# AID : Un descripteur invariant affine pour SIFT

M. Rodríguez† G. Facciolo† R. Grompone von Gioi † P. Musé § J.-M. Morel† J. Delon‡

† CMLA, ENS Paris-Saclay, CNRS, Université Paris-Saclay, 94235 Cachan, France
§ IIE, Universidad de la República, Uruguay
‡ MAP5, Université Paris Descartes, France

mariano.rodriguez@cmla.ens-cachan.fr

## Résumé

*Pour mettre en correspondance des images, il est classique de construire des descripteurs encodant l'information locale autour de points clefs et invariants à certaines transformations géométriques. Le succès de ces approches locales repose sur le fait que les déformations induites par les changements de points de vue entre images sont localement bien approchées par des transformations affines. Ainsi, de nombreux travaux ont été proposés dans la littérature pour construire des descripteurs locaux affine invariants. Cependant, malgré de nombreuses avancées en ce sens, aucun descripteur totalement affine invariant n'a pu être construit, comme montré dans [1, 2]. Afin de dépasser cette limitation, les méthodes plus récentes simulent plusieurs transformations affines des images à comparer pour atteindre une invariance affine plus complète [1]. Dans ce travail, nous proposons un descripteur local qui capture l'invariance affine sans avoir besoin de simulations de points de vue. Ce descripteur est construit en entraînant un réseau de neurones profond à associer des représentations vectorielles similaires à des patchs liés par des transformations affines. Ces vecteurs peuvent ensuite être comparés très efficacement pour l'étape de mise en correspondance. Nous montrons que le descripteur ainsi construit, nommé SIFT-AID, surpasse l'état de l'art en matière de conservation des propriétés d'invariance affine.*

## Mots Clef

Mise en correspondance d'images, invariance affine, IMAS, SIFT, RootSIFT, réseaux de neurones convolutifs.

## Abstract

*The classic approach to image matching consists in the detection, description and matching of keypoints. The descriptor encodes the local information around the keypoint. An advantage of local approaches is that viewpoint deformations are well approximated by affine maps. This motivated the quest for affine invariant local descriptors. Despite numerous efforts, such descriptors remained elusive, ultimately resulting in the compromise of using viewpoint simulations to attain affine invariance. In this work we propose a CNN-based patch descriptor which captures affine invariance without the need for viewpoint simulations. This is achieved by training a neural network to associate similar vectorial representations to patches related by affine transformations. During matching, these vectors are compared very efficiently. The invariance to translation, rotation and scale is still obtained by the first stages of SIFT, which produce the keypoints. The proposed descriptor outperforms the state-of-the-art in retaining affine invariant properties.*

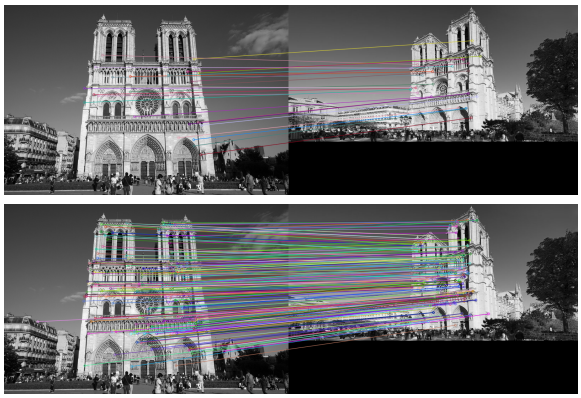

FIGURE 1 – À Gauche : correspondances obtenues par Affine-RootSIFT (48 correspondances). À Droite : correspondances obtenues par la méthode proposée dans cet article, SIFT-AID (295 correspondances).

## Keywords

image comparison, affine invariance, IMAS, SIFT, RootSIFT, convolutional neural networks.

