



HAL
open science

Découverte et exploitation d'objets visuels fréquents dans des collections multimédias

Pierre Letessier

► **To cite this version:**

Pierre Letessier. Découverte et exploitation d'objets visuels fréquents dans des collections multimédias. Multimédia [cs.MM]. Telecom ParisTech, 2013. Français. NNT: . tel-00912992v1

HAL Id: tel-00912992

<https://theses.hal.science/tel-00912992v1>

Submitted on 3 Dec 2013 (v1), last revised 6 Jan 2014 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ParisTech

INSTITUT DES SCIENCES ET TECHNOLOGIE
PARIS INSTITUTE OF TECHNOLOGY

TELECOM
ParisTech



2013-ENST-00xx



EDITE - ED 130

Doctorat ParisTech

THÈSE

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Signal et Images »

présentée et soutenue publiquement par

Pierre LETESSIER

le ?/?/?/?/?/?

Découverte et exploitation d'objets visuels fréquents dans des collections multimédias

Directeur de thèse : **Nozha BOUJEMAA**

Co-encadrement de la thèse : **Olivier BUISSON, Alexis JOLY**

Jury

M. Georges QUÉNOT, Docteur (HDR), groupe MRIM, LIG

M. Stéphane MARCHAND-MAILLET, Professeur, Viper group, University of Geneva

M. Patrick PÉREZ, Docteur (HDR), Unité de recherche, Technicolor

Mme Nozha BOUJEMAA, Docteur (HDR), INRIA Saclay Ile de France

M. Olivier BUISSON, Docteur, groupe de recherche audiovisuelle, Ina

M. Alexis JOLY, Docteur, équipe ZENITH (LIRMM), INRIA Sophia-Antipolis

Rapporteur

Rapporteur

Examineur

Examineur

Examineur

Examineur

TELECOM ParisTech

école de l'Institut Télécom - membre de ParisTech

THÈSE

Table des matières

3	Remerciements	13
4	1 Introduction	15
5	1.1 Motivations	15
6	1.2 Problématique	16
7	1.3 Contributions	17
8	1.4 Organisation de la thèse	18
9	I Etat de l'art	19
10	2 État de l'art des méthodes de recherche d'objets visuels	21
11	2.1 Introduction	21
12	2.2 Description d'images	21
13	2.2.1 Descripteurs bas niveau	21
14	2.2.1.1 Descripteurs globaux	23
15	2.2.1.2 Descripteurs locaux	23
16	2.2.1.2.1 Descripteurs basés histogrammes	24
17	2.2.1.2.2 Descripteurs basés sur les différentielles et les moments	25
18	2.2.1.2.3 Descripteurs basés sur les fréquences spatiales	25
19	2.2.1.2.4 Analyse	25
20	2.2.2 Représentation des images	26
21	2.2.2.1 Représentation totale (locale)	26
22	2.2.2.2 Sac de mots visuels	26
23	2.2.2.3 Fisher vectors	27
24	2.3 Partitionnement de l'espace visuel	27
25	2.3.1 Partitionnement indépendant de la distribution des données	28
26	2.3.1.1 Partitionnement basé sur des structures déterministes	28
27	2.3.1.1.1 Treillis	28
28	2.3.1.1.2 Courbes remplissant l'espace	28
29	2.3.1.2 Familles de hachage basées sur des transformations aléatoires	29
30	2.3.1.2.1 Locality Sensitive Hashing	29
31	2.3.2 Partitionnement dépendant de la distribution des données	30
32	2.3.2.1 Partitionnement hiérarchique	31
33	2.3.2.2 Familles de hachage dépendant des données	32
34	2.3.2.2.1 Restricted Boltzmann Machine	32
35	2.3.2.2.2 Spectral Hashing	32
36	2.3.2.2.3 Spherical Hashing	32
37	2.3.2.2.4 Kernelized Locality Sensitive Hashing	33
38	2.3.2.2.5 Product Quantization	33

39	2.3.2.2.6	Random Maximum Margin Hashing	33
40	2.3.2.3	Partitionnement basé regroupement	34
41	2.3.2.3.1	K-Means	34
42	2.3.2.3.2	Mean-shift	35
43	2.3.3	Le problème du partitionnement « brutal »	35
44	2.4	Structures d'indexation et recherche par similarité	35
45	2.4.1	Les structures	36
46	2.4.1.1	Tables de hachage et listes inversées	36
47	2.4.1.2	Les structures hiérarchiques	36
48	2.4.2	Recherche dans une structure d'index	36
49	2.4.2.1	Recherche par accès simple	37
50	2.4.2.2	Recherche par accès simple dans de multiples structures	37
51	2.4.2.3	Le soft-assignment	38
52	2.4.2.4	Recherche par accès multiples	38
53	2.4.2.5	Recherche avec calcul de distances asymétriques	39
54	2.4.2.6	Analyse des méthodes de recherche	39
55	2.5	Vérification de la cohérence géométrique	39
56	2.5.1	Contraintes géométriques faibles	40
57	2.5.2	Méthodes basées sur l'estimation de la transformation géométrique	40
58	2.5.2.1	Estimateurs robustes des moindres carrés	40
59	2.5.2.2	Transformée de Hough	41
60	2.5.2.3	RANSAC	41
61	2.5.3	Méthodes basées cosegmentation	42
62	2.6	Extension de requête	42
63	2.7	Recherche d'objets visuels à grande échelle	43
64	3	État de l'art des méthodes de découverte et fouille d'objets visuels	47
65	3.1	Méthodes de fouille d'objets visuels par regroupement de motifs visuels	48
66	3.2	Méthodes de construction de graphes d'appariement d'images contenant des objets similaires	49
67	3.3	Méthodes de découverte de concepts visuels fréquents par extraction de sujets latents	49
68	3.4	Approximation de graphes d'appariement d'instances d'objets par hachage	50
69			
70	II	Contributions	53
71	4	Fouille d'objets visuels fréquents par requêtage aléatoire	55
72	4.1	Définition formelle des problèmes de fouille et de découverte d'objets visuels fréquents	55
73	4.1.1	Notations	55
74	4.1.2	Formalisation des problèmes de découverte et fouille d'objets fréquents	56
75	4.2	Recherche itérative par échantillonnage pondéré adaptatif (RANSAS)	57
76	4.2.1	Échantillonnage pondéré adaptatif	58
77	4.2.2	Recherche précise d'une région locale	60
78	4.2.3	Décision	61
79	4.3	Modèle de coût de l'algorithme RANSAS	62
80	4.4	Conversion des poids en fonction de probabilité	64
81	4.5	Choix et implémentation de la méthode de recherche précise	65
82	4.5.1	Description des images	65
83	4.5.2	Partitionnement des descripteurs SIFT	65
84	4.5.3	Indexation et recherche des descripteurs SIFT	66
85	4.5.4	Vérification de la cohérence géométrique	67
86	4.5.5	Extension de requêtes	69

87	5	Calcul des scores de vraisemblance par des méthodes de l'état de l'art	71
88	5.1	Calcul de scores de vraisemblance spécifiques à un type d'objet	71
89	5.1.1	Scores de vraisemblance basés sur la position dans les images	71
90	5.1.2	Scores de vraisemblance basés sur la détection de visages	72
91	5.2	Calcul de scores de vraisemblance généralistes	74
92	5.2.1	Calcul de scores de vraisemblance par des heuristiques simples	74
93	5.2.1.1	Scores de vraisemblance basés sur la densité spatiale des points d'intérêts	74
94	5.2.1.2	Scores de vraisemblance basés sur la variabilité dans l'espace des descripteurs	75
95			
96	5.2.2	Calcul du score de vraisemblance par comptage des appariements	75
97	5.2.3	Calcul du score de vraisemblance par Geometric min-Hashing	76
98	5.3	Expérimentations	77
99	5.3.1	Protocole d'évaluation	77
100	5.3.2	Évaluation de Geometric min-Hashing comme score de vraisemblance	78
101	5.3.2.1	Influence du nombre de <i>sketches</i>	78
102	5.3.2.2	Influence de la taille du vocabulaire	78
103	5.3.3	Comparaison des performances des scores de vraisemblance généralistes	80
104	6	Calcul des scores de vraisemblance par hachage visuel et ajout de contraintes géométriques faibles	81
105	6.1	Hachage visuel et filtrage des appariements candidats	82
106	6.1.1	Construction de l'index visuel	82
107	6.1.2	Filtrage visuel des appariements candidats	83
108	6.1.2.1	Filtrage sur la fréquence de collision intra-image	83
109	6.1.2.2	Filtrage sur la fréquence de collision inter-tables	83
110	6.1.2.3	Filtrage par les K plus proches voisins	84
111	6.2	Extraction et hachage d'attributs de géométrie faible	84
112	6.2.1	Création de vecteurs de géométrie faible	85
113	6.2.2	Hachage des attributs de géométrie faible	85
114	6.3	Calcul des scores de vraisemblance par accumulation des collisions	86
115	6.3.1	Vote dans les accumulateurs	86
116	6.3.2	Calcul des scores individuels	87
117	6.4	Solution alternative : ajout de contraintes géométriquement faibles à GmH	87
118	6.5	Expérimentations	87
119	6.5.1	Étude paramétrique	88
120	6.5.1.1	Paramétrage du hachage des descripteurs visuels	88
121	6.5.1.2	Longueur des clés de hachage des attributs de géométrie faible	89
122	6.5.1.3	Nombre d'accumulateurs	90
123	6.5.2	Apport des attributs de géométrie faible	90
124	6.5.3	Comparaison de notre méthode à GmH	92
125	III	Expérimentations et Applications	93
126	7	Découverte et fouille de logos	95
127	7.1	Le corpus FlickrBelgaLogos	95
128	7.2	Vitesse de convergence en fonction des différents scores de vraisemblance	98
129	7.2.1	Protocole d'expérimentations	98
130	7.2.2	Évaluation	99
131	7.3	Classification des instances par un algorithme de regroupement	104
132	7.3.1	L'algorithme MCL	105
133	7.3.2	Protocole d'évaluation	107
134	7.3.3	Évaluation	108

TABLE DES MATIÈRES

135	8 Découverte d'événements saillants dans des médias d'actualité	111
136	8.1 Détection d'évènements télévisuels	111
137	8.1.1 Création d'une grande collection de journaux télévisés	111
138	8.1.2 Approche transmédia pour la découverte d'objets informatifs	112
139	8.2 Découverte d'évènements transmédiés	113
140	9 Suggestion de requêtes visuelles	119
141	9.1 Principe de la suggestion de requêtes visuelles	120
142	9.2 Algorithme de <i>clustering</i> de graphe biparti	121
143	9.3 Scénarios de suggestion de requêtes visuelles	123
144	10 Conclusion	129
145	10.1 Synthèse des contributions	129
146	10.2 Analyse et bilan	130
147	10.3 Perspectives	131

