Compte rendu de la tâche 8 : algorithme de RANSAC

Interpolaspline

Avril 2020

1 Introduction

Il y a plusieurs méthodes différentes pour interpoler des données malgré des points aberrants. La méthode la plus intuitive est de détecter ces points et de les supprimer avant de tracer la spline. Cette méthode a été étudiée durant la tâche 4. La tâche 8 est une autre méthode pour interpoler des données : c'est l'étude d'un algorithme robuste, c'est à dire d'un algorithme qui interpole les données mais dont le résultat n'est pas réellement influencé par les points aberrants.

2 Objectif

L'objectif de la tâche est de comprendre l'algorithme de RANSAC, ainsi que de l'implémenter. Dans un second temps, une fois que tout fonctionne, l'objectif sera de comparer cet algorithme aux autres méthodes.

3 Explications

RANSAC est l'abréviation de RANdom SAmple Consensus. C'est un algorithme non déterministe, c'est à dire un algorithme qui peut renvoyer des résultats différents pour une même entrée (à cause de l'utilisation du hasard), c'est à dire pour les mêmes données et paramètres. Cet algorithme va un bon nombre de splines et va choisir celle qui correspond aux données mais qui engendre le moins d'erreur possible entre la spline et les données décrétées comme non aberrantes. Voilà, pour chaque modèle, les étapes :

1. L'algorithme choisit d'un certain nombre de points (distincts) aléatoires
2. Il calcule la spline cubique passant par ces points
3. Il mesure de la distance entre la courbe et chaque point afin de récupérer les points non aberrants suivant la spline créée
4. Si les points non aberrants trouvés lors de l'étape précédente sont peu nombreux, au moins un point utilisé pour créer la spline est alors sûrement aberrant. L'algorithme passe donc aux points suivants, en retournant à l'étape 1.
5. Si la spline passe proche de beaucoup de points, c'est qu'à priori elle correspond bien à la majorité des données (les autres sont décrétés comme aberrantes). L'algorithme calcule la spline de lissage correspondant aux données considérées comme non aberrantes.
6. La distance totale des points (non aberrants) par rapport à la courbe est calculée, c'est l'erreur de la spline de lissage. On sauvegarde le modèle s'il est meilleur que le précédent

(c'est à dire si cette erreur est plus faible), ou si c'est le premier modèle qui correspond aux données. Une fois qu'un certain nombre de données ont été testées, c'est le dernier modèle sauvegardé qui est considéré comme le modèle d'interpolation.

Avec cette version de l'algorithme, beaucoup de paramètres sont nécessaires (en plus des données) et doivent être réglés manuellement, en fonction des données et de ce que l'on souhaite obtenir : le nombre de spline à créer, l'erreur acceptée entre les points et la spline cubique jusqu'à laquelle les points ne sont pas considérés comme aberrants à l'étape 3, la fonction de distance à utiliser aux questions 3 et 6, le nombre de points non aberrants à partir duquel on considère que la spline correspond aux données, le nombre de points à considérer lors de la construction de la spline de l'étape 2 (c'est à dire le nombre de points tirés à l'étape 1), ainsi que le paramètre de lissage de la spline à l'étape 5.

Deux paramètres sont estimables au fur et à mesure de l'algorithme :
le nombre de spline à créer (c'est à dire le nombre d'itérations), et le nombre de points minimum à considérer pour avoir un résultat correct. Cela n'est possible qu'à condition de fournir à la place la probabilité d'avoir un résultat correct (< 1).
Les calculs, détaillés à la 6ième page du document cité en source, nous donne le nombre d'itérations à effecteur pour avoir une probabilité $p_{correct}$ d'avoir un résultat correct, avec une proportion ω de données non aberrantes (aussi appelées inlier) : $n_{iter} = \frac{log(1-p_{correct})}{log(1-\omega^n)}$

4 Implémentation

Les deux versions de l'algorithme évoquées ont été implémentées. RANSAC implémente la première version évoquée, tandis que la seconde, ransac_auto, calcule au fur et à mesure la proportion de données non aberrantes de l'échantillon et actualise le nombre maximum d'itérations grâce au résultat évoqué précédemment. C'est avec cette seconde que les tests ont été programmés.

5 Tests - Limites de l'algorithme

Cet algorithme possède plusieurs limites. On va essayer d'en expliquer quelques-unes : - Il faut, pour l'instant, définir certains paramètres manuellement, en fonction des exemples (certains ont été fixés avec ransac_auto). - L'algorithme est non déterministe, ce qui signifie que le résultat peut, avec peu de chances, être totalement faux. De plus il varie en fonction des essais. - Le paramètre de lissage est le même sur toute la fonction donc si celle-ci est stable à un endroit mais bouge beaucoup ailleurs, on ne peut pas avoir les deux. (test 22) - il est impossible (ou très difficile) de savoir si certains points sont aberrants ou pas (exemple num 20) : comment savoir si ce sont les deux/trois points du haut ou du bas qui sont aberrants ? (à gauche) - Certains points sont décrétés comme aberrants alors que visuellement, ils ne le sont pas. (Exemple num 11)

6 tests

7 Sources

`http://w3.mi.parisdescartes.fr/~lomn/Cours/CV/SeqVideo/Material/RANSAC-tutorial.pdf`