## 1 Introduction

L'approche classique pour la mise en correspondance d'images est généralement divisée en trois étapes : détection, description et appariement [3]. Tout d'abord, les points clés sont détectés dans les deux images à comparer. Ensuite, les régions autour de ces points sont décrites par

des descripteurs locaux. Enfin, tous ces descripteurs sont comparés et éventuellement appariés. Les étapes de détection et de description sont habituellement conçues pour assurer une certaine invariance aux divers changements géométriques ou radiométriques. L'intérêt des descripteurs locaux est que les déformations de points de vue sont bien approchées localement par des transformations affines. En effet, pour toute déformation régulière, son approximation de Taylor au premier ordre est une transformation affine. Cette observation a motivé le développement de méthodes de comparaison basées sur des descripteurs locaux aussi invariants aux transformations affines que possible.

Pour assurer cette invariance, certains auteurs ont proposé des détecteurs de régions d'intérêt à base de moments [4, 5], comme les détecteurs Harris-Affine et Hessian-Affine [6, 7]. Les détecteurs invariant aux transformations affines locales peuvent aussi être basés sur des bords [8, 9], l'intensité [10, 9], ou l'entropie [11]. Enfin, les détecteurs MSER (Maximally Stable Extremal Region) [12] et LLD (Level Line Descriptor) [13, 14, 15] reposent tous deux sur les lignes de niveau des images. Pourtant, l'invariance affine de ces descripteurs dans les images acquises par des caméras réelles est limitée par le fait que le flou optique et les transformations affines ne commutent pas, comme démontré dans l'article [16]. Ainsi, aucun des descripteurs mentionnés précédemment ne peut être considéré comme totalement invariant affine. Dans [2], il est démontré que RootSIFT [17] est le descripteur le plus robuste aux changements de point de vue affines (jusqu'à $60°$). Pour aller plus loin, plusieurs solutions basées sur des simulations de transformations ont été proposées : ASIFT [1], FAIR-SURF [18], MODS [19], Affine-AC-W [20]. Certaines versions optimales ont été proposées dans [21], dont Optimal Affine-RootSIFT.

De nos jours, les descripteurs locaux, autrefois conçus manuellement, sont de plus en plus générées automatiquement à partir de données, afin d'améliorer encore les performances de mise en correspondance. En imitant le processus classique de comparaison d'images, ces approches par réseaux de neurones consistent à apprendre une mesure de similarité entre les patchs des images. Dans [22] par exemple, trois architectures de score de similarité sont introduites, contenant un réseau convolutionnel (CNN) et un réseau décisionnel. Pour l'appariement stéréo, deux architectures basées sur les CNNs sont proposées dans [23], l'une d'elles calculant le score de similarité avec l'opérateur de proximité cosinus.

La correspondance géométrique entre images basée sur des réseau des neurones a également été testée pour le cas des transformations affines et homographiques [24, 25]. Dans [24], la couche POOL4 du réseau VGG-16 [26] est utilisée pour acquérir des caractéristiques à partir d'images et de cartes de corrélation données en entrée à un réseau de régression qui fournit la meilleure transformation affine pour rapprocher la requête de l'image cible. Les auteurs de [25] proposent un réseau estimant directement l'homographie reliant la requête à l'image cible. Les deux approches [24, 25] sont entraînées sur des images de synthèse, mais ni l'une ni l'autre ne tient compte du flou causé par le zoom arrière ou le point de vue des caméras.

Dans cet article, nous combinons certaines idées d'une méthode conçue manuellement (SIFT) avec une étape d'apprentissage, afin d'obtenir un algorithme de mise en correspondance d'images à la fois invariant affine et rapide, et surtout capable de capturer de forts changements de points de vue. La méthode proposée est basée sur les premières étapes de SIFT [3, 27], qui assurent l'invariance aux transformations dites de similarités (translations, rotations et zooms) jusqu'aux petites perturbations affines (voir [28] pour une preuve mathématique). A ce stade, le descripteur SIFT est remplacé par un réseau convolutionnel (Figure 2) qui prend en entrée un patch $60 \times 60$ et produit un nouveau descripteur vectoriel de 6272 éléments. Le réseau est entraîné sur un ensemble de données contenant des paires de patchs liés par des transformations affines, de manière à produire des vecteurs descripteurs similaires pour des paires affines, et des vecteurs différents [23] le reste du temps.