Table des figures

149	1.1	Plusieurs instances d'objets de la classe « maison »	16
150	1.2	Plusieurs instances de l'objet « la maison blanche »	17
151	2.1	Architecture générale d'un système de recherche d'images par le contenu	22
152	2.2	Exemple d'un treillis hexagonal	28
153	2.3	Les six premières itérations d'une courbe de Hilbert (certains droits réservés, licence CC-By-SA, Braindrain0000)	29
154			
155	2.4	Exemple de hachage en 2D avec projections aléatoires et 4 bits (4 hyperplans)	30
156	2.5	Exemple de partitionnement avec un <i>Kd-tree</i>	31
157	2.6	Exemple de hachage en 2D avec <i>spherical hashing</i> et 4 bits (4 hypersphères)	33
158	2.7	Exemple de hachage en 2D avec RMMH	34
159	2.8	Exemple de partitionnement par <i>K-Means</i>	34
160	2.9	Illustration du partitionnement « brutal ». Les points B et C sont proches dans l'espace mais ne sont pas dans la même partie de l'espace.	36
161			
162	4.1	Schéma récapitulatif du fonctionnement de l'algorithme RANSAS	59
163	4.2	Comparaison de la surface couverte (en pourcentage du nombre de descripteurs) en fonction du nombre d'itérations T de RANSAS, et du type d'échantillonnage (simple ou adaptatif).	61
164			
165	4.3	Comparaison des tailles d'instances des bases BelgaLogos et Oxford Buildings	63
166	4.4	Histogramme des poids z_0 (en échelle logarithmique)	65
167	4.5	Compromis entre qualité (mesurée par la mAP sur Oxford Buildings) et temps de calcul	67
168	5.1	Score de vraisemblance basé sur la position centrale dans les images. Plus la couleur est chaude, et plus on est proche du centre de l'image	72
169			
170	5.2	Apport du score de vraisemblance « position centrale » par rapport à l'uniforme	73
171	5.3	Score de vraisemblance basé sur la détection de visages. Plus la couleur est chaude et plus on est proche d'un visage détecté.	74
172			
173	5.4	Densité spatiale des points d'intérêts. Plus la couleur est chaude et plus la densité est forte.	75
174	5.5	Variabilité dans l'espace des descripteurs. Plus la couleur est chaude et plus la variabilité est importante.	76
175			
176	5.6	Influence du nombre de <i>sketches</i> k sur la précision et le rappel. Un vocabulaire de 1 million de mots a été utilisé.	79
177			
178	5.7	Influence de la taille du vocabulaire (Voc) sur la précision et le rappel. 100 000 <i>sketches</i> ont été générés.	79
179			
180	5.8	Comparaison des performances des scores de vraisemblance générés par Geometric min-Hashing, par appariements de descripteurs (mots visuels), par variabilité des descripteurs, ou encore par densité spatiale des descripteurs. La courbe GmH a été obtenue avec un vocabulaire d'un million de mots visuels et 100 000 <i>sketches</i>	80
181			
182			
183			
184	6.1	Réglage du nombre de tables de hachage L	88

TABLE DES FIGURES

185	6.2 Effets de la longueur des clés de hachage de la géométrie faible	89
186	6.3 Influence du nombre d'accumulateurs	90
187	6.4 Contribution des attributs de Géométrie Faible ($\theta, \sigma, \chi, \psi$) sur la précision et le rappel, pour la méthode à base de hachage visuel.	91
188		
189	6.5 Contribution de la géométrie faible sur les méthodes de calcul de score de vraisemblance.	92
190	7.1 Un échantillon des 10 000 images de la base FlickrBelgaLogos	98
191	7.2 Rappel et de la précision instance en fonction du nombre d'itérations de RANSAS	100
192	7.3 Rappel objet en fonction du nombre d'itérations de RANSAS	101
193	7.4 Les deux premières requêtes issues de l'échantillonnage pondéré par le score de vraisemblance HV + GF (L=64), et quatre de leurs résultats de recherche	104
194		
195	7.5 Pureté moyenne (AvgPurity) en fonction du nombre de <i>clusters</i> obtenus en faisant varier le paramètre d'inflation de MCL entre 1.01 et 6, pour différents scores de vraisemblance, et différents nombre d'itérations de RANSAS (1K, 10K et 100K).	109
196		
197		
198	8.1 Chaînes de télévision incluses dans la collection FrenchTVFrames	112
199	8.2 Un <i>cluster</i> filtré par le tri sur le score de diversité (en haut), et les trois meilleurs <i>clusters</i> (événements transmédias) de la première semaine de septembre 2011, représentant les primaires socialistes, l'US Open, et le scandale du Médiateur.	116
200		
201	8.3 Meilleurs événements transmédias détectés pour les semaines du 12 septembre, 3 octobre, 17 octobre et 14 novembre 2011	117
202		
203		
204	9.1 Illustration de la méthode proposée pour suggérer des requêtes visuelles dans l'image I_4 . Les germes S_2, S_5 et S_9 appartiennent à un premier <i>cluster</i> (en bleu), tandis que S_3, S_6 et S_8 appartiennent à un second (en vert).	122
205		
206	9.2 Génération des fenêtres rectangulaires pour la suggestion d'objets visuels au survol de la souris	124
207		
208	9.3 Génération des miniatures par recadrage pour la suggestion d'objets visuels en complément d'une requête textuelle	125
209		
210	9.4 Capture d'écran de l'interface de suggestion d'objets visuels au survol de la souris	126
211		
212	9.5 Capture d'écran de l'interface de suggestion d'objets visuels comme facétisation d'une requête textuelle	127
213		

Liste des tableaux

215	2.1	Comparaison des systèmes de recherche d'images par le contenu sur Oxford Buildings et BelgaLogos.	45
216			
217	4.1	Comparaison des temps de calcul (en secondes) de AKM et RMMH.	66
218	5.1	Nombre de détections par catégorie	73
219	6.1	Réglage du paramètre $\hat{\tau}$ optimum théorique en fonction de L , et τ utilisé en pratique.	88
220	7.1	Logos présents dans la vérité terrain de la base BelgaLogos	96
221	7.2	Temps et accélération (en vert) obtenus avec différents scores de vraisemblance.	103
222	8.1	Meilleurs évènements découverts (plus gros scores S), par catégorie (petit, moyen, gros), dans les 2 051 heures de télévision Française.	114
223			



ELSEVIER

<http://thecostofknowledge.com>

Par cette occasion, je souhaite témoigner de mon refus de participer à l'enrichissement de certains éditeurs scientifiques comme Elsevier, Springer Verlag, Association for Computing Machinery, IEEE Publications, ou tout autre éditeur ne pratiquant pas des tarifs à la mesure de ses frais de fonctionnement.

Remerciements

224

225 Je tiens tout d'abord à remercier chaleureusement Alexis Joly et Olivier Buisson, qui m'ont encadré
226 pendant ces trois dernières années. J'ai énormément appris à leurs cotés, et je leur en suis extrêmement
227 reconnaissant.

228

229 J'aimerais également témoigner toute ma gratitude à Nozha Boujemaa, qui a accepté de diriger ma
230 thèse et qui m'a accueilli dans l'équipe IMEDIA de l'INRIA Rocquencourt.

231

232 Merci à Marie-Luce Viaud, Agnès Saulnier, Benjamin Renoust et Jean-Étienne Noiré pour les bons
233 moments passés à l'Ina.

234

235 Et merci à ceux qui m'ont aidé et conseillé au cours de ma thèse, je veux parler de Félicien Vallet,
236 Nicolas Hervé, Louis Laborelli et beaucoup d'autres qui je l'espère ne m'en voudront pas.

237

238 Merci à Amel Hamzaoui et Julien Champ avec qui il a été un réel plaisir de collaborer.

239

240 J'aimerais remercier Gilbert Garnier, Isabelle Bloch, Séverine Dubuisson et Najib Gadi, pour m'avoir
241 aidé et poussé à aller plus loin tout au long de mes stages et études.

242

243 Enfin, je ne peux pas finir ces remerciements sans penser à mes parents et à mon frère, qui m'ont
244 soutenu depuis le début, et qui m'ont donné la chance de pouvoir m'accomplir.

245

246 Merci à tous ceux qui auront le courage de lire cette thèse en entier, et en particulier mes rapporteurs !

Chapitre 1

Introduction

1.1 Motivations

L'Ina¹ gère actuellement environ 5 millions d'heures de télévision et radio, et plus d'un million de photographies. Chaque année s'y ajoutent 800 mille heures de plus, soit plus de 2 000 heures par jour. Flickr détient environ 5 milliards de photographies, Facebook plus de 10 milliards, avec une croissance de cent millions chaque mois. Environ 35 heures de vidéos sont envoyées sur YouTube chaque minute.

L'accroissement du nombre de données visuelles à archiver impose des contraintes de productivité nouvelles sur la documentation. Il devient ainsi de plus en plus difficile d'annoter manuellement les documents archivés, de par le volume de données, mais également par l'évolution de la finesse de description. Plus la masse d'informations augmente, et plus le nombre d'occurrences d'un objet visuel donné augmente en conséquence. Un élément jusqu'alors inconnu, devient alors potentiellement remarquable, à partir du moment où ce nombre d'apparitions dépasse un certain seuil.

A l'Ina, plus d'une centaine de documentalistes travaillent chaque jour à l'annotation manuelle des programmes à archiver. L'une de leurs activités consiste, pour chaque programme télévisé, à fournir des mots clés (personnalités, lieux, thèmes) issus d'un thésaurus, ainsi que des images représentatives du programme. Les plateformes d'hébergement de vidéos (comme YouTube, Dailymotion, etc.) utilisent généralement un système de métadonnées générées par les utilisateurs eux-mêmes. Le développement d'outils automatiques et semi-automatiques pour assister ce travail d'annotation est désormais indispensable afin d'exploiter au mieux la très grande quantité d'informations disponibles.

L'un des systèmes de fouille d'images ayant acquis une certaine notoriété auprès du public est le logiciel Picasa², permettant la détection et la reconnaissance de visages. Bien que remplissant parfaitement son rôle, l'application reste limitée aux visages, et ne gère pas la découverte de lieux fréquents (façades de bâtiments, décors, scènes, etc.) ou d'objets d'intérêts divers, autant d'informations qui pourraient permettre aux utilisateurs de mieux naviguer dans leurs milliers de photographies.

1. Institut National de l'Audiovisuel : <http://www.ina.fr>

2. <http://picasa.google.com/>

274

275 Outre les photographes et les fournisseurs de contenu visuel, la fouille d'objets visuels peut également
 276 intéresser les journalistes et les sociologues désireux d'analyser le contenu visuel des médias, afin de
 277 détecter les événements médiatiques, de mesurer le suivi de l'information, etc.

278 1.2 Problématique

279 Ces dernières années, sont apparues des techniques d'analyse et d'indexation de contenu visuel ou
 280 audio, permettant d'effectuer des recherches et de la fouille de données dans de grands volumes de
 281 contenu vidéo.

282 L'objectif principal de cette thèse est la découverte d'objets visuels fréquents dans de grandes
 283 collections multimédias (images ou vidéos). Comme dans de nombreux domaines (finance, génétique,
 284 etc.), il s'agit d'extraire une connaissance de manière automatique ou semi-automatique en utilisant la
 285 fréquence d'apparition d'un objet au sein d'un corpus comme critère de pertinence. Dans le cas visuel,
 286 le problème est différent de la fouille de données classique (ADN, textuel, etc.) puisque les instances
 287 d'apparition d'un même objet ne constituent pas des entités identiques mais doivent être appariées.
 288 Cette difficulté explique également pourquoi nous nous focalisons sur la découverte des objets rigides
 289 (logos, objets manufacturés, décors, bâtiments, etc.), et non des classes (catégories) d'objets de plus
 290 haut niveau sémantique (maison, voiture, chien, etc.). Tout comme en programmation orientée objet,
 291 une classe est une catégorie définissant un concept (pouvant apparaître sous différentes formes), tandis
 292 qu'une instance d'objet est une représentation de cet objet. Ainsi, on distingue par exemple la classe
 293 « maison » (voir figure 1.1), de l'instance « la maison blanche » (voir figure 1.2 page suivante).



(a) certains droits réservés, licence CCA 2.0, Jean-Pol GRANDMONT (b) certains droits réservés, licence CC-BY-SA-3.0, Ji-Elle (c) certains droits réservés, licence CC-BY-SA-3.0, Spedona

FIGURE 1.1 – Plusieurs instances d'objets de la classe « maison »

294 Bien que les techniques de recherche d'objets rigides aient atteint une certaine maturité, le
 295 problème de la découverte non supervisée d'instances d'objets dans des grandes collections d'images
 296 est à l'heure actuelle encore difficile. D'une part parce que les méthodes actuelles ne sont pas assez
 297 efficaces et passent difficilement à l'échelle. D'autre part parce que le rappel et la précision sont encore
 298 insuffisants pour de nombreux objets. Particulièrement ceux ayant une taille très restreinte par rapport
 299 à l'information visuelle contextuelle qui peut être très riche (par exemple le logo d'un parti politique

300 apparaissant ponctuellement dans un sujet de journal télévisé).



(a) certains droits réservés, licence CC-BY-SA-3.0, Matt Wade

(b) certains droits réservés, licence CC-BY-SA-3.0, Zach Rudisin

(c)

FIGURE 1.2 – Plusieurs instances de l'objet « la maison blanche »

301 1.3 Contributions

302 Une première contribution de la thèse est de fournir un formalisme aux problèmes de découverte et
 303 de fouille d'instances d'objets visuels fréquents. Ces deux problèmes sont en effet définis de manière
 304 très confuse dans les quelques travaux récents de la littérature les abordant. Cette modélisation nous a
 305 permis entre autres choses de mettre en évidence le lien étroit qui existe entre la taille des objets à
 306 découvrir et la complexité du problème à traiter.

