}$ **: les lignes de l'image sont indépendantes**, laquelle permettra notamment de paralléliser les calculs sur différents coeurs.

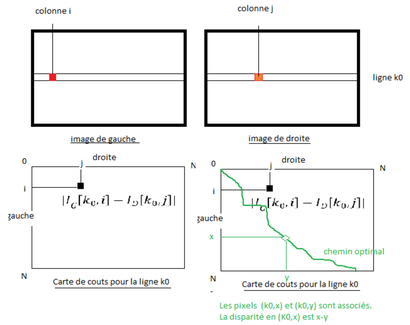

FIGURE 4: Illustration d'une carte des coûts et d'un chemin dans cette carte.

Associer au mieux les pixels des images revient donc à trouver, pour chaque ligne k_0, un «meilleur» chemin c^* dans la carte 2D des différences d'intensités entre les pixels de la ligne k_0 de I_g et ceux de la ligne k_0 de I_d (on l'appellera la carte des coûts). Alors, la disparité du pixel (k_0, i) de I_g correspondra à :

$$d(k_0, i) = |i - c^*((k_0, i))|$$

où $c^*((k_0, i))$ est la colonne du pixel de I_d associé au pixel (k_0, i) de I_g.

Il faut donc définir ce qu'est un « meilleur chemin ». Pour cela on introduit la fonction objectif suivante, qu'on souhaitera minimiser sur tous les chemins d'un certain ensemble $\mathcal{C}$ qui reste à définir :

$$\min_{c \in \mathcal{C}} \sum_{(i,j) \in c} p_{k_0}(i, j) \qquad \text{où } p_{k_0}(i, j) = |I_g(k_0, i) - I_d(k_0, j)|$$

Enfin, pour restreindre $\mathcal{C}$, on introduit l'hypothèse $\mathbf{H_3}$ **: le chemin qui associe le mieux les pixels est croissant.** Cette hypothèse correspond physiquement à un ordre des pixels conservé entre les deux prises de vues.

Recherche de la solution par programmation dynamique

La programmation dynamique rajoute une contrainte de « temps » qui, avec $\mathbf{H_3}$ permet de reformuler $\mathbf{H_1}$ plus précisément : on associe à un certain pixel de I_g le pixel de I_d d'intensité la plus proche parmi ceux n'ayant pas déjà été associés.
On doit donc chercher le chemin croissant optimal de (i_0, j_0) à (i_N, j_N) sur tous les $i_0,\ j_0,\ i_N,\ j_N$ où $i_N = N$ ou $j_N = N$.
L'équation de programmation dynamique utilisée s'écrit comme suit :

$$b(i, j) = p_{k_0}(i, j) + \min \begin{cases} b(i-1, j-1) \\ b(i-1, j) \\ b(i, j-1) \end{cases}$$

où $b(i, j)$ est le coût pour associer les pixels de I_g en colonnes $[1, i]$ et ceux de I_d en colonnes $[1, j]$.

Premier résultat

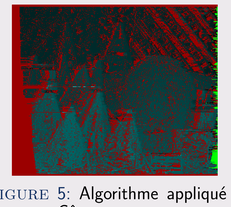

FIGURE 5: Algorithme appliqué à l'image « Cône ».

Plus un pixel est sombre, plus il est profond. Les pixels rouges ont été «sautés» (considérés comme **occultés**) et les pixels verts sont considérés comme immobiles d'une image à l'autre.

On observe des défauts dans l'image notamment dans les zones 1 (voir Figure 3) où $\mathbf{H_3}$ n'est pas respectée, et en zone 2 là où les pixels n'apparaissent que dans une image (ce problème d'occultation survenant à plusieurs endroits).

Améliorations : détection des occultations et points de contrôle

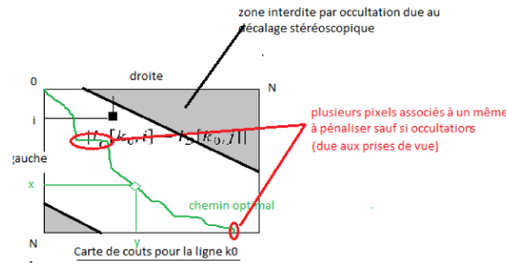

FIGURE 6: Deux types d'occultations dans une paire d'images stéréo

1. Utilisation des bords : L'association des pixels va donc être modifiée de manière à prendre en compte les zones occultées dans chaque image (justification en Figure 6). Celles-ci ont plus de chance d'apparaître sur les bords d'un objet, que l'on va ainsi détecter typiquement par un calcul de « gradient ». Plus précisément, on applique sur l'image l'opérateur de Sobel, qui en donne une approximation grâce à un voisinage de taille $(3, 3)$ autour de chaque point :

$$G_x = \begin{matrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{matrix} \quad \text{et} \quad G_y = \begin{matrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$$

A cause de la difficulté à estimer la « bonne » pénalité d'une association multiple en fonction du gradient en un point, on préfère fixer simplement un seuil discriminant bien les bords de l'image. Ainsi pour un pixel de I_g (respectivement I_d) n'appartenant pas à un bord noté $\mathcal{S}(I_g)$ (respectivement $\mathcal{S}(I_d)$) défini par le seuil, on interdit purement et simplement l'association multiple de ce pixel à plusieurs pixels de I_d (respectivement I_g), en assignant un coût infini pour un déplacement vertical (respectivement horizontal) dans la carte des coûts.