Une manière simple de mesurer la similarité entre deux descripteurs est d'utiliser l'opérateur de proximité cosinus, c'est-à-dire

$$\cos(\mathbf{x}, \mathbf{y}) \coloneqq \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|}.$$

Nous entraînons donc notre réseau afin qu'il envoie deux patchs liés par une transformation affine sur deux descripteurs proches pour cette mesure de similarité (donc proches en terme d'angle). Comme alternative, un descripteur binaire peut être construit en ne conservant que le signe de chaque composante vectorielle. Cela permet d'économiser de la mémoire et d'accélérer le processus d'appariement, tout en conservant le même niveau de performance et le même pouvoir discriminatif. La Figure 1 compare sur un exemple la méthode proposée et la méthode Affine-RootSIFT.

# 2 Simulations affines

Pour construire notre descripteur AID à l'aide d'un réseau profond, nous avons besoin de générer des données synthétiques qui serviront à entraîner ce réseau, et donc de modéliser correctement les changements de points de vue affines. Soit $u$ une image, $\mathcal{A}$ l'ensemble des transformations affines et $\mathcal{S}$ l'ensemble des similitudes, i.e. les combinaisons de translations, rotations et zooms. On définit $Au(\mathbf{x}) = u(A\mathbf{x})$ pour $A \in \mathcal{A}$. Nous définissons

$$\mathbf{A}^+ = \{A + b \in \mathcal{A} | \det(A) > 0\}$$

où $A$ est une application linéaire et $b$ un vecteur de translation. Nous définissons également le sous-ensemble

$$\mathbf{A}^+_* = \mathbf{A}^+ \setminus \mathcal{S}.$$

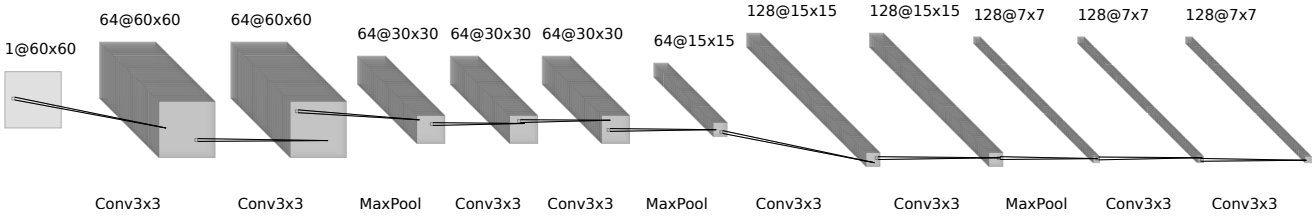


FIGURE 2 – Le descripteur proposé est calculé à l'aide d'un réseau de neurones qui produit un vecteur de dimension 6272.

Il a été prouvé dans [16] que chaque $A \in \mathbf{A}_*^+$ peut être décomposé de manière unique sous la forme

$$A = \lambda R_1(\psi) T_t R_2(\phi), \tag{1}$$

où $R_1$, $R_2$ sont des rotations et $T_t = \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix}$ avec $t > 1$, $\lambda > 0$, $\phi \in [0, \pi)$ et $\psi \in [0, 2\pi]$. De plus, cette décomposition est accompagnée d'une interprétation géométrique (voir Figure 3) où la longitude $\phi$ et la latitude $\theta = \arccos \frac{1}{t}$ caractérisent les angles de vue (ou inclinaison) de la caméra, $\psi$ est le paramètre de rotation de la caméra et $\lambda$ correspond au zoom de la caméra.

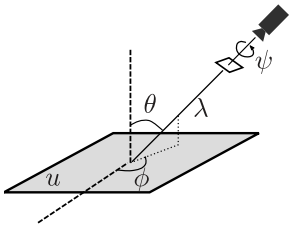


FIGURE 3 – Interprétation géométrique de l'équation (1).