307

308 La deuxième contribution de la thèse est une méthode générique de résolution de ces deux types de
 309 problèmes, reposant d'une part sur un processus itératif d'échantillonnage d'objets candidats, et d'autre
 310 part sur une méthode efficace d'appariement d'objets rigides à large échelle. L'idée est de considérer
 311 l'étape de recherche d'instances proprement dite comme une simple boîte noire à laquelle il s'agit de
 312 soumettre des régions d'images ayant une probabilité élevée d'appartenir à un objet fréquent de la
 313 base. Une première approche étudiée dans la thèse consiste à simplement considérer que toutes les
 314 régions d'images de la base sont équiprobables, avec comme idée conductrice que les objets les plus
 315 instanciés sont ceux qui auront la couverture spatiale la plus grande et donc la probabilité la plus élevée
 316 d'être échantillonnés. En généralisant cette notion de couverture à celle plus générique de couverture
 317 probabiliste, il est alors possible de modéliser la complexité de notre méthode pour tout score de
 318 vraisemblance donné en entrée, et de montrer ainsi l'importance de cette étape.

319

320 La troisième contribution de la thèse s'attache précisément à construire un score de vraisemblance
 321 s'approchant au mieux de la distribution parfaite, tout en restant *scalable* et efficace. Cette dernière
 322 repose sur une approche originale de hachage à deux niveaux, permettant de générer efficacement un
 323 ensemble d'appariements visuels dans un premier temps, et d'évaluer ensuite leur pertinence en fonction
 324 de contraintes géométriques faibles. Les expérimentations montrent que contrairement aux méthodes de
 325 l'état de l'art notre approche permet de découvrir efficacement des objets de très petite taille dans des

326 millions d'images.

327

328 Pour finir, plusieurs scénarios d'exploitation des graphes d'appariement visuel produits par notre
329 méthode sont proposés et expérimentés. Ceci inclut la détection d'évènements médiatiques transmédia
330 et la suggestion de requêtes visuelles.

331 **1.4 Organisation de la thèse**

332 La thèse est organisée en trois parties. La première a pour objectif de passer en revue les différentes
333 méthodes de recherche (cf. chapitre 2 page 21) et découverte d'objets visuels (cf. chapitre 3 page 47) de
334 l'état de l'art. On s'attachera en particulier dans le chapitre 2 à détailler les différentes étapes composant
335 la recherche d'objets, soit la description, le partitionnement des vecteurs, l'indexation et la recherche,
336 ainsi que la vérification de la cohérence géométrique et l'extension de requêtes.

337

338 Dans la deuxième partie, on présentera les contributions principales de cette thèse, à travers trois
339 chapitres. Le chapitre 4 page 55 commence par formaliser les problèmes de découverte et fouille
340 d'instances, puis propose une architecture générale pour résoudre ces problèmes. Le chapitre 5 page 71 a
341 pour objectif de proposer différentes approches de calcul de scores de vraisemblance issues de méthodes
342 état de l'art, alors que le chapitre 6 page 81 expose notre nouvelle méthode de calcul de scores basée sur
343 le hachage des descripteurs visuels et des attributs de géométrie faible. Ces deux chapitres offrent tous
344 deux des expérimentations quantitatives afin de comparer les performances de chaque méthode.

345

346 Dans la troisième et dernière partie, nous proposons différentes expérimentations et contextes
347 applicatifs. Le chapitre 7 page 95 s'attache à la découverte de logos, et permet d'évaluer quantitativement
348 l'ensemble de notre système de découverte d'objets visuels sur une collection d'images adaptée au
349 problème. Le chapitre 8 page 111 montre les résultats de notre système de découverte dans une
350 collection de plus de 2 000 heures de journaux télévisés, avec un filtrage transmédia permettant de
351 détecter les principaux événements médiatiques. Enfin, le chapitre 9 page 119 décrit notre proposition de
352 système de suggestion de requêtes visuelles à partir des résultats de fouille d'objets.

353

Première partie

354

Etat de l'art

Chapitre 2

État de l'art des méthodes de recherche d'objets visuels

2.1 Introduction

Il est important de bien distinguer les différences entre les systèmes de recherche d'objets visuels et d'autres systèmes comme ceux concernant la détection de copies [1], la recherche d'images similaires [2, 3, 4], la reconnaissance de concepts sémantiques [5]. Dans cet état de l'art, nous nous focaliserons sur les méthodes de recherche d'objets visuels, autrement nommées recherche d'instances (ainsi défini par la campagne d'évaluation Trecvid¹). La plupart des méthodes que nous présenterons seront pour autant tout à fait adaptées à la recherche d'images similaires, bien que parfois sur-dimensionnées par rapport au besoin.

La figure 2.1 page suivante présente le schéma général de l'architecture d'un système de recherche d'images par le contenu. Les images sont tout d'abord représentées par un ensemble de descripteurs (cf. section 2.2). Ces descripteurs sont ensuite partitionnés (cf. section 2.3 page 27), puis indexés pour être efficacement retrouvés lors de la recherche (cf. section 2.4 page 35). La recherche d'une image (ou zone d'image) produit en sortie une liste d'images contenant des descripteurs similaires à ceux requêtés. Une vérification de la cohérence géométrique des appariements est ensuite souvent appliquée (cf. section 2.5 page 39). Enfin, une extension de requête est optionnellement utilisée afin d'améliorer le rappel (cf. section 2.6 page 42).

2.2 Description d'images

2.2.1 Descripteurs bas niveau

Le contenu visuel des images peut être décrit par des caractéristiques bas niveau (aussi appelées descripteurs) représentant statistiquement le signal image, comme la texture, la forme, etc. ou par des concepts visuels de plus haut niveau, comme des objets reconnaissables. Ces concepts visuels de plus

1. <http://www-nlpir.nist.gov/projects/tv2012/tv2012.html#ins>

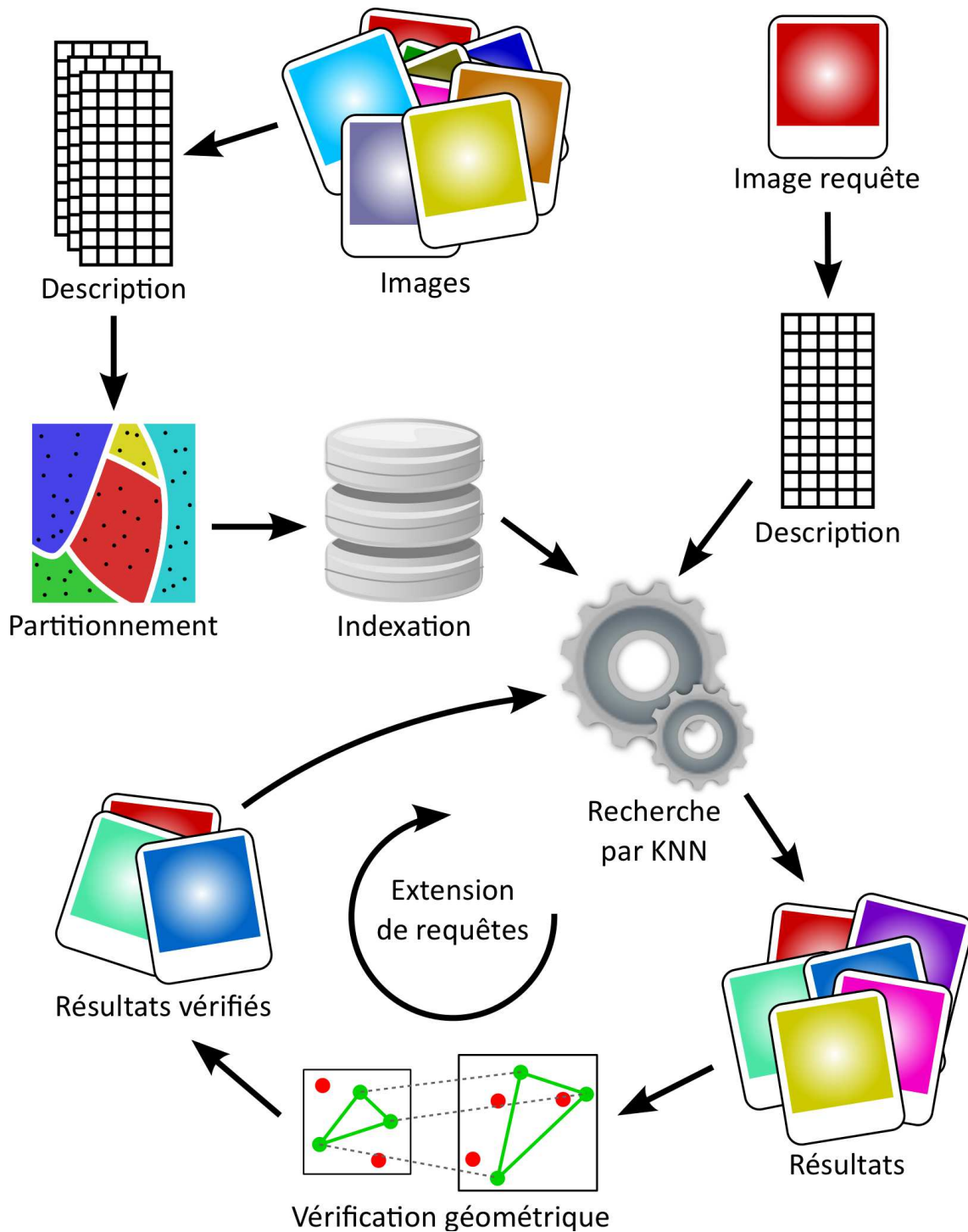


FIGURE 2.1 – Architecture générale d'un système de recherche d'images par le contenu

379 haut niveau sont généralement obtenus à partir des descripteurs bas niveau des images, et non pas
 380 directement à partir des images brutes. Par la suite, nous passerons en revue les descripteurs bas niveau
 381 les plus utilisés dans le domaine de la recherche d'objets visuels (sans nous arrêter sur les concepts
 382 visuels de haut niveau).

383 Il existe deux catégories principales de descripteurs bas niveau : ceux opérant à l'échelle de l'image
 384 (les descripteurs globaux), et ceux opérant à l'échelle de portions d'images (les descripteurs locaux).

385 2.2.1.1 Descripteurs globaux

386 Couleurs, textures et formes sont connues pour être les principales caractéristiques visuelles pouvant
 387 décrire le contenu global des images. Swain et Ballard [6] furent parmi les premiers à proposer d'utiliser
 388 les histogrammes de couleurs pour l'indexation d'images. De nombreuses améliorations furent ensuite
 389 apportées par [7, 8, 9, 10, 11]. La transformation en ondelettes de Daubechie a été utilisée pour la
 390 description de texture dans [12]. Les descripteurs visuels inclus dans le standard MPEG7 [13] sont parmi
 391 les plus connus. Ils consistent en un ensemble de descripteurs de couleur et de texture adaptés aux
 392 images et vidéos naturelles [14].

393 L'un des descripteurs globaux les plus utilisés est GIST [15], qui propose une représentation de l'image
 394 en faible dimension. Les images sont divisées en 4x4 zones spatiales, dans lesquelles sont extraits des
 395 histogrammes d'orientation.

396 Les descripteurs globaux ont été longtemps utilisés, mais le sont de moins en moins, au profit
 397 des descripteurs locaux. La raison principale de cette disparition provient de l'impossibilité pour ces
 398 descripteurs de représenter des objets couvrant une faible portion d'image. Les caractéristiques de ces
 399 petits objets sont ainsi noyées dans un seul descripteur global par image.

400 Si certains systèmes continuent aujourd'hui d'employer de tels descripteurs, c'est en raison de leurs
 401 deux principaux avantages : ils sont très peu coûteux en mémoire, et très rapides à calculer. Mais
 402 d'autres avantages sont également à prendre en considération : la plupart de ces descripteurs sont des
 403 histogrammes, et il est donc très facile de les combiner. Enfin, ils font toujours partie des outils les plus
 404 efficaces dans les domaines de la catégorisation de scènes ou encore de la reconnaissance de type
 405 d'image (photographie naturelle, clip-art, dessin au trait,...).

