ROB 313 - IA 323
Vision pour la Robotique
Corrigé du Contrôle des Connaissances

28 Janvier 2022

Barème : **2 points par question (Notation sur 22).**

1 Caméras RGB-d par lumière structurée

Quelles propriétés doit respecter la mire projetée dans une scène par le projecteur d'une caméra RGB-d active à lumière structurée ? Donner deux exemples de mires possibles. Quelles sont les difficultés et limites de ces caméras ?

La mire projetée doit être telle qu'elle permet toujours d'associer un angle à chaque point de l'image à partir des valeurs de son voisinage. Par conséquent, au moins une des dimensions de chaque point de la mire doit être codée de façon unique par le voisinage du point.

On peut obtenir un tel codage avec des mires 1d formées par des bandes de couleur où chaque couleur forme un symbole et la séquence de bandes forme une suite de De Bruijn associée à ces symboles. Une autre solution consiste à créer des mires pseudo-aléatoires à partir d'un motif aléatoire réarrangé associé à un algorithme de recherche.

La première difficulté est d'ordre combinatoire : il s'agit de trouver un compromis entre la résolution de la mire, la taille du voisinage examiné, et la précision de la profondeur estimée. La principale limitation est technologique : la projection devant avoir une grande profondeur de champ (le voisinage projeté doit apparaître net quelle que soit la profondeur), on utilise de faibles ouvertures, ce qui limite la luminosité des mires projetées et empêche en général leur utilisation en extérieur ou lorsque la lumière ambiante est forte.

2 Contrainte épipolaire

Qu'est-ce que la contrainte épipolaire qui lient 2 images de la même scène perçue depuis deux points de vue différents ? Comment s'exprime-t-elle mathématiquement ? Lorsque les deux points de vue sont fournis par la même caméra qui s'est déplacée entre deux instants t_1 et t_2, comment peut-on utiliser cette contrainte pour détecter les objets qui se sont déplacés par rapport à la scène entre t_1 et t_2 ?

La contrainte épipolaire est une contrainte géométrique qui exprime le fait qu'un point M qui se projette dans 2 plans focaux distincts π_1 et π_2 forme, avec ses projections respectives m_1 et m_2 dans chaque plan focal, un plan contenant aussi les 2 centres optiques associés Ω_1 et Ω_2. Ce plan coupant donc π_1 (resp. π_2) en une droite d_1 (resp. d_2), il suffit, connaissant m_1, de rechercher son homologue m_2 le long de la droite d_2.

Cette contrainte s'exprime mathématiquement par l'intermédiaire de la matrice fondamentale F, qui représente à la fois la matrice de calibration des 2 caméras et leur position relative. On a, en coordonnées homogènes : $m_2^T F m_1 = 0$. On a également $d_2 = F m_1$ et $d_1 = m_2^T F$.

Dans le cas d'une même caméra en déplacement, cette contrainte n'est bien sûr valable que pour les points qui n'ont pas bougé dans la scène entre t_1 et t_2. Pour détecter les objets mobiles, on calculera le déplacement apparent de tous les points (i.e. flot optique, sans utiliser la contrainte épipolaire !). Puis pour chaque paire appariée (m_1, m_2), on pourra faire l'hypothèse que le point 3d M correspondant appartient à un objet qui a bougé lorque la distance entre m_2 et "sa" droite épipolaire $\delta(m_2, F m_1)$ est suffisament grande.

3 Détection 3d monoculaire de personnes

Soit une caméra (fixe) observant le grand hall de l'ENSTA. Supposons qu'on applique une détection de personnes à chaque image qui fournit les coordonnées 2d de la boîte englobant chaque personne qui se trouve dans le grand hall. Comment peut-on estimer la position 3d de la personne, c'est-à-dire sa distance à la caméra, et les dimensions réelles (i.e. en cm) de la boîte englobante ?

Supposons pour simplifier que le plan de la caméra est orthogonal à la surface du grand hall. Soit H_c la hauteur de la caméra par rapport au sol (distance du centre optique Ω au sol), et f sa distance focale. Si l'on fait l'hypothèse (raisonnable...) que les pieds des personnes détectées touchent le sol, on peut déduire la distance D de la personne à la caméra à partir de la coordonnée verticale dans l'image y_p de la partie inférieure de leur boîte englobante par : $D = f\left(1 + \frac{y_p}{H_c}\right)$. De même la vraie hauteur H de la personne pourra se déduire de la hauteur h de la boîte englobante dans l'image par : $H = h\left(\frac{D}{f} - 1\right)$. Idem pour la vraie largeur L.