Une image numérique $\mathbf{u}$ obtenue par n'importe quelle caméra à l'infini peut être écrite comme

$$\mathbf{u} = \mathbf{S}_1 \mathbb{G}_1 A \mathcal{T} u_0$$

où $\mathbf{S}_1$ est l'opérateur d'échantillonnage (sur une grille unitaire), $A$ est une transformation linéaire, $\mathcal{T}$ une translation du plan, $u_0$ une image continue et $\mathbb{G}_\delta$ est la convolution par un noyau Gaussien qui assure qu'aucun aliasing n'aura lieu lors de l'échantillonnage de pas $\delta$. Malheureusement, $\mathbb{G}_1$ et $A$ ne commutent pas lorsque la transformation $A$ contient un tilt $t \neq 1$ ou un zoom-arrière. Par conséquent, un simple warping $A(\mathbf{u}_0)$ de l'image frontale $\mathbf{u}_0 := \mathbf{S}_1 \mathbb{G}_1 u_0$ ne constitue pas une simulation optique affine correcte de $\mathbf{u}$. Comme indiqué dans [16, 2], la bonne façon de simuler un tilt $t$ dans la direction $x$ est :

$$\mathbf{u} \to \mathbf{S}_1 T_t^x \mathbb{G}^x_{\sqrt{t^2-1}} I \mathbf{u}, \tag{2}$$

où $I$ est l'interpolateur de Shannon-Whittaker et l'exposant $x$ indique que l'opérateur a lieu uniquement dans la direction $x$. Nous notons $\mathbb{T}_t^x := T_t^x \mathbb{G}^x_{\sqrt{t^2-1}} I$ (de même pour la direction $y$).

Lors de cette opération, il est clair qu'il y a une perte d'information due au flou. En effet, l'opérateur $\mathbb{T}_t^x$ n'est pas inversible. Ce qui signifie que, selon l'image $\mathbf{u}$, il peut ne pas y avoir de transformation optique $\mathbb{A}$ satisfaisant

$$\mathbb{A}(\mathbf{u}_1) = \mathbf{u}_2 \quad \text{ou} \quad \mathbf{u}_1 = \mathbb{A}(\mathbf{u}_2).$$

Considérons, par exemple, $\mathbf{u}_1 = \mathbb{T}_t^x \mathbf{u}$ et $\mathbf{u}_2 = \mathbb{T}_t^y \mathbf{u}$.
Dans ce contexte, nous concevons un schéma de génération de données qui, étant donné une image $\mathbf{u}$ et une paire de transformations affines aléatoires $\mathbb{A}_1$ et $\mathbb{A}_2$, simule des vues affines $\mathbf{u}_1 = \mathbb{A}_1(\mathbf{u})$ et $\mathbf{u}_2 = \mathbb{A}_2(\mathbf{u})$. Les deux transformations $\mathbb{A}_1, \mathbb{A}_2$ ont des angles de vues par rapport à $\mathbf{u}$ qui peuvent aller jusqu'à $75°$. Les images $\mathbf{u}$ utilisées proviennent de trois ensembles de données, fournis par MS-COCO [29], indépendants pour l'entraînement, la validation et les tests. Les paires de patchs de la même scène construits à partir de $\mathbf{u}_1$ et $\mathbf{u}_2$ définissent l'appartenance à la même *classe* et sont utilisées pour entraîner le réseau descripteur.

# 3 Descripteur et mise en correspondance

Inspiré par [23], notre réseau descripteur $\mathcal{D}$ est construit de manière à produire des vecteurs descripteurs similaires pour des paires de patchs appartenant à la même classe, et des vecteurs bien différents pour des paires de patchs de classes différentes. L'architecture du réseau est adaptée de [25], voir Figure 2. Il se compose de 4 blocs de deux couches convolutionnelles suivies chacune d'une normalisation par lots (Batch Normalisation) et d'activations ReLU. Entre chaque bloc, une couche "maxpool" est introduite. Un dropout spatial 2D avec une probabilité de $0.5$ est appliqué après la dernière couche convolutionnelle.
Ici, le dropout n'est pas utilisé pour éviter le surajustement (over-fitting) mais pour encourager le réseau descripteur à utiliser toutes les dimensions du vecteur. De plus, cela facilite le processus d'apprentissage : la fonction de perte dans la validation s'avère beaucoup plus stable que sans dropout.
L'approximation affine n'est valide que localement, ce qui suggère l'utilisation de petites tailles de patchs. En contrepartie, les petits patchs contiennent moins d'information. En guise de compromis, nous avons donc fixé la taille des patchs à $60 \times 60$, ce qui fournit un bon équilibre entre la localité et la richesse d'informations sur les points de vue.