406 2.2.1.2 Descripteurs locaux

407 Comme on l'a vu précédemment, l'intérêt d'utiliser des descripteurs locaux plutôt que des globaux
 408 provient de leur capacité à représenter des petites portions d'images centrées autour d'un point ou
 409 d'une région d'intérêt, et donc à décrire beaucoup plus finement les images. Une très large variété de
 410 descripteurs locaux a été proposée, certains étant dépendants de la localisation de points ou de régions
 411 d'intérêts, d'autres non (cf. approches denses où les descripteurs sont extraits depuis des positions
 412 aléatoires [16], ou depuis une grille [17]).

413 On trouve de nombreuses méthodes de calcul de point d'intérêt dans la littérature. La tâche principale
 414 est de détecter une zone locale en utilisant des informations invariantes à certaines transformations.

415 Parmi ces transformations, on distingue les transformations géométriques (rotation, changement
416 d'échelle...), les transformations photométriques (changements d'illumination), le bruit, la compression...

417 Certaines sont basées sur la détection de coins comme les méthodes de Harris [18], Shi et Tomasi
418 [19], SUSAN [20] ou FAST [21]. D'autres sont basées sur la détection de blobs utilisant des laplaciens de
419 gaussiennes (LoG) [22], des différences de gaussiennes (DoG) [23], ou des ondelettes de Haar (SURF)
420 [24]. Les points d'intérêts les plus utilisés dans les systèmes de recherche d'images par le contenu sont
421 généralement ceux résistant à une transformation affine, comme MSER [25], Harris-affine [26], ou encore
422 Hessian-affine [27, 28].

423 On peut distinguer trois catégories de descripteurs locaux : ceux utilisant des histogrammes, ceux
424 utilisant les différentielles et les moments, et enfin ceux basés sur les fréquences spatiales.

425 2.2.1.2.1 *Descripteurs basés histogrammes*

426 Ces descripteurs intègrent dans un histogramme différentes caractéristiques visuelles comme la
427 couleur, la luminance, les contours, les orientations de gradients, les distances radiales, ou la texture.
428 Ainsi, [29, 30, 31] proposent d'utiliser des histogrammes représentant la distribution des intensités
429 des pixels. Dans [32], Zabih et Woodfill proposent une méthode décrivant les textures, robuste aux
430 changements d'illumination, basée sur la relation entre les intensités des pixels plutôt que directement
431 sur les intensités.

432 Belongie et al. [33] calculent un histogramme de contours extraits par le détecteur de Canny
433 et la position est quantifiée dans le domaine log-polaire. Ce descripteur, appelé « shape-context », a
434 récemment montré d'excellentes performances dans le domaine de la reconnaissance foliaire, lors de la
435 tâche « identification de plante » à ImageCLEF 2012².

436 Lowe [23] propose un descripteur invariant au changement d'échelle (SIFT), ainsi qu'à la rotation
437 et à l'illumination. Le descripteur est représenté par un histogramme 3D de gradients locaux orientés,
438 pondérés par l'amplitude du gradient. Il est stocké dans un vecteur à 128 dimensions (8 intervalles
439 d'orientation pour 4x4 zones spatiales autour du point d'intérêt). La richesse de l'information contenu
440 dans SIFT en fait un descripteur robuste aux petites distorsions géométriques et aux petites erreurs de
441 détection du point d'intérêt. Ce descripteur est de loin le plus populaire de ceux actuellement utilisés.

442 Diverses améliorations de SIFT ont été suggérées. Par exemple, Ke et Sukthankar [34] proposent
443 PCA-SIFT, une méthode dans laquelle ils appliquent une PCA sur les patchs de pixels. Micolaczyk et al. [35]
444 proposent une variante de SIFT appelée GLOH (Gradient Location-Orientation Histogram) dont le but est
445 d'améliorer la robustesse et la distinctivité, en utilisant une grille en log-polaire avec 3 intervalles dans la
446 direction radiale et 8 dans la direction angulaire. Les orientations des gradients étant quantifiées sur 16
447 intervalles, ils appliquent ensuite une PCA pour retrouver un total de 128 bits.

448 Plus récemment, Morel et Yu [36, 37] ont introduit une version affine de SIFT (ASIFT). En prenant en
449 compte les paramètres de la caméra (angles) en plus des paramètres de zoom, rotation et translation
450 utilisés dans SIFT, ils obtiennent un descripteur invariant aux changements de point de vue de la

2. <http://www.imageclef.org/2012>

451 caméra. Ceci permet donc d'augmenter le nombre de correspondances entre deux images similaires.
452 L'inconvénient de ASIFT est que sa complexité est environ 13 fois celle de SIFT, et que le nombre de
453 descripteurs obtenus dans une image est également beaucoup plus élevé que pour SIFT.

454 Associé à leur point d'intérêt (SURF), Bay et al. [24] proposent un autre descripteur également robuste
455 aux changements d'échelle et de rotation. Comme le point d'intérêt, le descripteur est calculé à partir des
456 coefficients des ondelettes de Haar.

457 *2.2.1.2.2 Descripteurs basés sur les différentielles et les moments*

458 Ce type de descripteurs incorpore l'information relative aux dérivées des variations de l'image autour
459 des points d'intérêt. Tout comme précédemment, les variations considérées sont du type couleurs,
460 contours, ...Les moments invariants peuvent également être utilisés pour modéliser la structure des
461 régions. Les propriétés des dérivées locales (Local Jet) ont été étudiées par Koenderink et van Doorn
462 [38]. Florack et al. [39] dérivent les invariants différentiels, qui combinent les composantes des Local
463 Jets pour obtenir l'invariance en rotation. Van Gool et al. [40] combinent également les Local Jets de
464 différentes façons pour obtenir également l'invariance aux transformations affines et aux changements
465 photométriques.

466 Freeman et Adelson proposent dans [41] d'utiliser des filtres orientables leur donnant une invariance
467 en rotation.

468 Joly [42] utilise des dipôles dissociés pour construire des descripteurs normalisés de dimension 20.
469 Ces dipôles dissociés sont des opérateurs différentiels non locaux, construits à partir d'une paire de lobes
470 gaussiens. Les expérimentations dans [42] montrent que ces descripteurs donnent de meilleurs résultats
471 que SIFT (en termes de mean Average Precision) sur la recherche d'images de copies d'images sur le web.

472 BRIEF [43] est un descripteur basé sur de simples tests binaires entre pixels dans une image lissée,
473 permettant une extraction très rapide. Ses performances sont similaires à celles de SIFT sur de nombreux
474 aspects comme la robustesse aux changements d'illumination, au flou. Il est par contre très sensible
475 aux rotations dans le plan et aux changements d'échelle, ce qui le rend peu adapté à la recherche ou
476 découverte d'instances, en milieu naturel.

477 *2.2.1.2.3 Descripteurs basés sur les fréquences spatiales*

478 Ces descripteurs modélisent la fréquence spatiale dans le voisinage des points d'intérêts en
479 transformant l'image en un ensemble de coefficients fréquentiels obtenus par différentes transformations
480 (Fourier, TCD, Gabor, ondelettes). Une comparaison des descripteurs de texture basés sur Gabor
481 est présentée dans [44]. Les descripteurs basés fréquences présentent l'avantage d'être proche des
482 caractéristiques utilisées par le cerveau humain, mais l'inconvénient d'être coûteux en mémoire.

483 *2.2.1.2.4 Analyse*

484 Une comparaison de plusieurs de ces descripteurs est disponible dans [45, 27, 35, 46]. La conclusion
485 que l'on peut tirer est que SIFT et ses nombreuses variantes sont un excellent compromis pour la

486 recherche d'objets visuels. Ces descripteurs sont à la fois capables de stocker une grande quantité
487 d'informations représentant au mieux la zone d'image décrite, et à la fois capable de résister à de
488 nombreuses transformations, qu'elles soient géométriques ou photométriques. SIFT semble aujourd'hui
489 le détecteur et descripteur de points d'intérêt le plus utilisé dans les systèmes de recherche d'objets
490 visuels. L'usage de ASIFT pourrait permettre d'améliorer significativement les résultats de recherche de
491 petits objets, où le nombre d'appariements est souvent très faible. Cependant, ses coûts mémoire et
492 calculatoire sont un frein à son utilisation sur de grandes bases.

493 2.2.2 Représentation des images

494 A l'issue de l'étape d'extraction de descripteurs bas niveau, deux grandes stratégies de représentation
495 des images s'opposent à nouveau : une stratégie locale, et une stratégie globale (sac de mots visuels,
496 Fisher vectors) .

497 2.2.2.1 Représentation totale (locale)

498 Cette stratégie de représentation, issue de l'appariement d'images consiste à conserver toute
499 l'information locale des images. La recherche d'une image (ou zone d'image) est alors effectuée en
500 recherchant les descripteurs similaires à chacun des descripteurs composant la requête. Le score de
501 similarité pouvant alors être le nombre de correspondances similaires. Toutefois, ce score est souvent
502 revu lors d'une phase de vérification de la cohérence géométrique.

503 2.2.2.2 Sac de mots visuels

504 La stratégie du sac de mots visuels, introduite par Sivic et Zisserman [3] est la première à s'être
505 imposée dans les systèmes de recherche d'images par le contenu à grande échelle. Le terme mot visuel
506 tient son origine de l'analogie que l'on peut faire avec les moteurs de recherche textuels. Dans la méthode
507 classique, les documents textuels sont en effet représentés sous forme de sacs de mots où la faisabilité de
508 cette approche tient du fait que le vocabulaire est défini par le langage utilisé. Il n'en va évidemment
509 pas de même pour le domaine visuel, dans lequel il n'existe pas de vocabulaire : les descripteurs sont
510 généralement des vecteurs de nombres flottants. Il en existe donc potentiellement une infinité. Afin de se
511 replacer dans le cadre des moteurs de recherche textuels, Sivic et Zisserman [3] proposent donc de créer
512 un dictionnaire de mots visuels dont la taille est maîtrisée. La création de ces mots est souvent réalisée
513 grâce à un *clustering* de type K-Means [3].

514 Une image est ensuite représentée par un histogramme à K cellules comptant le nombre
515 d'occurrences de chaque mot visuel présent.

516 La similarité entre deux images peut ensuite être calculée comme étant le cardinal de l'intersection
517 entre leurs sacs de mots visuels. La mesure est donc une mesure de similarité globale, bien qu'un faible
518 score puisse tout de même indiquer la présence d'une similarité locale.

519 Pyramid Match Kernel [47], proposé par Grauman et Darrell, est un algorithme étendant le modèle
520 des sacs de mots visuels en calculant des histogrammes multi-résolutions. Pour cela, l'espace des

521 descripteurs est quantifié en utilisant des cellules de taille croissante. Ce partitionnement de l'espace
 522 permet à certaines résolutions, de regrouper des descripteurs qui ne l'auraient pas été autrement.
 523 Grauman et Darrell [47] présente ensuite un noyau permettant de calculer la taille de l'intersection de ces
 524 histogrammes pyramidaux. Ce calcul est linéaire en nombre de descripteurs, contrairement à d'autres
 525 approches basées noyau.

526 Lazebnik et al. [48] préfèrent utiliser des méthodes de quantification de vecteur standard, mais
 527 utilisent par contre des pyramides pour quantifier l'espace spatial à différentes résolutions. Concrètement,
 528 pour chaque type différent (mot visuel), on obtient par image un ensemble d'histogrammes comptant le
 529 nombre de descripteurs tombant dans chaque cellule de l'histogramme.

530 2.2.2.3 Fisher vectors

531 Les Fisher vectors introduits par [49], ont tout d'abord été utilisés dans le domaine de la catégorisation
 532 de scènes [50], de la réduction de dimensions [51] ou de la détection de zones saillantes [52].
 533 Contrairement au modèle du sac de mots visuels, les Fisher vectors n'encodent pas la présence d'un mot
 534 visuel dans une image, mais plutôt la dissimilarité entre le contenu de l'image et le vocabulaire généré
 535 au préalable. Cet encodage génère donc des vecteurs denses puisque chaque cellule de l'histogramme
 536 contient une valeur. Cette représentation améliore le modèle du sac de mots visuels sur deux points :
 537 premièrement elle n'est pas simplement limitée au comptage des occurrences de chaque mot visuel, et
 538 deuxièmement elle encode des informations additionnelles sur la distribution des descripteurs.

