Forêt de Régression Précise basée sur des Caractéristiques Éparses pour la Relocalisation de Caméra en Temps-Réel

Nam-Duong DUONG[1], Catherine SOLADIE[1,2], Amine KACETE[1], Pierre-Yves RICHARD[1,2], Jerome ROYAN[1]

[1]IRT b-com
1219 Avenue des Champs Blancs, 35510 Cesson-Sevigné, France

[2]IETR/CentraleSupelec
Avenue de la Boulaie, 35576 Cesson-Sevigné, France

{Nam-Duong.Duong, Amine.Kacete, Jerome.Royan}@b-com.com
{Catherine.Soladie, Pierre-Yves.Richard}@supelec.fr

Résumé – La relocalisation de caméra se réfère à la problématique de définir la pose d'une caméra qui est six degrés de liberté (6-DoF) exprimée dans le système de coordonnées du monde sans contrainte temporelle. La relocalisation de caméra est nécessaire dans les systèmes de localisation. Cependant, il est encore difficile d'avoir une méthode précise et temps réel. Dans cet article, nous introduisons notre méthode hybride qui fusionne apprentissage automatique et approche géométrique pour une relocalisation rapide et précise de caméra à partir d'une seule image RVB. Nous proposons une forêt de régression qui utilise à chaque fonction de répartition un vecteur complet de caractéristiques, afin d'améliorer la précision des correspondances de points 2D-3D. Les résultats indiquent que notre méthode est quasi temps réel ($50ms$ par image), et est aussi précise que les meilleures méthodes de relocalisation. De plus, les résultats montrent également que notre méthode est capable de relever les défis de l'occlusion partielle ainsi que du changement d'éclairage.

Abstract – Camera relocalization refers to the problematics of defining the pose (translation and rotation) of a camera which is six degrees of freedom (6-DoF) expressed in a the world coordinate system with no temporal constraint. Camera relocalization is needed in localization systems. However, it is still challenging to have both a real-time and accurate method. In this paper, we introduce our hybrid method merging machine learning approach and geometric approach for fast and accurate camera relocalization from a single RGB image. We propose an efficient regression forest that uses a whole feature vector at each split function of our regression forest to improve the accuracy of 2D-3D point correspondences. The results indicate that our method is a near real-time hybrid method ($50ms$ per frame), and is as accurate as the best camera relocalization methods. Besides, the results also demonstrate that our method is able to address challenges concerning partial occlusion as well as illumination changes.

1 Introduction

La solution classique pour l'estimation de la pose de caméra est connue sous le nom de Simultaneously Localization And Mapping (SLAM). SLAM traite une séquence ordonnée d'images. Dans le cas d'un mouvement rapide de caméra ou d'un changement soudain de point de vue, comme avec une caméra portative, l'échec de suivi interrompt l'estimation de la pose de caméra. Lorsque cela se produit, la relocalisation de caméra est nécessaire pour récupérer la pose de caméra après le suivi perdu, plutôt que de redémarrer la localisation à partir de zéro. Cependant, les méthodes existantes de relocalisation de caméra dans SLAM doivent stocker un grand nombre de points-clés [7] ou d'images-clés [4]. Par conséquent, l'utilisation de la mémoire ainsi que le temps de traitement augmentent conjointement à la taille des modèles.

Un bon système de relocalisation de caméra nécessite deux exigences principales : la précision et le temps de calcul. De plus, il doit être résistant à des occlusions et à des changements d'éclairage. Récemment, les approches d'apprentissage automatique pour la relocalisation de caméra ont semblé s'attaquer à ces contraintes. Ces méthodes [6, 5] peuvent estimer la pose de caméra en temps réel à partir de chaque image. Cependant, les limitations de ces méthodes résident dans leur précision et l'absence de score de confiance pour chaque estimation de pose. Afin d'améliorer la précision aussi bien que de régler des prédictions de l'incertitude dans les approches d'apprentissage en profondeur, [8, 2] ont présenté des méthodes hybrides utilisant l'apprentissage automatique et des algorithmes géométriques pour la relocalisation des caméras avec une plus grande précision. En revanche, ils prennent plus de temps pour optimiser la pose de caméra à partir de milliers de correspondances et ne peuvent donc pas s'adresser aux applications de la réalité augmentée qui nécessite temps réel.