### 3.1 Entraînement du réseau

Pendant l'entraînement, le réseau descripteur est immergé dans un réseau siamois, représenté Figure 4. Le réseau siamois se compose de deux sous-réseaux identiques reliés au sommet par une couche virtuelle qui calcule la perte "hinge" entre leurs deux sorties :

$$\lambda_p = \cos(\mathcal{D}(P_a), \mathcal{D}(P_p)),$$
$$\lambda_n = \cos(\mathcal{D}(P_a), \mathcal{D}(P_n)),$$

où les patchs $P_a, P_p$ appartiennent à la même classe alors que $P_n$ n'en fait pas partie. Pendant l'entraînement, nous simulons des changements de contraste aléatoires sur tous les patchs d'entrée. La fonction de perte "hinge", c'est-à-dire

$$L(\lambda_p, \lambda_n) \coloneqq \max(0, m + \lambda_n - \lambda_p),$$

est utilisée avec le paramètre $m$ fixé à 0.2 dans nos expériences.

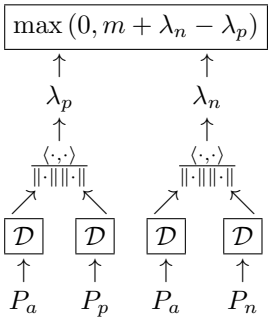


FIGURE 4 – Schéma du réseau siamois pour l'entraînement du réseau descripteur $\mathcal{D}$.

### 3.2 Le descripteur binaire

Une fois l'entraînement terminé, le réseau descripteur est déconnecté du réseau siamois et est censé produire des descripteurs qui capturent les propriétés affines invariantes des patchs d'entrée. Nous appelons cette description "BigAID" (6272 flottants). La Figure 5 montre les estimations de densité sur chaque dimension de BigAID. Remarquons l'implication de toutes les dimensions dans la description et la symétrie de toutes les densités autour de zéro.

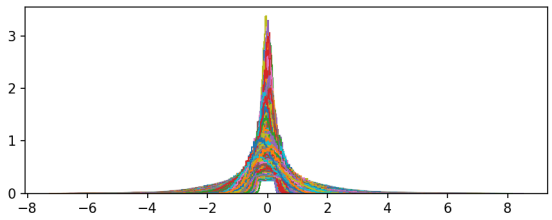

FIGURE 5 – Courbes de densité pour chaque dimension du descripteur BigAID (6272), calculées en utilisant $5 \cdot 10^4$ patchs aléatoires de l'ensemble de données de test.

Dans cet esprit, nous proposons un nouveau descripteur invariant affine, que nous appelons *AID* (6272 bits), qui ne conserve que l'information de signe du descripteur BigAID. Deux descripteurs AID $\mathbf{x}$ et $\mathbf{y}$ sont donc appariés via la mesure d'alignement de signe, c'est-à-dire

$$\sum_i \mathbb{1}_{\text{sign}(x_i)=\text{sign}(y_i)}.$$

Les estimations de la densité pour les mesures intra-classe et extra-classe sont présentées Figure 6 pour RootSIFT (128 flottants = 4096 bits) et pour nos descripteurs. Ces résultats suggèrent que pour les descripteurs BigAID et AID, un simple seuillage de leurs mesures respectives est suffisant pour distinguer les classes.

