ROB313
TP3: Analyse vidéo et Tracking

Jahmal BAPTISTE
Bunthet SAY

Janvier 2020

Contents

1 Introduction

Dans ce TP, nous allons programmer et expérimenter l'algorithme de `Meanshift` et `La Transformée de Hough Généralisé` en utilisant la bibliothèque de traitement d'image `OpenCV` pour le suivi d'objet dans la séquence de vidéo.

Le travail que nous avons fourni est dans ce répositoire Git.

2 Meanshift

2.1 Question 1

L'algorithme Mean-Shift permet de trouver itérativement la moyenne géométrique (pondérée par une fonction noyau) d'un ensemble de points. La pondération en question crée une certaine localité dans l'obtention de la moyenne - autrement dit, la moyenne calculée est locale.

Le principal avantage de cet algorithme est qu'il est **non-paramétrique** - il n'a en effet pas besoin de modèle pour déterminer sa moyenne. Son deuxième plus grand avantage est qu'il est **rapide**. Ces deux avantages lui permettent d'être utilisé pour un suivi d'objet en temps-réel.

Cet algorithme souffre néanmoins d'un défaut majeur : il tombe très facilement sur des moyennes locales robustes. Nous avons pu observer cet écueil dans nos expérimentations avec l'algorithme de suivi de base : la fenêtre de suivi se fixe après quelques secondes sur une zone immobile (comme dans la FIGURE 1).

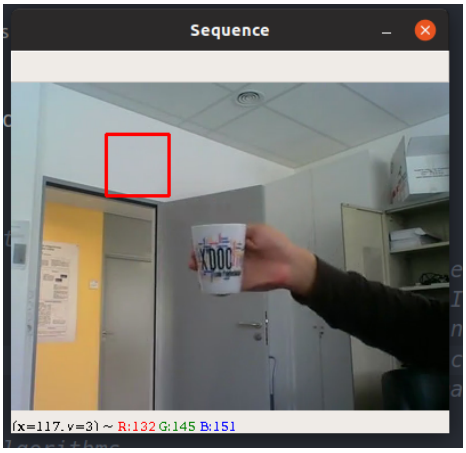

Figure 1: fenêtre de suivi coincée hors de l'objet

2.2 Question 2

L'affichage de la rétro-projection de l'histogramme des teintes permet de mieux comprendre le défaut dont l'algorithme de base souffre le plus : nous voyons à la FIGURE 2 que la rétro-projection possède plusieurs amas de valeurs élevées (révélant une forte multimodalité de la distribution de la rétro-projection). Ces amas fonctionnent comme des points attracteurs pour l'algorithme Mean-Shift (et coïncident avec les points de divergence observés plus tôt).

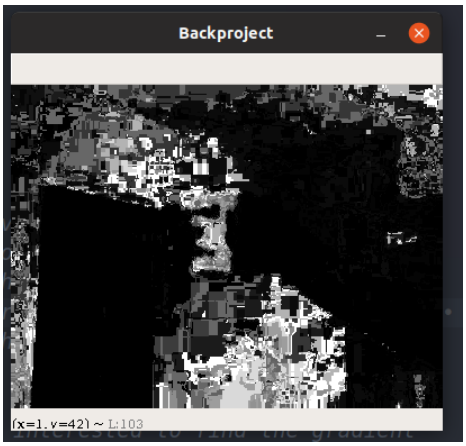

Figure 2: Rétro-projection de l'histogramme des teintes

Pour remédier à ce problème nous avons penser à deux approches différentes:

1. **actualiser l'histogramme des teintes de la fenêtre suivie**
   L'implémentation de cette modification s'est traduite par une mise à jour systématique de l'histogramme des teintes en prenant la fenêtre suivie actuelle comme nouvelle base pour le calcul de l'histogramme permettant la détermination de la fenêtre suivante;

2. **appliquer un masque à l'ensemble de l'image, et non pas à l'objet suivi seul**
   Nous avons appliqué le même masque que celui appliqué à l'objet à chaque image avant de calculer la rétro-projection (sur la base de l'histogramme de l'objet suivi).

Il s'est avéré que **seule l'application du masque généralisé s'est révélée concluante**. Il fallait cependant faire attention à bien choisir les intervalles discriminants pour réaliser le masque car la robustesse de la méthode est fortement dépendante de ces intervalles. Il fallait en effet calibrer ces intervalles de manière à ne garder que (dans l'idéal) les teintes qui ne se retrouvaient que dans l'objet que l'on désirait suivre.