Dans cet article, notre objectif est de proposer une méthode de relocalisation temps réel et précise en utilisant uniquement des images RVB. La limitation principale des méthodes hybrides de l'état de l'art concerne le temps de calcul de la partie géométrique du processus. Ce temps du calcul est élevé, en rai-

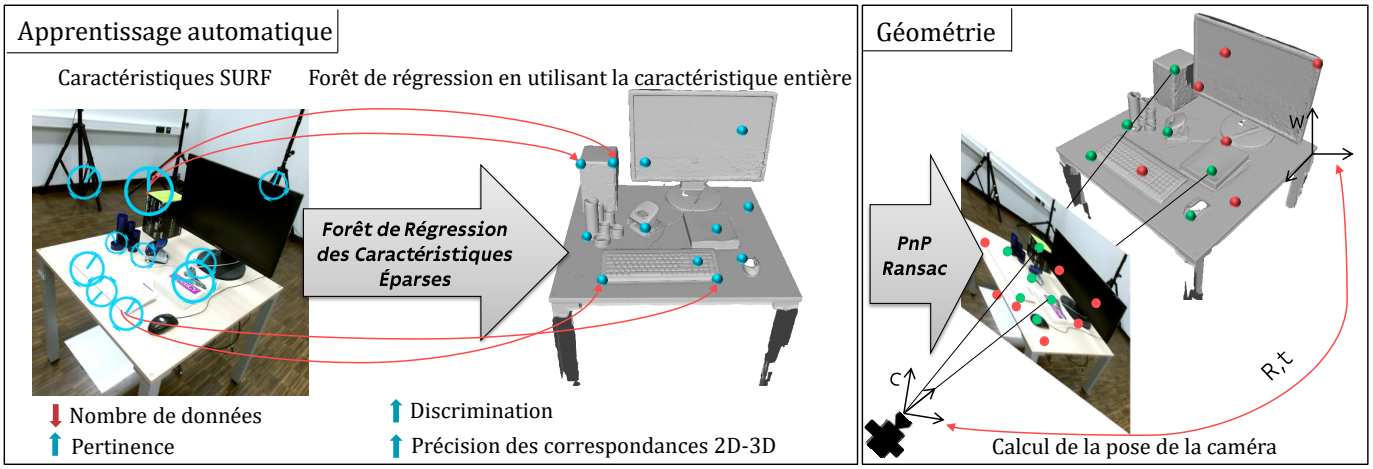

FIGURE 1 – Notre méthode hybride. A partir d'un ensemble de caractéristiques SURF extraites sur chaque image RVB, notre forêt de régression prédit un ensemble de positions 3D (points bleus) dans le système de coordonnées du monde. La partie géométrique (algorithmes classiques PnP et RANSAC) filtre les inliers (points verts), élimine les outliers (points rouges) et calcule la pose de caméra.

son du nombre excessif de données utilisées comme entrées de cette partie géométrique, et aussi de leur faible précision. Par conséquent, nous présentons dans cet article une méthode hybride, qui se concentre sur l'amélioration des données fournies par la partie apprentissage automatique du processus.

Nos contributions consistent à diminuer le nombre de prédictions des correspondances 2D-3D traitées à chaque étape de la partie d'apprentissage automatique, et à augmenter leur pertinence en même temps. Nous proposons une nouvelle fonction de répartition à chaque noeud d'une forêt de régression, qui prend des vecteurs de caractéristiques entiers du SURF (Speed Up Robust Features) [1] comme entrées. Cette fonction de répartition permet d'améliorer la précision des correspondances de points 2D-3D grâce à la prise en compte de toutes les informations pertinentes du vecteur de caractéristiques. En outre, notre méthode est robuste pour les changements d'éclairage ainsi que les changements d'échelle et de rotation, car nous utilisons des caractéristiques SURF. De plus, nous abordons le défi de l'occlusion partielle en utilisant des caractéristiques éparses (sparse feature) au lieu d'une image entière.

2 Méthode proposée

Dans cet article, nous présentons une méthode hybride pour la relocalisation des caméras en temps réel à partir d'images RVB. La figure 1 illustre le pipeline correspondant, qui peut se résumer en trois étapes principales : l'extraction de caractéristiques éparses ; notre forêt de régression pour définir des correspondances de points 2D-3D ; calcul géométrique de la pose de caméra à partir de ces correspondances.