539 Dans [53], Perronnin et al. proposent une méthode de compression adaptée afin de réduire le coût
 540 mémoire ainsi que le coût de calcul.

541 Les Fisher vectors ont récemment montré d'excellentes performances sur la base ImageNet³ pour la
 542 classification d'objets.

543 2.3 Partitionnement de l'espace visuel

544 Préalablement à l'étape d'indexation des descripteurs, on effectue un partitionnement de l'espace
 545 des descripteurs afin de pouvoir construire les index.

546 Le principe du partitionnement de l'espace visuel est de réaliser la transformation $\mathbb{R}^d \mapsto u, u \in \mathbb{N}$, qui
 547 à un vecteur x associe un entier u , et qui à un vecteur y similaire à x , va associer ce même entier u avec
 548 une certaine probabilité.

549 Cet entier u est nommé différemment selon les méthodes de partitionnement : on le nomme clé
 550 de hachage dans les méthodes de hachage, *cluster* dans les méthodes de clustering, feuille dans les
 551 méthodes utilisant des arbres, région de Voronoi pour les treillis ...

552 Il est à noter que les « mots visuels » utilisés dans certaines méthodes sont également des parties de
 553 l'espace.

554 Ce point de vue est d'ailleurs partiellement partagé par Keim et al. [54], qui considèrent les méthodes
 555 de partitionnement comme des méthodes de *clustering*.

3. <http://www.image-net.org/>

556 En réalisant un tel partitionnement, on tend vers deux objectifs principaux :
 557 – Indexer les données ;
 558 – Compresser les données. On avait des vecteurs de grande dimension (e.g. 128×4 octets), on a
 559 maintenant un entier (codable avec quelques dizaines ou centaines de bits) ;
 560 Il y aurait des dizaines de manières différentes de « partitionner » toutes les méthodes de
 561 partitionnement. Un point de vue intéressant est celui qui consiste à séparer les méthodes indépendantes
 562 des données, de celles qui en sont dépendantes.

563 2.3.1 Partitionnement indépendant de la distribution des données

564 2.3.1.1 Partitionnement basé sur des structures déterministes

565 2.3.1.1.1 Treillis

566 Les treillis (*lattices* en anglais) sont des structures régulières et infinies, définies par un ensemble
 567 de vecteurs. Chaque point d'un treillis peut être identifié par un unique nombre entier. Les régions de
 568 Voronoi autour des points d'un treillis ont toutes la même forme et le même volume. Deux vecteurs ayant
 569 été encodés avec le même point d'un treillis sont séparés par une distance maximum, dépendant des
 570 paramètres du treillis. Les treillis ont été utilisés pour la quantification vectorielle dans [55] et [56]. Dans
 571 le cas d'une distribution uniforme, [55] montre que les treillis sont plus efficaces que les quantifieurs
 572 scalaires. Dans [57], Tuytelaars et Schmid utilisent un treillis en forme de grille carrée. Chaque dimension
 573 du descripteur est quantifiée sur 2 bits. Devant le très grand nombre de points composant le treillis, les
 574 auteurs proposent de ne garder que les points qui se voient affecter au moins un descripteur.

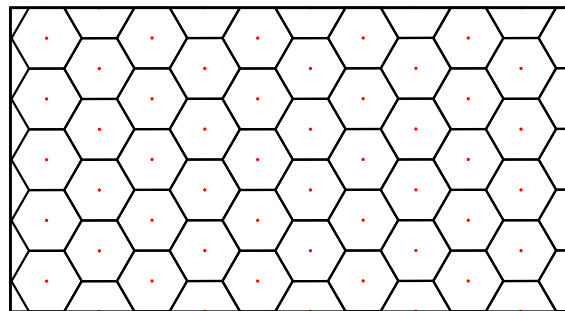


FIGURE 2.2 – Exemple d'un treillis hexagonal

575 2.3.1.1.2 Courbes remplissant l'espace

576 Les courbes remplissant l'espace (*space filling curves*, voir [58] pour une revue), comme la courbe
 577 de Hilbert ou l'ordonnancement en Z sont des courbes définies de manière récursive dans l'espace
 578 multidimensionnel et dont le tracé tend vers un recouvrement complet de l'espace lorsque l'ordre de
 579 la courbe tend vers l'infini. Elles permettent d'établir une transformation bijective entre un espace de
 580 dimension D et un espace monodimensionnel (coordonnée curviligne d'un point). Les distances entre

581 points ne sont pas préservées mais une certaine forme de localité est préservée. Ainsi deux points voisins
 582 sur la courbe sont proches dans l'espace. Inversement, tous les points proches dans l'espace ne le sont
 583 pas systématiquement sur la courbe. Le principe du partitionnement à base de *space filling curves* est de
 584 convertir chaque vecteur en une position sur la courbe.

585 Cette position est quantifiée avec un nombre de bits dépendant de l'ordre de la courbe.

586 Les *space filling curves* ont principalement été employées dans le domaine de la recherche
 587 d'informations pour la détection de copies [59, 60].

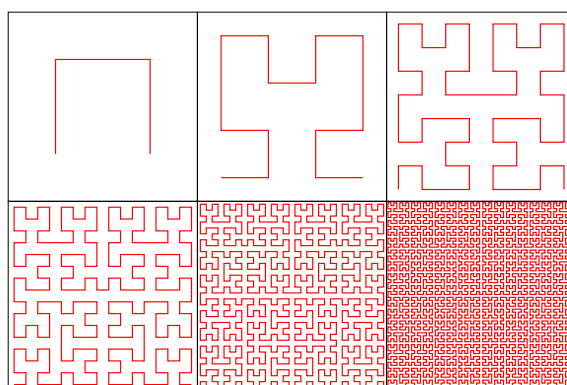


FIGURE 2.3 – Les six premières itérations d'une courbe de Hilbert (certains droits réservés, licence CC-By-SA, Braindrain0000)

588 2.3.1.2 Familles de hachage basées sur des transformations aléatoires

589 Les méthodes aléatoires sont généralement considérées comme plus adaptatives dans le cas de
 590 données ayant une distribution hétérogène, et sont donc plus efficaces que les méthodes basées sur une
 591 structuration déterministe (les descripteurs tels que SIFT ne respectent pas une distribution homogène).
 592 Ces méthodes sont un moyen de gérer les très grandes dimensions (> 30).

593 2.3.1.2.1 Locality Sensitive Hashing

594 Ces méthodes utilisent des familles de fonctions définies par des processus aléatoires, comme les
 595 fonctions de type *Locality Sensitive Hashing* (LSH), introduite par Indyk et al. [61].

596 Une fonction de type LSH associe une clé de hachage (un bit ou un entier) à chaque vecteur de
 597 l'espace à partitionner. Cette association est faite de telle sorte que les vecteurs similaires ont une
 598 probabilité plus forte que les vecteurs dissimilaires d'obtenir la même clé de hachage. En concaténant
 599 plusieurs clés de hachage obtenues avec différentes fonctions, on obtient une version compressée des
 600 vecteurs partitionnés.

601 Les fonctions de hachage utilisées sont généralement choisies pour approximer une distance
 602 particulière :

- 603 – Les fonctions de hachage par échantillonnage de bits [61] permettent d'approximer la distance de
 604 Hamming entre les vecteurs. Dans ce cas, la fonction de hachage sélectionne aléatoirement un bit i

- 605 dans le vecteur x , c'est-à-dire $h = x[i]$;
- 606 – Les fonctions de hachage de type cosinus qui approximent le produit scalaire entre vecteurs. Ces
 607 fonctions projettent les vecteurs sur des hyperplans aléatoires r passant par l'origine, et conservent
 608 le signe après projection (cosinus positif ou négatif), soit $h = \text{sign}(\langle x, r \rangle)$;
- 609 – Les fonctions qui approximent la distance L_1 : $h = \lfloor (x[i] - b)/w \rfloor$;
- 610 – Les fonctions qui approximent la distance L_2 [62] par projection sur des hyperplans aléatoires avec
 611 biais : $h = \lfloor (\langle x, r \rangle - b)/w \rfloor$;
- 612 – Plus généralement, les fonctions qui approximent la distance L_p par l'utilisation de distributions
 613 p-stables [63] ;
- 614 – Les fonctions qui approximent les distances sur une hypersphère de rayon unitaire [64] (*Spherical*
 615 *Simplex*, *Spherical Orthoplex*, *Spherical Hypercube*) ;
- 616 – Les *Shift-invariant kernels* [65], qui approximent tout noyau invariant en translation ;
- 617 – *Min-Hashing* [66] qui estime la distance de Jacquard entre deux ensembles ;

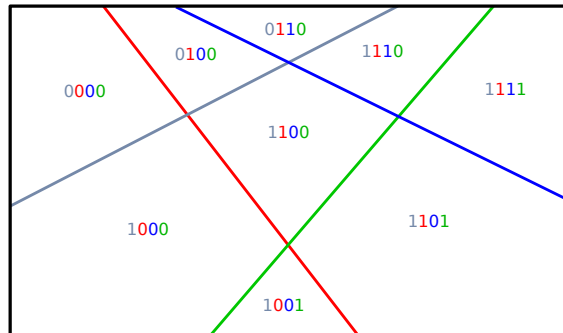


FIGURE 2.4 – Exemple de hachage en 2D avec projections aléatoires et 4 bits (4 hyperplans)

618 2.3.2 Partitionnement dépendant de la distribution des données

619 Les méthodes de partitionnement dépendant de la distribution des données sont comme leur nom
 620 l'indique, des méthodes qui s'adaptent à la distribution des données. Les descripteurs ne présentant
 621 généralement pas une distribution uniforme dans l'espace, on voit toute de suite l'intérêt de ce type de
 622 méthode. Cette distribution peut également varier en fonction de la base de données étudiée (une base
 623 contenant des photographies prises par des touristes en vacances sera différente d'une base d'images
 624 aériennes).

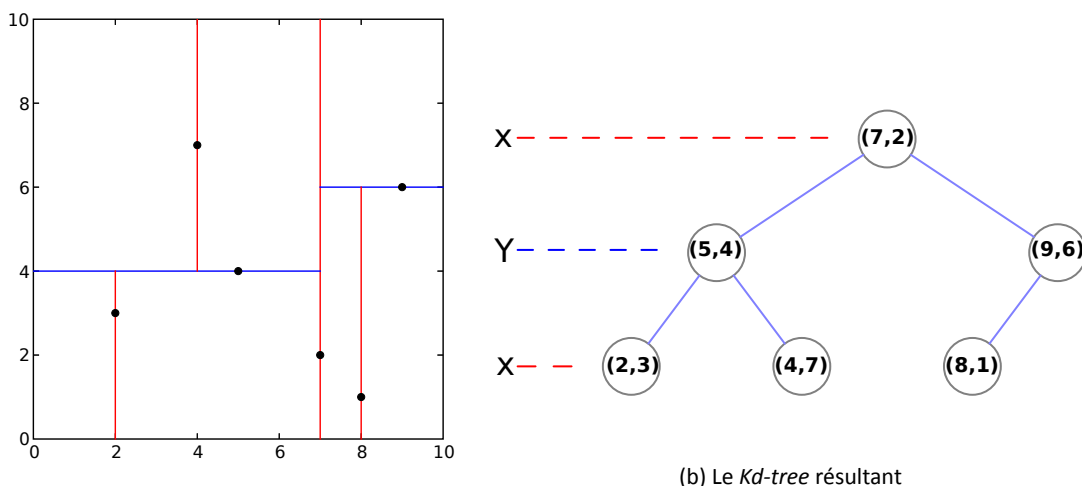
625 Mais il est important de noter que rien n'empêche d'apprendre un partitionnement sur un ensemble
 626 de données, et d'appliquer ensuite ce partitionnement à un autre ensemble. Cette approche est parfois
 627 utilisée dans les méthodes qui nécessitent un très long temps d'apprentissage. Avec un clustering
 628 *K-Means* par exemple, on peut apprendre la position des centres des clusters sur une petite base de
 629 données, et réaliser la phase d'affectation sur une autre base beaucoup plus grande. Cette astuce dégrade
 630 malheureusement significativement les résultats, comme le montre l'étude réalisée dans le tableau 2.1
 631 page 45.