Nous allons maintenant nous intéresser à la méthode que nous avons employée pour choisir ces intervalles discriminants.

Nous avons créé la fonction `hsv_bound_input` dans le fichier `utils.py` pour rentrer à la main les valeurs minimales et maximales de la *saturation* et la *valeur* (en coordonnées HSV) admises. Cette fonction affiche les composantes S et V (pour se faire une première idée des valeurs à conserver) et les histogrammes de ces valeurs (pour quantifier l'intuition obtenue avec les images).

En appliquant cette démarche sur la vidéo `Antoine_Mug.mp4`, nous avons pris l'intervalle [50., 255.] (respectivement [5., 100.]) sur la *saturation* (respectivement la *valeur*) pour masquer les pixels non-pertinents pour le suivi de l'objet.

En ce qui concerne l'échec de l'actualisation de l'histogramme des teintes, nous pensons que cette actualisation converge naturellement vers des zones immobiles car il est plus facile de trouver des ressemblances dans une zones qui ne se voit pas modifiée que dans une autre qui souffre de flous de bougé (notamment) - ce type de flou étant particulièrement présent dans les suivis d'objet.

3 Transformée de Hough

3.1 Question 3

En ce qui concerne le calcul du gradient, nous avons décidé de nous appuyer sur la valeur de la composante *valeur* de la transformée HSV.

L'orientation et le module du gradient, une fois ce support fixé, étaient calculables de manière assez immédiate.

L'affichage de l'orientation quant à lui n'était pas aussi immédiat. Il fallait en effet d'abord représenter les pixels qui n'avaient pas gradient assez prononcé en rouge mais il fallait aussi se poser la question de la couleur des autres pixels.

Nous avons décidé de représenter chaque angle par son cosinus et son sinus *(nous avons cherché à savoir si cette représentation avait un nom mais sans succès...)*. Chacune de ses composantes s'est alors vue attribuée une couleur (bleu (respectivement vert) pour le cosinus (respectivement le sinus)). L'attribution de la couleur s'est faite après une transformation affine de l'intervalle [-1., 1.] en l'intervalle [0., 255.]. Nous avons choisi une telle transformation (combinée avec une telle normalisation) pour conserver l'information sur la direction et le sens de l'orientation - deux orientations proches ont aussi une couleur proche.

La Figure 3 montre, avec un seuil en module de gradient à 75., l'image type obtenue avec cette représentation.

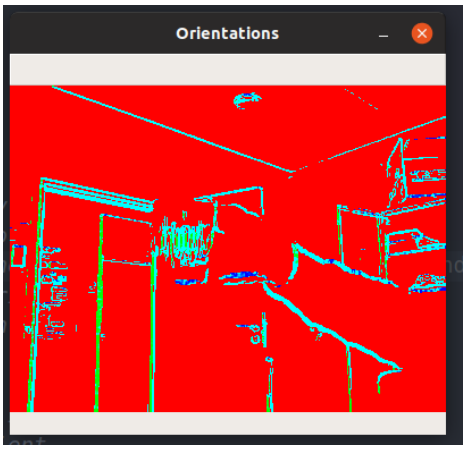

Figure 3: Orientations avec un seuil de module de gradient à 75.

3.2 Question 4

La *transformée de Hough* repose sur le principe de vote des composantes géométriques de l'objet suivi. Ces composantes sont tout simplement les orientations discrétisées des pixels de l'image objet (pour un module de gradient dépassant le même seuil mentionné plus tôt) stocké dans une **R-Table**. La structure de données que nous avons choisie pour les R-Tables est une `collections.OrderedDict()`.

La transformée de Hough typique (que nous avons obtenue) est donnée FIGURE 4.

Nous avons le regret d'annoncer le non-fonctionnement (conformément au comportement attendu) de notre algorithme... En effet, la valeur maximale de notre transformée de Hough est systématiquement en (0, 0) - mais ça ne s'arrête pas là.

Pour essayer de remédier à ce problème nous avons décidé de prendre en compte les `NUM_MAX_POINTS` plus grandes valeurs de notre transformée de Hough pour définir notre fenêtre de suivi - cette dernière étant définie par le plus petit rectangle contenant tous ces points. Cet essai s'est retrouvé infructueux, comme nous pouvons le voir en FIGURE 5, qui montre que les plus grandes valeurs de la transformée de Hough ne sont pas nécessairement dans l'objet...