2.1 Extraction de caractéristiques éparses

À partir de chaque image RVB, nous extrayons un ensemble de caractéristiques à partir de points clés, notés par $\{f_i; p_i\}$ où f_i est un vecteur de caractéristique et $p_i = (u_i, v_i)$ définit les coordonnées 2D dans l'image. Nous sélectionnons l'extraction de caractéristiques SURF [1] qui est invariante à l'échelle et à la rotation. Cela améliore la capacité à localiser les positions 3D à partir de points de vue différents.

Pour la phase d'apprentissage, nous devons labelliser les données d'entraînement $\{f_i; p_i\}$ avec les coordonnées du monde 3D correspondantes : $P_i^w = (X_i^w, Y_i^w, Z_i^w)$. Donc, nous pouvons utiliser les images RVB-D de caméra calibrée avec leurs poses de caméra correspondantes pour définir les labels des positions 3D (Notez que dans la phase de test, nous utilisons uniquement des images RVB). À partir de la position $p_i = (u_i, v_i)$ de chaque point-clé détecté dans l'image RVB et de la valeur de profondeur correspondante D_i dans l'image de profondeur, les coordonnées du monde P_i^w correspondant est calculée :

$$P_i^w = D_i K^{-1} \left[\begin{array}{c|c} R & t \end{array} \right] \begin{bmatrix} P_i^c \\ 1 \end{bmatrix}$$

Où K est une matrice de paramètres intrinsèques de caméra. La pose $T = [R|t]$ de caméra incluant la matrice de rotation R et le vecteur de translation t pour chaque trame est supposée être connue à l'avance.

2.2 Forêt de régression

Comme introduit dans [3], une forêt de régression est un ensemble de N arbres de décision. Chaque arbre est composé de noeuds et de feuilles. Chaque noeud est représenté par les paramètres d'une fonction de répartition. Au lieu d'utiliser la fonction de répartition binaire en utilisant l'intensité des pixels comme dans [8], nous proposons une nouvelle fonction de répartition qui utilise le vecteur de caractéristiques entier pour séparer des données de manière optimale. Un noeud i de notre forêt est paramétré par $\theta_i = \{ref_i, \tau_i\}$ où ref_i est un vecteur de caractéristique de référence et τ_i est un seuil.

Pour chaque arbre de décision, un vecteur de caractéristiques f_j est considéré à partir du noeud de racine et descendant jusqu'à une feuille en évaluant de manière répétée la fonction de

répartition à chaque noeud i :

$$h(f_j, \theta_i) = \begin{cases} 0, & \text{si } d(ref_i, f_j) < \tau_i, \text{ gauche} \\ 1, & \text{si } d(ref_i, f_j) \geqslant \tau_i, \text{ droite} \end{cases} \quad (1)$$

Où $d(ref_i, f_j) = \|ref_i - f_j\|_2^2$ est la fonction de distance euclidienne entre les deux vecteurs.

Pour **l'apprentissage de la forêt de régression**, ensemble de vecteurs de caractéristiques à partir des images d'apprentissage. Pour chaque arbre, un sous-ensemble de caractéristiques S est choisi au hasard. À chaque noeud de répartition i, nous évaluons un ensemble de caractéristiques S_i qui est un sous-ensemble de S. Les paramètres θ_i permettent de répartir l'ensemble des caractéristiques S_i en noeud enfant gauche S_i^G et en noeud enfant droite S_i^D. Les paramètres θ_i sont appris pour maximiser une fonction d'objectif visant à réduire la variance spatiale.

$$Q(S_i, \theta_i) = V(S_i) - \sum_{d \in G, D} \frac{|S_i^d(\theta_i)|}{|S_i|} V(S_i^d(\theta_i)) \quad (2)$$

$$\text{avec } V(S) = \frac{1}{|S|} \sum_{m \in S} \|m - \overline{m}\|_2^2 \quad (3)$$

Où m est les coordonnées 3D dans le repère du monde et $\overline{m}$ est la moyenne de m. L'apprentissage se termine lorsqu'il atteint une profondeur maximale D_{max} ou lorsque l'ensemble S_i contient peu de données. Chaque feuille stocke une distribution gaussienne $\mathcal{N}(m, \bar{m}, \Sigma_m)$ correspondant aux positions 3D.

Dans la phase de prédiction de la forêt de régression, chaque caractéristique extraite d'une image RVB passe par la forêt de régression. Nous obtenons un ensemble de prédictions. De plus, la covariance de chaque feuille fournit un score de confiance de chaque prédiction. Ce paramètre nous aide à sélectionner les prédictions les plus fiables pour calculer la pose de caméra. Ceci exploite l'incertitude de la prédiction à l'image des méthodes d'apprentissage en profondeur [5].