632 **2.3.2.1 Partitionnement hiérarchique**

633 De nombreuses méthodes à base d'arbres ont été appliquées à la structuration de descripteurs visuels,
 634 comme les *R-tree* [67, 68, 69], les *SS-Tree* (*Similarity Search Tree*) [70], les *SR-Tree* (*Sphere-Rectangle Tree*)
 635 [71, 72], ou encore les *X-tree* (*eXtended node Tree*) [73].

636 Malgré la profusion de méthodes à base d'arbres, seuls les *Kd-tree* [74] et les méthodes de *K-Means*
 637 hiérarchiques sont fréquemment utilisés dans les systèmes récents.

638 Un *Kd-tree* est un arbre binaire divisant l'espace des descripteurs en deux, à chaque niveau de l'arbre,
 639 sur une seule dimension. Cette division garantit un partitionnement sans aucun recouvrement entre les
 640 feuilles. On voit une illustration d'un partitionnement par *Kd-tree* dans la figure 2.5. Ce type d'arbre
 641 étant mal équilibré, plusieurs variantes ont été proposées pour résoudre ce problème, comme *Adaptive*
 642 *KD-tree* [75], *KDB-tree* [76], *LSDh-tree* [77], *Randomized Kd-tree* [78, 79, 80].



(a) Partitionnement de l'espace (certains droits réservés, licence CC-BY-SA-3.0-MIGRATED, KiwiSunset)

FIGURE 2.5 – Exemple de partitionnement avec un *Kd-tree*

643 Dans l'algorithme HKM (*Hierarchical K-Means*) [81], Nistér et Stewénus appliquent un *K-Means* avec
 644 un très faible nombre K (10 par exemple), puis recommencent un nouveau *K-Means* sur chaque
 645 sous-ensemble de descripteurs créés par les K clusters. L'algorithme itère ensuite jusqu'à obtention d'un
 646 nombre de *clusters* total suffisant. Ce nombre étant égal à K^n , avec n le nombre d'itérations. HKM
 647 permet de réduire la complexité de l'algorithme à $O(N \log K)$.

648 Bien que beaucoup plus efficace en termes de temps de calcul, HKM ne produit pas d'aussi bons
 649 résultats de *clustering* [82] qu'un *K-Means* standard.

650 BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies*) [83] a l'avantage de s'adapter
 651 aux ressources disponibles (temps et mémoire). Trois étapes sont nécessaires :

- 652 – La première consiste à créer une structure hiérarchique (*CF-Tree*) qui va permettre d'isoler les
 - 653 vecteurs éloignés. Les feuilles d'un *CF-Tree* définissent des micro-clusters ;
 - 654 – La deuxième est un *clustering* ascendant opérant sur les micro-clusters générés à l'étape
 - 655 précédente ;
 - 656 – La troisième étape consiste en l'affectation des vecteurs de la base aux *clusters* issus de la deuxième
 - 657 étape ;
- 658 Berrani [84] propose d'améliorer la première phase de BIRCH, afin de résoudre plusieurs problèmes
- 659 liés au coût d'insertion dans le *CF-Tree*, et à la taille des micro-clusters.

660 2.3.2.2 Familles de hachage dépendant des données

661 Dans ce type de méthode, les fonctions de hachage sont définies à partir d'une base d'apprentissage

662 représentative des données (sous-ensemble des données à hacher). L'objectif de ces méthodes est de

663 s'adapter au mieux à la distribution des données, pour améliorer la sélectivité et le balancement des clés

664 de hachage, tout en préservant la sensibilité à la localité.

665 2.3.2.2.1 *Restricted Boltzmann Machine*

666 RBM [85] est basé sur l'apprentissage d'un réseau de neurones multi-couches. L'une des couches est

667 composée de neurones visibles, et l'autre de neurones cachés. Les neurones n'ont pas de connexions

668 avec ceux de leur couche. Ils sont par contre connectés avec tous les neurones de l'autre couche. Cette

669 connexion est bidirectionnelle et symétrique.

670 2.3.2.2.2 *Spectral Hashing*

671 Une autre méthode souvent citée dans la littérature est *Spectral Hashing* [86], basée sur la théorie

672 du partitionnement des graphes. Dans cette approche, les données sont binarisées de telle sorte que

673 la distance de Hamming approxime la distance Euclidienne. Il est à noter que la méthode voit ses

674 performances décroître lorsque plus de 512 bits sont utilisés. Cette limite vient du fait que les auteurs

675 utilisent une analyse en composantes principales. Les premiers bits générés sont donc très utiles, mais les

676 sont de moins en moins au fur et à mesure du hachage.

677 2.3.2.2.3 *Spherical Hashing*

678 Contrairement aux méthodes de type LSH qui utilisent des hyperplans pour hacher les données, Heo

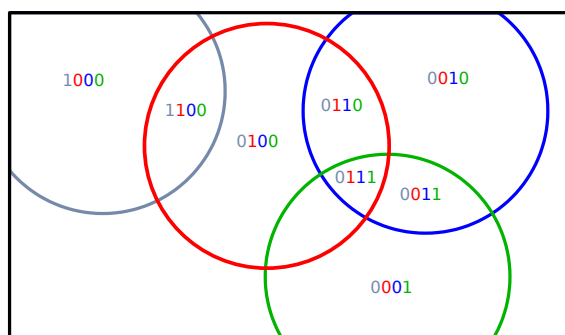
679 et al. [87] proposent une nouvelle fonction de hachage basée sur des hypersphères. Chaque hypersphère

680 définit la valeur d'un bit (le vecteur est compris dans l'hypersphère ou non). Ils proposent également

681 une nouvelle fonction de distance binaire appropriée à leur fonction de hachage. La méthode présentée

682 à l'avantage de créer un partitionnement équilibré, avec des fonctions de hachage indépendantes, qui

683 donnent d'excellents résultats lorsqu'il s'agit d'approximer la distance L2 sur des descripteurs GIST.

FIGURE 2.6 – Exemple de hachage en 2D avec *spherical hashing* et 4 bits (4 hypersphères)

684 2.3.2.2.4 Kernelized Locality Sensitive Hashing

685 Dans KLSH [88, 89], Kulis et Grauman présentent une approche légèrement différente, étant donné
 686 que leur principal objectif est de généraliser le hachage à n'importe quel noyau de Mercer plutôt que
 687 d'obtenir de meilleurs résultats. Ce type de généralisation semble connaître un certain engouement, avec
 688 des méthodes comme RMMH [90] ou une amélioration de *Product Quantization* [91].

689 2.3.2.2.5 Product Quantization

690 *Product Quantization* [92, 93] est une méthode actuellement très populaire. Les données sont tout
 691 d'abord partitionnées par un *clustering K-Means* (avec un assez faible nombre de *clusters*), puis chaque
 692 sous *cluster* et à nouveau partitionné. Ce deuxième partitionnement est effectué sur les résidus (du
 693 premier partitionnement) des sous ensembles de composantes des vecteurs. Chaque sous-ensemble
 694 quantifié produit une clé de hachage qui peut être concaténée à d'autres pour former une clé de hachage
 695 beaucoup plus discriminante. On constate toutefois que la méthode se limite à environ 64 bits, alors que
 696 certaines applications demanderaient une plus grande sélectivité. La réalisation d'un tel partitionnement
 697 peut nécessiter un coût de calcul assez important. En effet, même si l'apparition des méthodes de
 698 *K-Means* approximatif, comme AKM [94], a permis de réduire la complexité de la phase d'affectation
 699 de $O(NK)$ à $O(N \log K)$ (avec N le nombre de vecteurs et K le nombre de *clusters*), la complexité
 700 reste très importante. Que l'on ait $K = 10^3$, ou $K = 10^6$, ne change la complexité que d'un facteur 2
 701 ($\log(10^6) = 2 \log(10^3)$).

702 Dans [91], Bourrier et al. étendent la méthode aux *kernels* χ^2 , en exploitant le concept
 703 d'« *embedding explicite* » permettant de plonger les données dans un espace Euclidien, pour y effectuer
 704 la recherche.

705 2.3.2.2.6 Random Maximum Margin Hashing

706 Plusieurs études [93, 86, 85] ont montré qu'un hachage dépendant des données permettait
 707 l'obtention de meilleurs résultats de recherche. Mais ceci n'est généralement vrai que pour des tailles de
 708 clé de hachage limitées, ne dépassant pas 64 ou 128 bits. En effet, le problème de ces méthodes est que

709 le bénéfice de leur utilisation voit sa croissance diminuer au fur et à mesure qu'augmente le nombre de
 710 fonctions de hachage. Ceci s'explique par le manque d'indépendance entre les fonctions.

711 Dans *Random Maximum Margin Hashing* (RMMH) [90], Joly et Buisson proposent de nouvelles
 712 fonctions de hachage applicables à n'importe quel type de noyau, résolvant en partie ce problème
 713 d'indépendance. Pour chaque fonction de hachage, un sous-ensemble des données est sélectionné
 714 aléatoirement, et séparé en deux groupes de même taille. Un séparateur à vaste marge (SVM)
 715 appris sur ces données labellisées. L'utilisation de SVM leur permet d'éviter le sur-apprentissage et offre
 716 de bonnes capacités de généralisation.

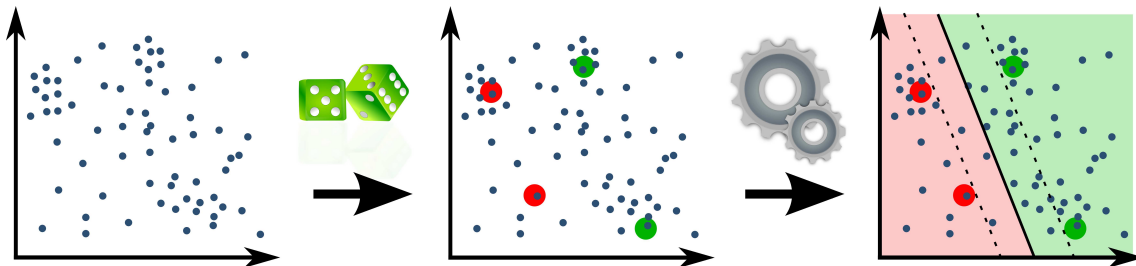


FIGURE 2.7 – Exemple de hachage en 2D avec RMMH

717 2.3.2.3 Partitionnement basé regroupement

718 2.3.2.3.1 *K-Means*

719 La méthode de structuration la plus fréquemment utilisée dans les systèmes existants est
 720 certainement celle réalisant un *clustering* des descripteurs. Bien que plusieurs algorithmes de *clustering*
 721 aient été envisagés pour cette tâche, c'est le *clustering* par *K-Means* qui semble remporter le plus de
 722 succès [3, 30, 95].

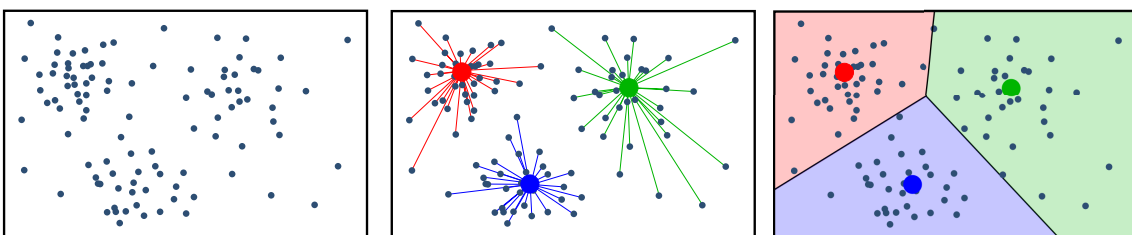


FIGURE 2.8 – Exemple de partitionnement par *K-Means*

723 La plupart des travaux utilisant *K-Means* emploient la version standard, dont l'utilisation devient
 724 totalement déraisonnable pour de grandes tailles de base et de vocabulaire (plusieurs millions de
 725 descripteurs), la complexité d'une itération étant de $O(NK)$, avec N le nombre de descripteurs, et K
 726 le nombre de *clusters*.