Une méthode de test

Afin de déterminer si notre manière de distinguer les occultations pour mieux associer les pixels est sensible ou non au bruit, nous avons créé des images de synthèse que nous avons perturbées artificiellement avec un bruit gaussien. Nous avons ensuite augmenté progressivement la valeur de l'amplitude du bruit.
Ci-dessous, la paire d'images de synthèse « stéréo », et la carte de disparité pour un bruit gaussien $\mu = 255/2,\ \sigma = \sqrt{255/2}$ d'amplitude $\alpha = 0.01$. En pratique, on arrive encore à faire la différence entre les profondeurs en poussant α jusqu'à l'écart-type de l'image non bruitée. Cela nous permet donc de penser que la méthode proposée sera resistante au bruit.

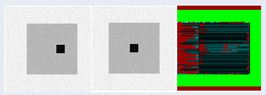

FIGURE 7: Test sur une paire d'images de synthèse

Notre nouvelle équation de programmation dynamique est :

$$b(i, j) = p_{k_0}(i, j) + \min \begin{cases} b(i-1, j-1) \\ b(i-1, j) + \infty.\mathbf{1}_{I_d(k_0,j) \notin \mathcal{S}(I_d)} \\ b(i, j-1) + \infty.\mathbf{1}_{I_g(k_0,i) \notin \mathcal{S}(I_g)} \end{cases}$$

où l'on pose $\infty.0 = 0$.

2. Utilisation de points de contrôle : Une autre amélioration que nous avons tenté de développer est l'utilisation de « points de contrôle » (GCP) qui correspondent à des associations quasi-certaines de pixels (au moins une par lot de pixels de I_d dont les voisinages ont mêmes intensités que celui d'un pixel de I_g). Le meilleur chemin sera ainsi contraint à passer par les *GCP*, dans l'espoir que ceux-ci le guident vers les associations correctes dans les cas où il y a plus de choix.

Conclusion

Perspectives :

- Ajouter de la cohérence entre les lignes (Hirschmueller 2005, *semi-global matching*), lever l'hypothèse de croissance H_3 quand nécessaire (Dhond-Aggarwal 1995, traitement des *narrow-occluding objects*)
- Extraire d'autres caractéristiques intéressantes de l'image pour améliorer la qualité des associations ou réduire leur espace de recherche, en exploitant au maximum la connaissance de notre propre vision stéréoscopique. (En particulier, les deux améliorations présentées ici sont consistantes avec la psychologie expérimentale : importance des bords dans la perception des occultations et de notre représentation spatiale, aide de certains points de repère pour positionner les points plus ambigus).
- Se tourner vers les techniques de Machine Learning qui semblent montrer de très bonnes performances sur certaines bases de données (voir [5]).

Quelques champs d'applications possibles :
C'est l'application visée qui doit décider des critères à prioriser (temps de calcul, nombre de pixels correctement prédits, détection précise des zones d'occultation) et donc des méthodes à choisir :

- Véhicules intelligents, pour la détection d'obstacles (lots d'images potentiellement plus difficiles à traiter à cause des conditions d'éclairage par exemple...).
- Robotique d'aide à la personne, par exemple pour la préhension d'objets (images de la vie de tous les jours possédant beaucoup d'occultations !)
- Archéologie, histoire, urbanisme, pour la reconstruction 3D d'environnements à partir de photos, peintures, vues aériennes, etc...

Résultat final

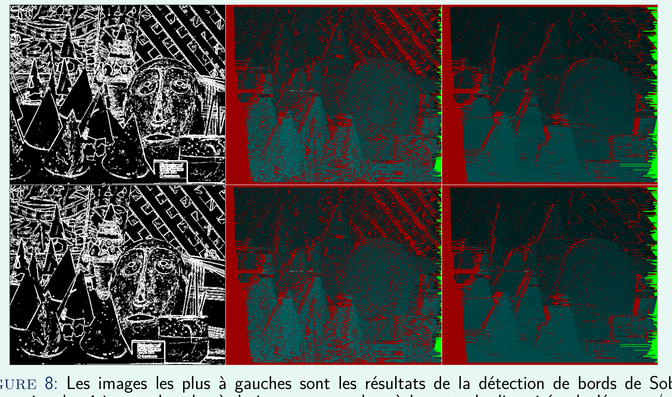

FIGURE 8: Les images les plus à gauches sont les résultats de la détection de bords de Sobel. La matrice des 4 images les plus à droites correspondent à la carte de disparité calculée avec/sans (droite/gauche) l'amélioration des bords, et avec/sans (bas/haut) l'implémentation des *GCP*.

On peut voir que l'utilisation des bords améliore grandement la qualité du résultat. On remarque premièrement que la quantité de pixels non associés a bien diminué. De plus, leur prédiction semble cohérente : les pixels de même profondeur appartiennent bien à un même plan. On note en particulier la meilleure interprétation du quadrillage en arrière-plan. On réduit donc la quantité de sauts non justifiés, et la solution est améliorée.

L'implémentation des *GCP* ne semble pas donner de résultat probant. Une hypothèse que l'on peut avancer pour expliquer ce résultat est que le chemin minimisant optimal passait déjà par les *GCP*. Il faut peut-être chercher des points plus subtils. Les *GCP* permettent néanmoins de réduire l'espace des chemins à explorer (et donc le temps de calcul) à condition d'effectuer un pré-processing sur l'image pour les trouver intelligemment.