4 Bag-of-features

Un robot ménager muni d'une caméra couleur a été entraîné pour reconnaître la pièce dans laquelle il se trouve à partir de son apparence visuelle, en utilisant une approche de type bag-of-features. Décrivez une procédure possible d'apprentissage pour le robot, ainsi qu'une méthode de prédiction.

Il y a beaucoup de solutions possibles. Une approche standard consiste à construire une base d'entraînement à partir d'un grand nombre d'images des différentes pièces, en annotant simplement chaque image par la pièce (classe) correspondante. Ensuite dans une phase d'apprentissage non supervisée, on extrait des points d'intérêt de l'ensemble des images d'entraînement, en calculant pour chaque point un descripteur, puis on applique un algorithme de regroupement (clustering, comme K-Means) à l'ensemble des descripteurs, de façon à construire un dictionnaire (codebook) de caractéristiques (features). Dans un deuxième temps (phase d'apprentissage supervisée) on extrait pour chaque image un descripteur global sous la forme d'un histogramme de mots du dictionnaire construit précédemment, puis on entraîne un modèle de classification (comme SVM ou perceptron multi-couches) sur l'ensemble des histogrammes construits, supervisé par la classe (pièce).

La méthode de prédiction consisterait dans ce cas pour le robot à acquérir des images de la pièce dans laquelle il se trouve, et pour chaque image, extraire les points d'intérêt et leurs descripteurs, associer à chaque descripteur son mot du dictionnaire associé (plus proche voisin), pour calculer un histogramme de mots (bag-of-features), qu'on soumet ensuite au classifieur supervisé pour prédire la classe. La répétition sur plusieurs images permet de rendre la reconnaissance plus robuste.

5 Homographies

Qu'est-ce qu'une homographie ? Dans quel(s) conditions un couple d'images est-il relié par une homographie ? Peut-on estimer, selon le cas, la profondeur par appariement de points ?

L'homographie est la transformation projective la plus générale qui relie deux points m_1 et m_2 projetés sur 2 plans depuis le même point 3d M. Elle s'exprime en géométrie projective par une matrice 3×3 ayant 8 degrés de liberté.

Dans le cas général de deux images de la même scène prises depuis des points de vue différents, chaque appariement (m_1, m_2) aura sa propre homographie H telle que $m_2 = Hm_1$. Cependant l'homographie peut être la même pour tous les appariements de l'image dans deux cas particuliers : (1) Tous les points 3d M sont coplanaires. Dans ce cas, il est possible d'estimer la profondeur des points à partir de la disparité des points appariés et du déplacement de la caméra. (2) Le mouvement de la caméra est une rotation pure autour du centre optique. Dans ce cas on ne peut pas estimer la profondeur car les points conservant leur droite de projection entre les deux positions, la triangulation n'est pas possible.

6 Prédiction monoculaire auto-supervisée de cartes de profondeur

Comment se calcule la fonction de perte/coût photométrique permettant d'entraîner des réseaux profonds à prédire des cartes de profondeur de façon auto-supervisée ? Quelles sont les contraintes et difficultés potentielles ?

La fonction de coût photométrique est calculée à partir d'un couple d'images (I_1, I_2) capturées à deux instants t_1 et t_2, et de l'estimation de la carte de profondeur Z_1 au temps t_1, et de l'estimation des rotation et translation $(R, \mathbf{t})$ de la caméra (de matrice de calibration K) entre t_1 et t_2. Elle est fondée sur la différence entre l'image I_2 observée et l'image $\hat{I}_2$ prédite à partir de I_1, Z_1 et $(R, \mathbf{t})$. En effet chaque point m_1 de I_1 se rétro-projette sur le point 3d M par : $M = Z_1(m_1)K^{-1}\tilde{m}_1$, lequel se re-projette sur I_2 par $\tilde{m}_2 = (K|\mathbf{0})\,(R|\mathbf{t})\,M$. On a donc $\hat{I}_2(m_2) = I_1(m_1)$.

Les difficultés sont multiples. Outre les erreurs d'estimation de Z et $(R, \mathbf{t})$, m_2 obtenu par les calculs précédents n'est pas entier, il faut donc interpoler $\hat{I}_2$. D'autre part l'information contenue dans $\hat{I}_2$ est partielle car réduite à l'intersection des vues entre I_1 et I_2. Elle est en outre perturbée par les occultations (objets visibles par I_1 mais pas par I_2), les désoccultations (l'inverse), ainsi que les objets à diffusion non lambertienne (comme les reflets, qui rayonnent différemment en direction de I_1 et de I_2).

