INF6803

TRAITEMENT VIDÉO ET APPLICATIONS

H2017 – Travail Pratique #1

Segmentation vidéo par soustraction d'arrière-plan

Objectifs :

- Permettre à l'étudiant de se (re)familiariser à la manipulation de données et au traitement d'images/vidéos en C++ à l'aide d'OpenCV, ou bien avec le logiciel Matlab
- Permettre à l'étudiant de se familiariser avec les algorithmes de segmentation vidéo par modélisation d'arrière-plan dans une séquence vidéo capturée par une caméra fixe (« background subtraction »)

Remise :

- Tout votre code source (.m pour Matlab, .hpp/cpp pour C++, mais pas les deux!)
- Un rapport (**format .pdf, 3 ou 4 pages**) contenant :
  - un survol bref de votre travail : présentation ViBe dans vos mots + détails implémentation
  - un tableau de vos métriques de performance pour toutes les séquences, avec discussion
- Avant le 13 février 2017, 14h00, sur Moodle – **aucun retard accepté**

Références :

- Voir les notes de cours sur Moodle (Chapitre 2)
- https://doi.org/10.1109/TIP.2010.2101613

Autres directives :

- Pensez à commenter l'ensemble de votre démarche directement dans votre code! Sinon, difficile d'attribuer des points lorsque ça ne fonctionne pas...
- Les TDs s'effectuent seul, ou en équipe de deux – ne remettez par contre qu'une version du rapport/code!

Présentation

L'objectif de ce travail pratique est d'implémenter et d'évaluer la performance d'une méthode simple de segmentation vidéo par modélisation d'arrière-plan (« background subtraction »), ViBe. Vous pouvez travailler en C++ à l'aide de la librairie OpenCV, ou en Matlab. Les deux environnements ont leurs avantages et inconvénients; c'est à vous de choisir lequel utiliser en fonction de vos compétences! Notez que dans les deux cas, quelques fonctions utilitaires vous sont déjà fournies sur Moodle pour vous aider.

Pour les détails complets sur la méthode demandée, lisez l'article de journal qui l'a introduite [1], ou bien allez revoir les notes de cours (Chapitre 2, p.27-31). La boucle principale de traitement de ViBe (pour chaque pixel) peut être séparée en trois parties :

1. Recherche de correspondances parmi les échantillons du modèle
2. Classification du pixel en fonction du nombre de correspondances trouvées
3. Mise à jour (aléatoire) d'échantillons d'arrière-plan

Assurez-vous de bien comprendre les conditions reliées à la mise à jour du modèle avant de vous lancer dans le code! N'hésitez pas à me poser des questions lors des séances de labo si vous avez des questions!

Amélioration de ViBe

Une fois la version classique de ViBe implémentée, on vous demande d'y incorporer une amélioration de votre choix. ViBe est une méthode très simple, alors il ne devrait pas être très difficile d'y ajouter quelque chose de bénéfique! Notez que cette amélioration ne devrait toutefois être triviale...

Quelques exemples d'améliorations possibles :

- filtrage des cartes de segmentation par composantes connectées (post-processing)
- utilisation d'une distance chromatique avec YCbCr ou HSV/HSL au lieu de L2 avec RGB
- utilisation d'un patron de texture binaire en plus de la couleur dans les échantillons

Évaluation

Pour ce travail, vous aurez vous-même à produire un rapport détaillé des performances de votre algorithme (**avec et sans amélioration**) sur les séquences de test simples fournies avec l'énoncé sur Moodle. Ces séquences sont données avec un masque de 'groundtruth' (vérité-terrain) par trame, que vous aurez à comparer à vos propres résultats. Puisque la segmentation vidéo étudiée ici est un problème de classification binaire, on vous demande de produire, **pour chaque vidéo,** les **trois métriques** suivantes dans votre rapport :

$$\text{Precision} = \#TP \,/\, (\#TP + \#FP) \quad (1)$$

$$\text{Recall} = \#TP \,/\, (\#TP + \#FN) \quad (2)$$

$$\text{F-Measure} = (2*\text{Precision}*\text{Recall}) \,/\, (\text{Precision} + \text{Recall}) \quad (3)$$

où

#TP = True Positives = nb pixels classifiés 'avant-plan' lorsque vérité est 'avant-plan' (4)

#FP = False Positives = nb pixels classifiés 'avant-plan' lorsque vérité est 'arrière-plan' (5)

#FN = False Negatives = nb pixels classifiés 'arrière-plan' lorsque vérité est 'avant-plan' (6)

Discutez dans votre rapport de la signification des métriques (1-3), et de la performance de ViBe en fonction de celles-ci (i.e. est-ce un algorithme plutôt sensible, ou robuste?). Discutez aussi du rôle des paramètres de ViBe, et comment ceux-ci peuvent affecter les métriques obtenues.

Remarque : #TP, #FP, et #FN doivent accumuler **tous** les pixels, de toutes les trames, mais **une vidéo à la fois**!

Barême

Implémentation et fonctionnement de ViBe : (10 pts)

- Initialisation modèle = 1 pt
- Recherche correspondances = 2 pts
- Classification = 1 pt
- Mise à jour = 3 pts
- Amélioration (pertinence+efficacité) = 3 pts

Rapport : (10 pts)

- Présentation ViBe et détails implémentation = 3 pts
- Présentation des paramètres choisis pour les tests = 1 pts
- Tableaux de résultats (métriques Precision/Recall/F-Measure) = 2 pts
- Commentaires et discussion sur efficacité = 2 pts
- Lisibilité, propreté et complétude = 2 pts

Total sur 20 pts

Références

[1] O. Barnich and M. Van Droogenbroeck, "ViBe: A Universal Background Subtraction Algorithm for Video Sequences," in *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709-1724, June 2011.

- Aide-mémoire (« Cheat sheet ») Matlab :

  http://web.mit.edu/18.06/www/Spring09/matlab-cheatsheet.pdf

- Guide complet Matlab :

  http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf

- C++ : utilisez OpenCV pour lire/écrire/modifier vos images!

  http://opencv.org/

  http://docs.opencv.org/doc/tutorials/tutorials.html