

SCALE-INVARIANT CORNER KEYPOINTS

Présentation de l'article

Louis MENNRATH
Charbel GHANEM

Master SETI
Université Paris-Saclay

21 novembre 2025

The logo of the University of Paris-Saclay, featuring the text "université" in a serif font and "PARIS-SACLAY" in a bold sans-serif font, both in white on a dark blue background. A small white dot is positioned above the "e" in "université".

université
PARIS-SACLAY



Plan de la Présentation

- 1 Présentation de SICK
- 2 Performances de SICK
- 3 Conclusion



Présentation de SICK



Edges, Corners, Scale-space

Idée générale

Détecter les arêtes à plusieurs échelles, puis trouver les endroits où ces arêtes changent fortement de direction.

Coin \iff changement de direction du gradient



Détection d'arêtes à plusieurs échelles

Gradient, CoM filter

$$G(t) = \sqrt{G_x(t)^2 + G_y(t)^2}$$

- Gradient calculé avec filtres CoM : tailles 3×3 à 17×17
- Chaque taille = une **échelle**



Non Maximum Suppression (NMS)

Maxima, Gradient

Principe

On conserve un pixel d'arête seulement s'il est un maximum local

- dans la direction du gradient (2D)
- entre échelles voisines (Scale-space)

$$G(t) = \max(G(t-1), G(t), G(t+1))$$



Pyramide d'images

Multi-résolution

- On crée plusieurs versions réduites de l'image
- À chaque niveau : détection des arêtes + NMS

$$I_0, I_1 = \text{down}(I_0), I_2 = \text{down}(I_1), \dots$$



Organisation des arêtes en chaînes

Edge chains

- Les pixels d'arêtes sont reliés en séquences 1D ordonnées
- Permet de parcourir chaque arête comme un chemin

$$\text{Chaîne} = \{p_1, p_2, \dots, p_t\}$$



Corneriness le long d'une arête

Corner score, Gradient direction

$$\mathbf{V}_l(t) = \sum_{i=t-w}^{t-1} \mathbf{g}(i) \quad \mathbf{V}_r(t) = \sum_{i=t+1}^{t+w} \mathbf{g}(i)$$

$$C(t) = \frac{\|\mathbf{V}_r(t) - \mathbf{V}_l(t)\|}{\|\mathbf{V}_r(t)\| + \|\mathbf{V}_l(t)\|}$$



1D NMS

- On applique le NMS **le long de chaque chaîne**
- On garde les maxima de $C(t)$

$$C(t) = \max(C(t-1), C(t), C(t+1))$$



Threshold

Score final du point clé

$$S_k(t) = C(t) G(t)$$

- On écarte les points avec un score trop faible
- On garde les coins les plus stables



Orientation des keypoints

Orientation, Rotation invariance

$$\theta(t) = \text{atan2}(G_y(t), G_x(t))$$

- Orientation directement donnée par le gradient
- Pas besoin d'histogrammes (plus simple que SIFT)



Performances de SICK



Conclusion

Conclusion



Sources



Bo Li, Haibo Li, U. S. (2014).
Scale-invariant corner keypoints.
IEEE ICIP.



deepanshut041 (2019).
Introduction to feature detection and matching.
medium.com.



Girod, B. (2013-2018).
Keypoint detection 1.
In Stanford University.



Lowe, D. G. (1999).
Object recognition from local scale-invariant features.
In International Conference on Computer Vision.



Mikolajczyk, K. (2005).
A comparison of affine region detectors.
International Journal of Computer Vision.



Stefan Leutenegger, Margarita Chli, R. Y. S. (2011).
Brisk : Binary robust invariant scalable keypoints.
IEEE International Conference on Computer Vision.

Limites de l'article

- Dataset utilisé dans l'article : benchmark datant de **2005** alors que l'article est publié en 2014.
→ Résultats moins représentatifs face aux méthodes modernes.
- SICK excellent pour **changement d'échelle** et **rotation**, mais seulement **moyen** pour le **viewpoint change** et **faible** en **illumination change**.
- Méthode **non intégrée** dans OpenCV \Rightarrow adoption limitée dans la communauté.



Conclusion de l'article

- L'article propose une nouvelle méthode de détection de coins, appelée **SICK**, basée sur l'analyse des **arêtes** et le suivi de leurs **changements de direction**.
- L'approche repose sur une combinaison efficace :
 - détection d'arêtes multi-échelles,
 - organisation en chaînes 1D,
 - mesure locale de cornerness.
- SICK offre une **invariance naturelle à l'échelle** et une excellente robustesse à la **rotation**, tout en restant computationnellement léger.
- Les résultats expérimentaux montrent que SICK peut se comparer favorablement aux détecteurs classiques, tout en maintenant une structure algorithmique simple.