7 Réseau de neurones : entraînement

Vous entraînez un réseau de neurones (CNN) avec 100 couches, sur une tâche de classification binaire, en utilisant une activation sigmoïde dans la couche finale, et tangente hyperbolique (tanh) pour toutes les autres couches. Vous remarquez que les poids du réseau de neurones cessent de se mettre à jour après la première epoch d'entraînement, même si votre réseau n'a pas encore convergé. Une analyse plus approfondie révèle que les gradients de ces couches sont complètement ou presque complètement nuls très tôt dans la formation. Laquelle des corrections suivantes pourrait vous aider ? (Notez également que votre fonction de perte/coût ne diverge pas).

1. Augmenter la taille de votre ensemble d'entraînement

2. **Changer les fonctions d'activation tanh pour des ReLU**
3. **Ajouter de la batch normalisation avant chaque activation**
4. **Augmenter le learning rate**

Réponses possibles : (2) et (4). C'est un problème de vanishing gradient.

8 Réseau de neurones : fonctions d'activation

Parmi les fonctions suivantes, lesquelles considérez-vous comme des fonctions d'activation valides (non-linéarités élément par élément) pour entraîner un réseau de neurones dans la pratique ?

1. $f(x) = -\min(2, x)$
2. $f(x) = 0.9x + 1$
3. $f(x) = \begin{cases} min(x, .1x) \text{ if } x \geq 0 \\ min(x, .1x) \text{ if } x < 0 \end{cases}$
4. $f(x) = \begin{cases} max(x, .1x) \text{ if } x \geq 0 \\ min(x, .1x) \text{ if } x < 0 \end{cases}$

Réponses possibles : (1) et (3), qui sont à la fois concaves et non linéaires. Les autres sont des fonctions linéaires, donc inutiles comme fonctions d'activation.

9 Réseau de neurones : Tracking

Supposons que vous souhaitiez entraîner un réseau de neurones (CNN) pour suivre les particules en biologie, nous considérons que vous disposez d'un jeu de données composé d'images de cellule et de leur emplacement correspondant. Quel type d'architecture d'apprentissage en profondeur pouvez-vous utiliser pour suivre la particule :

1. **faster RCNN**
2. **DeepLab**
3. **Unet**
4. **Yolo**

Réponses possibles : (1) et (4). Tracking implique détection d'instances.

10 Réseau de neurones : cartes de profondeur

Supposons que vous souhaitiez un réseau de neurones pour prédire des cartes de profondeur (3D), l'entrée étant composée d'une seule image. Ce type de CNN prédit une profondeur pour chaque pixel. Quel type d'architecture de CNN pouvez-vous utiliser après une modification mineure pour prédire la profondeur monoculaire :

1. **faster RCNN**
2. **DeepLab**
3. **Unet**
4. **Yolo**

Réponses possibles : (2) et (3). Il faut une prédiction par pixel.

11 Réseau de neurones : cartes de profondeur

Supposons que vous souhaitiez entraîner un réseau de neurones à prédire des cartes de profondeur (3D). Ce type de CNN prédit une profondeur pour chaque pixel. Notons $y \in \mathbb{R}^K$ la vérité terrain et $\hat{y} \in \mathbb{R}^K$ la prédiction du CNN qui sont toutes les deux des cartes de profondeur de taille K. Quel type de fonction de perte/coût pouvez-vous utiliser pour entraîner votre réseau :

1. **Cross entropie :** $H(y, \hat{y}) = -\sum_{i=1}^{K} y_i \log(\hat{y}_i)$
2. **Mean Squared Error :** $MSE(y, \hat{y}) = \frac{1}{K}\sum_{i=1}^{K}(y_i - \hat{y}_i)^2$
3. **Mean Absolute Error :** $MAE(y, \hat{y}) = \frac{1}{K}\sum_{i=1}^{K}|y_i - \hat{y}_i|$
4. **Divergence de Kullback Leibler :** $D_{\mathbf{KL}}(y, \hat{y}) = \sum_{i=1}^{K} y_i \log(\frac{y_i}{\hat{y}_i})$

Réponses possibles : (2) et (3). La somme est faite sur le support spatial, (1) et (4) sont sommés dans l'espace des valeurs car ils comparent des distributions.