REMARQUE IMPORTANTE : La transformée de Hough est particulièrement chronophage. Nous pensons que c'est essentiellement dû aux multiples boucles et tests que nous avons eu besoin d'employer pour son calcul.

Nous avons essayé de calculer cette transformée avec le moins de boucles et conditionnements possibles (en utilisant massivement la bibliothèque `Numpy`)

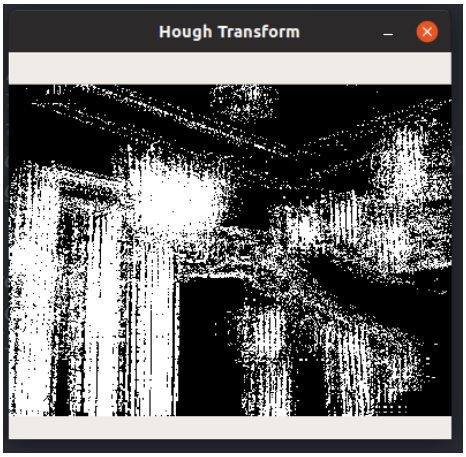

Figure 4: Transformée de Hough typique avec un seuil de module de gradient à 75.

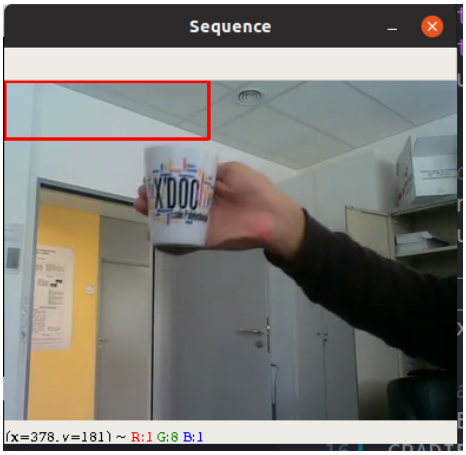

Figure 5: Fenêtre des 10 plus grandes valeurs de la transformée de Hough

mais nous n'avons finalement pas réussi à surmonter tous les obstacles de programmation (opérations entre types d'objets différents notamment). Notre tentative est visible (commentée) dans la fonction `compute_hough` du fichier `utils.py` et aboutit à un résultat qui se rapproche seulement de la transformée voulue.

3.3 Question 5

L'utilisation de l'algorithme Mean-Shift pour l'obtention de de la fenêtre de suivi fonctionne beaucoup mieux que l'algorithme précédent (maximum de transformée). Il fonctionne en fait à peu près (dans les faits) comme l'algorithme de base fourni avec ce TP.
Nous pouvons cependant ajouter quelques remarques :

- **le suivi tombe dans le même écueil que l'algorithme de base**
  Nous avons en effet observé une divergence de la fenêtre de suivi vers les zones modales fixes de la transformée de Hough;
- **le suivi est très peu robuste à la vitesse**
  les déplacements rapides créant un flou de bougé, le module du gradient au niveau de l'objet passe occasionnellement sous le seuil de coupure, faisant plusieurs points de la transformée disparaître; c'est à ces moments que le suivi diverge vers des zones modales fixes.

Nous pouvons ainsi dire que cette méthode de suivi fonctionne à condition que l'objet ne se déplace pas trop rapidement (sans prendre en compte le temps de calcul de la transformée de Hough).

4 Conclusion

De toutes les méthodes et variantes que nous avons essayé pour réaliser un suivi d'objet, nous pouvons affirmer que nous avons trouvé une méthode qui fonctionne bien (sur la vidéo testée en tout cas). Nous n'avons malheureusement pas réussi à en développer une autre mais cette méthode s'est montrée robuste sur tout le long de la vidéo, quelque soit la taille de la fenêtre initiale (ce qui n'est pas le cas des autres méthodes qui étaient assez sensibles à la première fenêtre - généralement elles fonctionnaient mieux avec de petites fenêtres).
Il s'agit de la **rétro-propagation de l'histogramme des teintes avec un masquage généralisé** (sans mise à jour de l'histogramme).

Nous aurions aimé pouvoir comprendre où nos autres algorithmes pourraient être améliorés mais nous n'avions pas le temps de développer plus en profondeur.