727 Les propositions permettant le passage à l'échelle de *K-Means* se sont principalement focalisées sur
 728 l'utilisation de *clusters* hiérarchiques [96], comme dans l'algorithme HKM (*Hierarchical K-Means*) [81]

729 pour lequel Nistér et Stewénus appliquent un *K-Means* avec un très faible nombre K (10 par exemple),
730 puis recommencent un nouveau *K-Means* sur chaque sous-ensemble de descripteurs créés par les K
731 clusters. L'algorithme itère ensuite jusqu'à obtention d'un nombre total de *clusters* suffisant. Ce nombre
732 étant égal à K^n , avec n le nombre d'itérations. HKM permet de réduire la complexité de l'algorithme à
733 $O(N \log K)$.

734 Bien que beaucoup plus efficace en termes de temps de calcul, HKM ne produit pas d'aussi bons
735 résultats de *clustering* [82] que la version standard.

736 Philbin et. al [94] proposent une version approximative de *K-Means* (AKM), qui contrairement à HKM,
737 minimise la même fonction de coût que la version standard. Ils montrent d'ailleurs que AKM donne
738 d'aussi bons résultats qu'un *K-Means* standard, tout en conservant la même complexité que HKM,
739 c'est-à-dire $O(N \log K)$. Ils parviennent à cette accélération en réduisant le coût principal du *K-Means* :
740 la détermination du centre de *cluster* le plus proche pour chaque point. Pour cela, ils appliquent une
741 méthode de recherche de plus proches voisins basée sur une forêt de plusieurs *Kd-tree* aléatoires [78, 79,
742 80].

743 2.3.2.3.2 *Mean-shift*

744 Les méthodes à base de *Mean-shift* sont très peu présentes dans la littérature. Mais on peut toutefois
745 s'intéresser à [97], dans lequel Jurie et Triggs utilisent un *clustering* à base de *Mean-shift*. *Mean-shift*
746 [98] est un algorithme de recherche de mode, basée sur l'analyse non paramétrique de l'espace des
747 descripteurs. Jurie et Triggs [97] montrent que leur méthode obtient de meilleurs résultats qu'un *K-Means*
748 standard.

749 2.3.3 Le problème du partitionnement « brutal »

750 Toutes les méthodes de partitionnement souffrent du même défaut : il n'existe pas de partitions
751 parfaites. Qu'importe la méthode utilisée, avec des données réelles, on a toujours des descripteurs
752 proches dans l'espace qui se retrouvent dans des partitions différentes. Cela a donc un impact négatif sur
753 la qualité de la recherche si l'on effectue cette recherche dans une seule partie de l'espace. La figure 2.9
754 page suivante illustre ce problème de partitionnement. On verra dans la section 2.4.2 page suivante
755 comment pallier à ce problème.

756 2.4 Structures d'indexation et recherche par similarité

757 Etant donné une méthode de partitionnement des données, il est maintenant possible d'indexer les
758 structures pour pouvoir les retrouver efficacement. Dans cette optique, il est nécessaire de définir une
759 structure adaptée au type de recherche effectuée.

760 Nous allons donc étudier les structures adaptées aux différents modes de partitionnement et les
761 méthodes de recherche associées.

762 Pour cela, nous allons commencer par présenter les structures d'indexation.

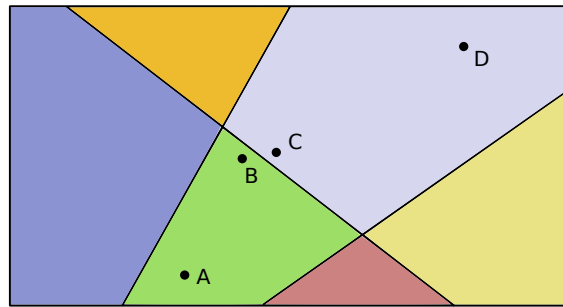


FIGURE 2.9 – Illustration du partitionnement « brutal ». Les points B et C sont proches dans l'espace mais ne sont pas dans la même partie de l'espace.

763 2.4.1 Les structures

764 2.4.1.1 Tables de hachage et listes inversées

765 La plupart des méthodes emploient les mêmes structures basiques :

- 766 – Une première structure de type « élément / clé », qui à un vecteur x associe une partie de l'espace de l'espace : une clé de hachage, un numéro de *cluster*, etc. ;
- 767
- 768 – Une deuxième structure de type « clé / éléments », qui à une partie de l'espace associe un ensemble de vecteurs appartenant à cette même partie. On appelle cette deuxième structure
- 769 « table de hachage » dans le cas des méthodes utilisant le hachage, ou « listes inversées » pour les
- 770 méthodes à base de *clustering* ;
- 771

772 Ce type de structures concerne les méthodes de partitionnement à base de hachage, de

773 regroupement, ou de structures déterministes.

774 Ces structures peuvent être implémentées par le biais de tableaux (coût mémoire élevé, mais accès

775 très rapide), ou de conteneur associatif de type « map » (coût mémoire réduit, mais accès plus lent).

776 2.4.1.2 Les structures hiérarchiques

777 Le principe des structures hiérarchiques est de donner l'accès à un ensemble de données respectant

778 un ensemble de critères, ces critères étant définis à chaque niveau de la hiérarchie. Dans les cas d'un

779 *KD-tree* [74], chaque nœud définit une partie de l'espace, dont la taille réduit à mesure que l'on descend

780 dans l'arbre. Les feuilles de l'arbre permettent donc d'obtenir l'équivalent des tables de hachage ou des

781 listes inversées. Les structures hiérarchiques sont, comme leur nom l'indique, adaptées aux méthodes de

782 partitionnement hiérarchiques.

783 2.4.2 Recherche dans une structure d'index

784 Il existe différentes méthodes de recherche dans une structure d'index : la recherche par accès simple,

785 et celles qui visent à résoudre le problème de la perte d'information lors du partitionnement.

2.4.2.1 Recherche par accès simple

La recherche par accès simple dans une structure d'index opère en deux phases :

- Identifier le numéro de la partie de l'espace dans laquelle se trouve le descripteur requête ;
- Obtenir la liste des descripteurs similaires : ceux référencés par cette partie de l'espace ;

Le descripteur peut (ou pas) faire partie de la base de données. Si tel est le cas, alors on connaît généralement déjà son numéro de partie d'espace, stocké dans une structure de type « élément /clé ». Sinon, le processus dépend de la méthode de partitionnement utilisée. Pour les méthodes de hachage ou déterministes, le problème est très simple puisqu'il s'agit simplement d'appliquer la fonction mathématique utilisée lors du partitionnement.

Pour les méthodes de type *clustering*, la méthode est plus coûteuse étant donné qu'il est nécessaire d'exécuter l'étape d'affectation au *cluster* le plus proche.

Pour les méthodes à base d'arbres, comme *KD-tree*, le descripteur requête doit être évalué à chaque niveau de la hiérarchie pour savoir quel branche emprunter. Cette méthode est efficace pour la recherche dans des espaces de petite dimension. Par contre, les performances se dégradent rapidement avec l'augmentation de la dimension. Au delà de 10 dimensions, Weber et al. [99] montrent que la recherche exhaustive est plus rentable.

La recherche par accès simple est la plus rapide de celles présentées dans ce document, mais elle est aussi la moins efficace en termes de qualité de recherche, puisqu'elle ne permet pas de résoudre le problème de perte d'information lors du partitionnement vu dans la section 2.3.3 page 35. Elle ne permet généralement pas de faire du contrôle de qualité.

C'est pourquoi on préférera se tourner vers des solutions comme la recherche par accès simple dans de multiples structures, la recherche par accès multiples, ou encore vers du *soft-assignment*, dont on détaillera les avantages et inconvénients dans les sections suivantes.

2.4.2.2 Recherche par accès simple dans de multiples structures

Comme son nom l'indique, la stratégie des structures multiples vise généralement à effectuer plusieurs partitionnements des données en introduisant une part d'aléatoire dans le processus de construction de chaque index. Étant donné que bon nombre de méthodes de partitionnement possèdent déjà une part d'aléatoire, il s'agit souvent simplement de construire plusieurs partitions.

Pour les méthodes déterministes, le principe est de faire vibrer (introduire une translation aléatoire) la structure utilisée.

Pour les méthodes à base de projections aléatoires, il suffit simplement de répéter plusieurs fois le partitionnement [63], afin d'augmenter le rappel. Dans *Frequency Based Locality Sensitive Hashing* [100], Ling et Wu proposent de calculer les plus proches voisins seulement sur les points ayant un nombre minimum de collisions avec la requête, à travers les tables.

Dans les méthodes de type *KD-tree (multiple randomized KD-tree)*, l'idée est d'introduire de l'aléatoire lors du choix de la dimension qui va être séparée, ou lors de la frontière (ne plus prendre le médian, mais une valeur aléatoire autour du médian) [78, 101]. Durant la recherche, une seule file de priorités

est maintenue pour tous les *KD-tree* aléatoires. Le compromis d'approximation est réglé par un nombre de nœuds parcourus. Après que ce nombre fixé de nœuds est atteint, la recherche se termine et les meilleurs résultats sont retournés. La méthode à base de *KD-tree* aléatoires est l'une des plus performantes méthodes à base d'arbre et est très utilisée dans le domaine de la recherche approximative de descripteurs locaux. Par contre, cette méthode n'a pas de paramètres de contrôle basé sur un pourcentage de qualité par rapport à une recherche exhaustive. De plus, à large échelle, les structures d'arbres multiples ont un coût mémoire important.

Dans les méthodes à base de *clustering*, il est possible de modifier l'initialisation des centres des *clusters*, ou comme l'ont fait Aly et al. dans [102], de faire plusieurs *clustering* en utilisant des sous-ensembles de données.

Bien que l'approche de la recherche dans de multiples structures résolve en partie le problème des erreurs de partitionnement, un autre problème se pose : celui du coût mémoire. Le coût unitaire est en effet multiplié par le nombre de structures, et réduit d'autant le nombre maximum de descripteurs gérables par une machine. Il a par exemple été montré [103] que le coût mémoire de LSH (en multi-tables) est en fait sur-linéaire en fonction de la taille de la base.

2.4.2.3 Le soft-assignment

Le *soft-assignment* est une méthode utilisée dans le *clustering* qui consiste à affecter un même vecteur à plusieurs *clusters*, en donnant un poids pour chaque affectation. Elle a été appliquée pour la première fois à des descripteurs locaux quantifiés en mots visuels dans [104]. Les auteurs de [104] obtiennent une réelle progression de la qualité de recherche.

Tout comme l'approche précédente, le coût mémoire augmente linéairement avec le nombre d'affectations par descripteur.

2.4.2.4 Recherche par accès multiples

Pour réduire le coût mémoire qui augmente proportionnellement avec le nombre de structures utilisées, certaines méthodes proposent d'accéder aux parties de l'espace proches de celle contenant la requête. La difficulté est alors de déterminer les parties de l'espace qui ont la meilleure probabilité de contenir des vecteurs similaires.

Une première approche est celle proposée dans [105] où les auteurs proposent une stratégie d'accès multiples dans des courbes de Hilbert.

En ce qui concerne les méthodes de hachage, [106, 103] ont introduit des méthodes de recherche alternatives (*Multi-Probe LSH*) permettant de diminuer de manière drastique le nombre de tables, d'un facteur allant de 10 à 40. La stratégie *Multi-Probe* consiste à chercher les descripteurs similaires dans une ou plusieurs clés de hachage par table.

Dans [106], l'idée est de fixer un nombre d'accès (*probes*), et de faire des requêtes de type « rayon », tandis que [103] préfère fixer une qualité de recherche, et de s'adapter à des requêtes quelconques lors de la phase d'apprentissage (y compris des requêtes de type KNN).

859 Dans [93], Jégou et al. commencent par filtrer les vecteurs candidats grâce à une liste inversée,
860 construite à partir d'un *clustering K-Means* (avec un assez faible nombre de *clusters*) sur l'ensemble des
861 vecteurs. Ils utilisent une stratégie d'affectation multiple (c'est-à-dire à plusieurs centroïdes) pour réduire
862 l'effet du partitionnement « brutal » (problème vu dans la section 2.3.3 page 35). Ils obtiennent ainsi un
863 ensemble de vecteurs candidats sur lesquels ils peuvent calculer une distance précise de façon rapide,
864 grâce à leur quantification sur des sous-ensembles de composantes.