2.3 Calcul de la pose de caméra

Quand des caractéristiques sont passe par la forêt de régression, nous obtenons un ensemble de correspondances 2D-3D. Nous utilisons d'abord PnP (Perspective-n-Points) et Ransac (Random sample consensus) pour supprimer les bruit (outliers) et conserver les prédictions exactes (inliers). Plus précisément, Ransac génère un ensemble de poses hypothétiques $\mathcal{T} = \{T_i\}$ en effectuant PnP sur des sous-ensembles aléatoires de correspondances de points 2D-3D. Pour chacune des poses ainsi obtenues, une erreur de re-projection est calculée, permettant de séparer les inliers des outliers à l'aide d'un seuil. Nous retenons alors les inliers associés à l'hypothèse qui en fournit le plus grand nombre, à savoir :

$$\max_{\forall T_i \in \mathcal{T}} \sum_{p_j \in p} \rho(\alpha_{ij}), \text{avec } \rho(\alpha_{ij}) = \begin{cases} 1, & \text{si } \alpha_{ij} < \tau \\ 0, & \text{autrement} \end{cases}$$

Où $\alpha_{ij} = \|p_j - K T_i^{-1} \hat{P}_j^w\|^2$ et τ est le seuil maximal d'erreur de re-projection qui définit les inliers. Ainsi le pixel j est considéré comme un inlier de l'hypothèse T_i si $\rho(\alpha_{ij}) = 1$. La pose de caméra finale est calculée en exécutant PnP une fois de plus sur tous ces inliers, afin de minimiser la fonction d'erreur de re-projection. Inférer la pose de caméra à partir de plusieurs patchs avec une méthode de filtrage basée sur les algorithmes PnP et Ransac permet d'éliminer les outliers sur les objets en mouvement dans la scène, afin de relever le défi des scènes avec occlusion partielle.

3 Expérimentations

Nous évaluons notre méthode sur la base de données de 7 scènes [8], la base de données CoRBS [9] et la base de données BCOM [1]. Les scènes sont toutes des scènes d'intérieur. Chaque base de données fournit des images RVB-D, une matrice intrinsèque de caméra et des annotations (pose de caméra pour chaque image).

7 scènes [8] contient sept scènes. Ces données sont extrêmement difficiles à traiter en raison de la rotation pure ou du mouvement rapide de caméra qui fournit de nombreuses images floues.

CoRBS [9] est plus précis grâce à l'utilisation de plusieurs capteurs. Elle compose de trois scènes qui contiennent de nombreuses surfaces sans texture et plates.

BCOM contient quatre squences correspondant à quatre trajectoires absolument différentes d'une même scène avec des changements d'éclairage et de rotation de caméra. Cette base de données vise à évaluer la capacité de généralisation des méthodes de relocalisation.

3.1 Comparaison avec les méthodes de l'état de l'art

Méthode	Translation Erreur (cm)	Rotation Erreur (°)	Temps(ms)
Active search [7]	4.9	2.46	$\geq$100
PoseNet 2 [5]	23.1	8.12	**5**
DSAC [2]	**3.9**	**1.6**	1500
Notre méthode	**3.9**	**1.7**	$\leq$**50**

TABLE 1 – Comparaison de notre méthode avec des méthodes de l'état de l'art en termes de précision et de temps de calcul. Le résultat est mesuré en faisant la moyenne de toutes les erreurs de pose médiane sur la base de données de 7 scènes. Le temps de calcul est mesuré pour une seule image en millisecondes sur des matérielles comparables.

Nous avons comparé notre méthode aux méthodes de l'état de l'art qui utilisent uniquement des images RVB dans les tests : méthodes de caractéristiques éparses [7] (Active Search), méthodes d'apprentissage en profondeur [5] (PoseNet2), méth-odes hybrides [2]. Nous utilisons la base de données de 7 scènes.

Le tableau 1 montre que notre méthode surpasse nettement les méthodes [5, 7] en terme de précision (erreur de translation et de rotation). Bien que PoseNet2[5] soit très rapide pour

1. https ://github.com/duongnamduong/bcom-dataset

relocaliser avec $5ms$ au cours des tests, ses résultats sont toujours peu précis. Notre méthode est six fois plus précise que la leur. Concernant le temps de test, notre méthode fonctionne toujours en temps réel. Notez également que l'apprentissage de notre forêt est plus rapide que PoseNet2 : notre méthode prend moins de deux minutes pour s'entraîner un arbre sur une CPU ; PoseNet2 prend entre quatre heures et une journée sur un GPU Titan X.