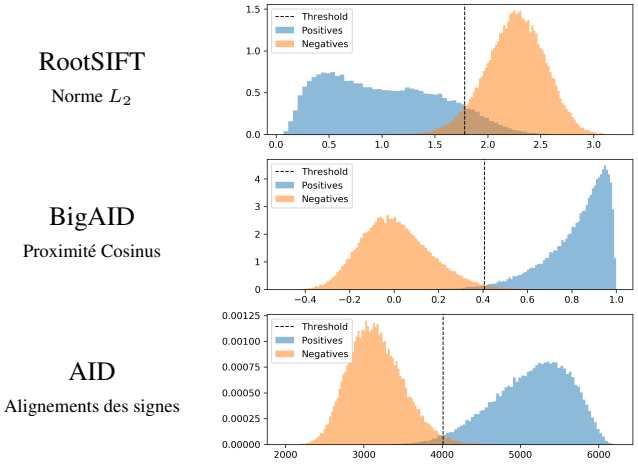


FIGURE 6 – Estimation de la densité pour les mesures dans les cas positifs et négatifs. Pour cela, $6 \cdot 10^5$ paires de patchs aléatoires intra-classe et extra-classe ont été utilisées. La ligne verticale représente le seuil minimisant les deux probabilités d'erreur : faux négatifs et faux positifs.

## 4 Résultats

Jusqu'à présent, le réseau descripteur $\mathcal{D}$ n'a vu que des patchs d'entrée simulés optiquement. La Figure 7 fournit un ensemble de données réaliste sous la forme de 5 paires d'images. Étant donné un ensemble fixe de points clés SIFT détectés dans ces images, les méthodes proposées sont comparées à RootSIFT dans la section Test I du Tableau 1. Le nombre de correspondances homographiques trouvées par ORSA [30] (RANSAC à validation a-contrario) montre la supériorité des descripteurs AID par rapport à RootSIFT. Le descripteur AID est plus compact et a une performance similaire à celle de BigAID. Pour ces raisons, nous préférons le descripteur AID et nous appelons *SIFT-AID* la méthode de comparaison résultant de sa combinaison avec les points clés SIFT.

La colonne A-RS (Test II) du tableau 1 indique le nombre de correspondances homographiques cohérentes pour Affine-RootSIFT. On remarque que SIFT-AID a des

| | Test I : avec points clés SIFT | | | | | Test II : avec points clés Affine-RootSIFT | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | # points clés par image | | Sans simulations affines | | | # points clés par image | | Avec simulations affines | Sans simulations affines | |
| | requête | cible | RS | BigAID | AID | requête | cible | A-RS | BigAID* | AID* |
| coke | 5443 | 5670 | 115 | 1316 | **1409** | 28609 | 31965 | 1395 | 5298 | **5346** |
| notredame | 2285 | 1235 | 14 | 282 | **295** | 11739 | 6444 | 48 | 590 | **731** |
| arc | 1384 | 1387 | 40 | **445** | 420 | 5719 | 4759 | 244 | 579 | **600** |
| graffiti | 1661 | 3117 | 0 | **182** | 172 | 14290 | 15225 | **613** | 502 | 516 |
| adam | 269 | 192 | 30 | 67 | **69** | 3647 | 2364 | 484 | 496 | **520** |

TABLE 1 – Test de performance de point de vue. RS, A-RS, BigAID et AID indiquent des correspondances homographiques cohérentes trouvées par ORSA pour RootSIFT, Affine-RootSIFT, BigAID et AID. Le ratio au deuxième plus proche voisin dans RootSIFT et Affine-RootSIFT a été fixé à 0,8. Les seuils pour BigAID et AID étaient respectivement de 0,4 et 4000. L'étoile (*) indique les points clés de l'oracle.

| | A-RootSIFT Norme $L_2$ | | SIFT-BigAID Prox. Cos. | | SIFT-AID Align. Signes | |
|---|---|---|---|---|---|---|
| | ET-D | ET-M | ET-D* | ET-M | ET-D* | ET-M |
| coke | 4.500 | 26.730 | 9.876 | 116.767 | 9.838 | 4.402 |
| notredame | 1.930 | 1.930 | 3.272 | 10.829 | 3.177 | 0.540 |
| arc | 1.520 | 0.670 | 2.581 | 7.372 | 2.465 | 0.389 |
| graf | 2.790 | 6.180 | 4.441 | 20.027 | 4.369 | 0.895 |
| adam | 1.210 | 0.230 | 0.601 | 0.241 | 0.525 | 0.048 |

TABLE 2 – Performance en temps pour Affine-RootSIFT, SIFT-BigAID et SIFT-AID. Temps écoulé (en secondes) pour la construction des descripteurs (ET-D) et leur appariement (ET-M) ; L'étoile (*) indique le temps GPU.