La précision de notre résultat est approximativement égale à celle de [2]. De plus, notre méthode améliore considérablement les approches hybrides en termes de temps de calcul. Notre méthode permet de relocaliser la caméra en temps réel à $50ms$ par image, alors que [2] prennent plus d'une seconde par image, car l'optimisation de la pose de caméra nécessite beaucoup de temps pour traiter des milliers de données saisies.

3.2 Robustesse à des scènes dynamiques

Nous présentons les avantages de notre méthode afin de tacler les scénes dynamiques dans ce paragraphe. Sur la base de données CoRBS, nous avons synthétisé de manière aléatoire des changements d'éclairage et de l'occlusion partielle, comme illustré dans la figure 2. Nous observons que notre méthode est capable d'adresser les changements d'éclairage et les occlusion partielle grâce à la caractéristique invariante SURF et RANSAC élimine les outliers.

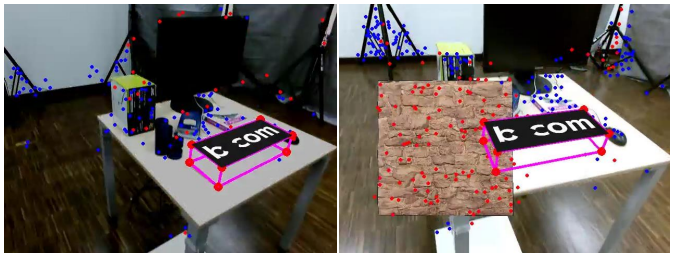

FIGURE 2 – Quelques exemples de scènes dynamiques en réalité augmentée. Notre méthode est robuste aux changements d'éclairage et à l'occlusion partielle.

3.3 Performance de généralisation

Test \ Entraînement	T_1	T_2	T_3	T_4
T_1	**3.1 / 1.72**	3.5 / 1.92	4.8 / 2.75	3.4 / 3.41
T_2	4.4 / 2.35	**2.6 / 1.63**	3.5 / 2.36	4.1 / 3.05
T_3	3.8 / 2.80	4.2 / 3.84	**2.7 / 1.86**	4.4 / 2.96
T_4	4.5 / 3.65	4.9 / 4.68	5.2 / 3.62	**2.5 / 2.38**

TABLE 2 – Performances de généralisation sur la base de données BCOM. Notre forêt de régression est apprise à partir de chaque trajectoire. Elle est ensuite utilisée pour évaluer les autres trajectoires. Les erreurs sont affichées au format (cm)/(°).

Nous évaluons la capacité de généralisation de notre méthode sur l'ensemble de données BCOM, qui contient quatre trajectoires absolument différentes. Nous choisissons respectivement chaque trajectoire pour entraîner notre modèle forêt de régression. Ensuite, nous utilisons ce modèle pour estimer la pose de caméra pour chaque image sur les trois autres trajectoires. Le tableau 2 montre que les meilleurs résultats sont obtenus lorsque la trajectoire de test est la même que la trajectoire d'apprentissage. Néanmoins, nous observons que sur les autres trajectoires, les résultats restent bons.

4 Conclusion

Dans cet article, nous avons proposé une nouvelle méthode hybride combinant l'approche d'apprentissage et l'approche géométrique pour une relocalisation de caméra précise et temps réel. Nous avons proposé une nouvelle fonction de répartition utilisant la caractéristique SURF entière à chaque noeud de la forêt de régression pour définir efficacement les correspondances de points 2D-3D. De plus, nous montrons les résultats favorables de notre méthode en comparant avec des méthodes de l'état de l'art. Nous démontrons que notre méthode est capable des scénes dynamiques ainsi que des généralisations.

Références

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3) :346–359, 2008.

[2] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac - differentiable ransac for camera localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] L. Breiman. Random forests. *Machine learning*, 45(1) :5–32, 2001.

[4] B. Glocker, J. Shotton, A. Criminisi, and S. Izadi. Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5) :571–583, 2015.

[5] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[6] A. Kendall, M. Grimes, and R. Cipolla. Posenet : A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2938–2946, 2015.

[7] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9) :1744–1756, 2017.

[8] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013.

[9] O. Wasenmüller, M. Meyer, and D. Stricker. Corbs : Comprehensive rgb-d benchmark for slam using kinect v2. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–7. IEEE, 2016.