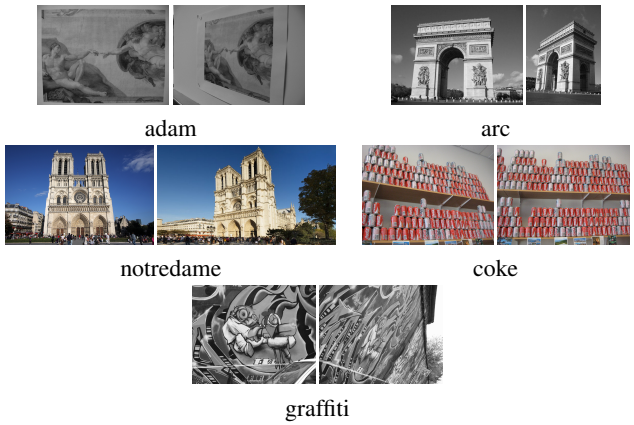


FIGURE 7 – Ensemble de données du défi des points de vue.

performances comparables sans utiliser de simulations de points de vue. Mais dans certains cas, il y a moins de correspondances, comme pour la paire *adam*. Comme indiqué dans [2], Affine-RootSIFT a environ 7 fois plus de points clés que SIFT. Certains de ces points clés proviennent exclusivement de versions simulées des images d'entrée, c'est-à-dire qu'ils n'appartiennent pas à la pyramide gaussienne des images d'entrée. Pour tester plus en détail les descripteurs AID, nous définissons un oracle qui donne des points clés précis dans la pyramide gaussienne d'origine, en se rapprochant au mieux de chaque point clé des premières étapes d'Affine-RootSIFT. Les points clés fournis par cet oracle sont les meilleurs choix possibles qui auraient pu être trouvés lors des premières étapes de SIFT. Le Tableau 1 (Test II) montre également le nombre de correspondances homographiques cohérentes pour les descripteurs AID + oracle. Cette expérience révèle que AID et BigAID auraient suffi à identifier presque toutes les correspondances Affine-RootSIFT, à condition que les points clés appropriés aient été correctement repérés par les premières étapes de SIFT. Dans le cas de la paire *graffiti*, la plupart des correspondances manquantes pour les descripteurs AID impliquent des angles de point de vue proches de 75°, l'angle de point de vue maximal présent dans l'ensemble de données pour l'entraînement.

Enfin, le tableau 2 indique le temps passé par SIFT-AID et Affine-RootSIFT à construire des descripteurs et à les apparier [1] (codes non optimisés). Dans l'ensemble, la méthode SIFT-AID permet d'obtenir des résultats en moins de temps qu'Affine-RootSIFT.

# 5 Conclusions

Nous avons proposé un descripteur de patchs basé sur un réseau de neurones capturant l'invariance affine. Nos expériences montrent que la méthode SIFT-AID atteint une performance comparable à celle d'Affine-RootSIFT sans

1. Hardware : (CPU) Intel(R) Core(TM) i7-6700HQ 2.60GHz ; (GPU) NVIDIA Corporation GM204GLM[Quadro M5000M].

qu'il soit nécessaire d'utiliser des simulations de points de vue. La plupart des correspondances manquantes sont dues aux échecs de l'étape de détection des points clés de SIFT ; des efforts supplémentaires sont nécessaires pour améliorer cette étape. La robustesse du point de vue de la méthode proposée pourrait être encore renforcée par des techniques de simulation affine similaires à celles de [16, 2]. Cette extension fera l'objet de travaux futurs. Enfin, l'architecture du réseau descripteur pourrait être optimisée pour améliorer les performances.

**Reproductibilité :** Le code source de SIFT-AID est disponible sur `https://rdguez-mariano.github.io/pages/sift-aid`

# Références


[1] G. Yu and J.-M. Morel, "ASIFT : An Algorithm for Fully Affine Invariant Comparison," *IPOL*, vol. 1, pp. 1–28, 2011.

[2] M. Rodriguez, J. Delon, and J.-M. Morel, "Covering the space of tilts. application to affine invariant image comparison," *SIIMS*, vol. 11, no. 2, pp. 1230–1267, 2018.

[3] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[4] T. Lindeberg and J. Garding, "Shape-adapted smoothing in estimation of 3-D depth cues from affine distortions of local 2-D brightness structure," *ECCV*, pp. 389–400, 1994.

[5] A. Baumberg, "Reliable feature matching across widely separated views," *CVPR*, vol. 1, pp. 774–781, 2000.

[6] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," *ECCV*, vol. 1, pp. 128–142, 2002.

[7] K. Mikolajczyk and C. Schmid, "Scale and Affine Invariant Interest Point Detectors," *IJCV*, vol. 60, no. 1, pp. 63–86, 2004.

[8] T. Tuytelaars, L. Van Gool, and Others, "Content-based image retrieval based on local affinely invariant regions," *Int. Conf. on Visual Information Systems*, pp. 493–500, 1999.

[9] T. Tuytelaars and L. Van Gool, "Matching Widely Separated Views Based on Affine Invariant Regions," *IJCV*, vol. 59, no. 1, pp. 61–85, 2004.

[10] T. Tuytelaars and L. Van Gool, "Wide baseline stereo matching based on local, affinely invariant regions," *BMVC*, pp. 412–425, 2000.

[11] T. Kadir, A. Zisserman, and M. Brady, "An Affine Invariant Salient Region Detector," in *ECCV*, 2004, pp. 228–241.

[12] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *IVC*, vol. 22, no. 10, pp. 761–767, 2004.

[13] P. Musé, F. Sur, F. Cao, and Y. Gousseau, "Unsupervised thresholds for shape matching," *ICIP*, 2003.

[14] P. Musé, F. Sur, F. Cao, Y. Gousseau, and J. M. Morel, "An A Contrario Decision Method for Shape Element Recognition," *IJCV*, vol. 69, no. 3, pp. 295–315, 2006.

[15] F. Cao, J.-L. Lisani, J.-M. Morel, P. Musé, and F. Sur, *A Theory of Shape Identification*, Springer Verlag, 2008.

[16] J. M. Morel and G. Yu, "ASIFT : A new framework for fully affine invariant image comparison," *SIIMS*, vol. 2, no. 2, pp. 438–469, 2009.

[17] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*, 2012, pp. 2911–2918.

[18] Y. Pang, W. Li, Y. Yuan, and J. Pan, "Fully affine invariant SURF for image matching," *Neurocomputing*, vol. 85, pp. 6–10, 2012.

[19] D. Mishkin, J. Matas, and M. Perdoch, "MODS : Fast and robust method for two-view matching.," *CVIU*, vol. 141, pp. 81–93, 2015.

[20] M. Rodriguez and R. Grompone von Gioi, "Affine invariant image comparison under repetitive structures," in *ICIP*, Oct 2018, pp. 1203–1207.

[21] M. Rodriguez, J. Delon, and J.-M. Morel, "Fast affine invariant image matching," *IPOL*, vol. 8, pp. 251–281, 2018.

[22] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *CVPR*, 2015, pp. 4353–4361.

[23] J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *JMLR*, vol. 17, no. 1-32, pp. 2, 2016.

[24] I. Rocco, R. Arandjelovic, and J. Sivic, "Convolutional neural network architecture for geometric matching," *TPAMI*, 2018.

[25] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Deep image homography estimation," *arXiv preprint arXiv :1606.03798*, 2016.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv :1409.1556*, 2014.

[27] I. Rey-Otero and M. Delbracio, "Anatomy of the SIFT method," *IPOL*, vol. 4, pp. 370–396, 2014.

[28] J. M. Morel and G. Yu, "Is SIFT scale invariant ?," *Inv. Problems and Imaging*, vol. 5, no. 1, pp. 115–136, 2011.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco : Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755.

[30] L. Moisan, P. Moulon, and P. Monasse, "Automatic Homographic Registration of a Pair of Images, with A Contrario Elimination of Outliers," *IPOL*, vol. 2, pp. 56–73, 2012.