865 Dans le cas des *KD-tree*, [107] a proposé d'introduire une variable ϵ de contrôle de la qualité pour
866 approximer les plus proches voisins. Ils ont aussi proposé une file de priorités permettant d'accélérer la
867 recherche à travers l'arbre. Cette file de priorités leur permet de visiter les nœuds suivant leur distance
868 par rapport à la requête courante.

869 2.4.2.5 Recherche avec calcul de distances asymétriques

870 Dans [108], Dong et al. proposent de calculer les plus proches voisins avec des distances asymétriques,
871 c'est-à-dire entre une requête non quantifiée (dans son espace d'origine), et les vecteurs de la base
872 ayant subi un partitionnement. Ce type de recherche, également repris par Jégou et al. [93], améliore
873 sensiblement la qualité de recherche.

874 2.4.2.6 Analyse des méthodes de recherche

875 Comme on l'a vu précédemment, les systèmes états de l'art utilisent principalement le *soft-assignment*
876 et les accès multiples, stratégies visant à réduire le problème du partitionnement « brutal ». En regardant
877 attentivement, on peut observer que ces deux stratégies sont en réalité très proches. Le *soft-assignment*
878 pourrait en effet être vu comme une stratégie d'accès multiples calculés *offline*, tandis que l'accès
879 multiple pourrait, lui, être considéré comme la version *online* du *soft-assignment*. *Product Quantization*
880 [93] emploie d'ailleurs une méthode proche de ces deux approches.

881 2.5 Vérification de la cohérence géométrique

882 Les méthodes de vérification géométrique servent à supprimer les fausses alarmes qui sont présentes
883 dans la liste des résultats formée par la recherche par similarité. Les faux appariements (*outliers*), à
884 opposer aux bons appariements (*inliers*) peuvent en effet être très nombreux, étant donné que les points
885 d'intérêts encodent une information très locale. Il est donc normal que la recherche par similarité donne
886 un très grand nombre d'*outliers* : de très nombreuses images peuvent présenter des zones de quelques
887 pixels très similaires à une zone de la requête effectuée, sans pour autant contenir un objet similaire.

888 Une grande partie des méthodes de vérification géométrique se basent sur l'estimation de la
889 transformation géométrique entre deux listes de points d'intérêts appariés (provenant de deux images).
890 Cependant, on trouve également des méthodes de cosegmentation qui préfèrent s'affranchir des points
891 d'intérêts pour descendre au niveau « pixel ». On trouve également des méthodes utilisant des contraintes
892 géométriques faibles afin de gagner en temps de calcul.

893 2.5.1 Contraintes géométriques faibles

894 Les méthodes dites de géométrie faible (*Weak Geometry*) consistent à garantir un minimum
895 de cohérence géométrique entre les appariements. Pour cela, [109] construit des histogrammes
896 représentant la différence d'angle et le rapport d'échelle des appariements de descripteurs locaux, pour
897 chaque couple d'images. Un pic dans l'histogramme tend ainsi à prouver la présence d'une transformation
898 géométriquement cohérente au sein des points appariés. Le score d'appariement entre la requête et
899 chaque image candidate peut donc être recalculé en fonction des histogrammes construits. Dans [110],
900 les auteurs introduisent la translation en plus de la différence d'échelle et d'angle.

901 Xie et al. [111] pointe trois problèmes majeurs de ce type de technique :

- 902 – Comme la plupart des points d'intérêt ont une échelle basse, l'histogramme des différences
903 d'échelle présente souvent un pic dans les valeurs basses ;
- 904 – La méthode ne prend pas en compte les transformations multiples ;
- 905 – Elle ne prend pas non plus en compte les changements de point de vue, ou encore les
906 transformations non rigides ;

907 Xie et al. [111] propose donc l'utilisation de paires d'appariements et montre une amélioration
908 significative des résultats.

909 2.5.2 Méthodes basées sur l'estimation de la transformation géométrique

910 Ces méthodes essaient d'estimer les paramètres d'une transformation géométrique entre deux
911 images (entre leur points appariés). Cette transformation est généralement affine, homographique, ou
912 plus simplement isométrique ou similaire.

913 En entrée, on a donc une liste d'appariements de points d'intérêts dont on connaît en général la
914 position dans l'image, l'échelle de détection et l'angle. En sortie, elles donnent un sous ensemble de ces
915 appariements, respectant la meilleure des transformations estimées.

916 Comme il est précisé dans la thèse de Rabin [112], trois approches sont envisageables afin d'estimer
917 cette transformation : les estimateurs robustes des moindres carrés, la transformée de Hough et RANSAC.

918 2.5.2.1 Estimateurs robustes des moindres carrés

919 Toujours selon [112], l'approche la plus simple pour estimer directement la transformation
920 d'un ensemble de correspondances est la méthode des moindres carrés. Elle consiste à définir la
921 transformation optimale minimisant l'erreur, définie comme la somme des résidus au carré de chaque
922 couple de points appariés. Cette approche est d'ailleurs utilisée par Lowe dans [23]. Le principal problème
923 de ces estimateurs provient du calcul de l'erreur quadratique qui la rend très sensible aux outliers.
924 L'estimation est perturbée dès lors qu'il y a ne serait-ce qu'un seul outlier.

925 Différentes améliorations proposent de calculer l'erreur médiane [113], ou de fixer le nombre
926 d'échantillons à utiliser dans le calcul de l'erreur [114].

927 Il faut cependant rappeler que nous cherchons ici à étudier des méthodes potentiellement capables
928 de retrouver des objets composés d'une dizaine de points d'intérêts dans des images en comportant

929 des milliers. Le nombre d'inliers est donc généralement très faible, voire nul lorsque les deux images
930 appariées ne comportent aucun objet en commun.

931 Des méthodes basées sur un consensus comme Hough ou RANSAC sont donc bien plus adaptées à
932 notre problème.

933 2.5.2.2 Transformée de Hough

934 La transformée de Hough, originellement introduite dans [115] est généralisée dans [116] pour
935 la détection de droites. Dans cette approche, l'espace des paramètres (ceux de la transformation
936 géométrique) est quantifié selon une grille régulière, définissant ainsi des accumulateurs. Les données
937 sont ensuite utilisées pour voter dans ces accumulateurs. En recherchant le mode dans les accumulateurs,
938 on trouve la transformation faisant consensus. Cette méthode est également utilisée par Lowe dans [23].

939 Outre la relativement faible précision de l'estimation, le principal inconvénient de Hough est le
940 problème de la complexité qui explose lorsque le nombre de paramètres excède quatre.

941 A cause de cette limitation, on préfère généralement utiliser l'algorithme RANSAC et ses variantes.

942 2.5.2.3 RANSAC

943 RANSAC (*RANdom SAmple Consensus*) [117] cherche un consensus au sein d'un ensemble
944 d'échantillons choisis aléatoirement. A chaque itération, une transformation est estimée à partir de
945 quelques échantillons choisis aléatoirement, et le consensus est ensuite mesuré sur l'ensemble des
946 données. L'algorithme s'arrête lorsqu'un consensus acceptable a été trouvé, qu'un nombre défini
947 d'itérations a été atteint, ou alors que toutes les combinaisons d'échantillons possibles ont été testées.

948 De très nombreuses variantes de RANSAC ont été présentées depuis la publication de l'algorithme
949 original. Une grande partie de ces variantes s'attachent à accélérer l'algorithme. RANSAC est en effet assez
950 coûteux en temps de calcul, en raison de la nécessité de faire de nombreuses itérations pour être sûr de
951 trouver un bon consensus. Certaines méthodes ont donc été proposées pour générer des hypothèses
952 (échantillons aléatoires) qui sont plus enclins à faire consensus. Par exemple, dans [118, 119, 120], le but
953 est de choisir les échantillons en exploitant les résultats des hypothèses précédentes. Dans ORSA [121]
954 (*Optimal Random Sampling*), lorsqu'une hypothèse est jugée suffisante, les échantillons sont choisis dans
955 le meilleur sous-ensemble trouvé. Dans [122, 123, 124], les auteurs choisissent les échantillons dans un
956 même voisinage spatial pour maximiser la probabilité que les échantillons soient tous des *inliers*.

957 Dans SCRAMSAC [125], une étape de vérification de la cohérence spatiale est effectuée afin de réduire
958 le nombre d'appariements, en conservant les plus cohérents.

959 Toujours dans l'idée de sélectionner des échantillons avec la meilleure probabilité possible,
960 l'algorithme PROSAC [126] favorise les échantillons qui ont été appariés avec le meilleur score de
961 confiance.

962 D'autres méthodes s'attachent davantage à contrôler la qualité des groupes de correspondances. Dans
963 [127], l'algorithme MLESAC fait l'hypothèse d'une distribution normale des *inliers* et uniforme des *outliers*.
964 Il introduit une nouvelle fonction de coût et une méthode pour trouver l'optimum avec l'algorithme EM

965 [128]. MINPRAN [129] fait lui simplement l'hypothèse de la distribution uniforme des *outliers*.

966 AC-RANSAC [121], introduit une méthode *a contrario* [130] afin de fixer les paramètres de RANSAC,
967 dont le réglage peut s'avérer très sensible.

968 Enfin, l'algorithme MAC-RANSAC [131] généralise AC-RANSAC et propose une méthode capable de
969 gérer les détections multiples, toujours grâce à des méthodes *a contrario*.

970 2.5.3 Méthodes basées cosegmentation

971 Le deuxième type de méthodes de vérification de la cohérence géométrique se base sur les pixels des
972 images [132, 133]. Ces méthodes cherchent à la fois à estimer la transformation mais aussi à segmenter
973 les deux images appariées afin de déterminer les zones communes. Ces méthodes permettent d'avoir une
974 estimation très précise de la transformation entre les deux images. La cosegmentation donne d'excellents
975 résultats, et permet en outre de réaliser une segmentation précise des objets détectés.

976 La cosegmentation suppose toutefois que le système puisse accéder aux pixels des images. Et bien
977 évidemment, un tel accès entraîne obligatoirement un temps de calcul conséquent. Pour ces raisons,
978 notre choix ne se portera donc pas sur ce type de méthodes, mais sur des méthodes de type RANSAC.

979 2.6 Extension de requête

980 Le concept de *Query Expansion* (QE, ou extension de requêtes) a été introduit dans le domaine de la
981 recherche d'objets visuels dans [134]. Le but des méthodes en découlant est d'améliorer le rappel des
982 objets recherchés. Cette méthode est inspirée des travaux effectués dans le domaine textuel utilisant
983 des vecteurs de mots [135, 136]. Cette approche était tout à fait naturelle du fait que cette première
984 proposition est basée sur les sacs de mots visuels. La méthode d'extension de requête à partir de sacs de
985 mots peut être décrite de la manière suivante : à partir d'une première requête est obtenu un premier
986 ensemble de résultats classés par leurs scores de similarités (par exemple à base d'un produit scalaire sur
987 les vecteurs de TF-IDF). A chaque résultat est associé une image décrite par son propre vecteur de mots
988 visuels. A partir de cet ensemble, il est possible de fusionner les différents vecteurs de mots visuels
989 résultats afin de créer une nouvelle requête qui est ensuite exécutée dans le moteur de recherche. Ce
990 processus peut être itéré autant de fois que désiré.

991 Dans le cas des requêtes globales à base de vecteurs de mots visuels, il a été proposé différentes
992 stratégies de fusion afin de formuler la nouvelle requête [134, 137, 138]. Les différentes stratégies de
993 fusion peuvent être décomposées en deux groupes : les génératives et les discriminantes. Les stratégies
994 génératives [134, 137] consistent à créer une nouvelle requête en calculant la somme des vecteurs
995 résultats. Les différentes méthodes génératives varient selon les méthodes de filtrages de résultats. La
996 stratégie discriminante, proposée dans [138], consiste à créer un classificateur discriminant à partir d'un
997 SVM appris sur les vecteurs résultats de la première requête. Le classificateur sert ensuite à classer les
998 images de la base. Les images ayant un score positif sont gardées comme résultats. Pour l'apprentissage
999 du SVM, les échantillons positifs sont les vecteurs les plus similaires du premier ensemble de résultats et
1000 les échantillons négatifs sont les vecteurs les moins similaires.

1001 Pour les requêtes locales, une stratégie de QE a été proposée dans [4]. Dans cette stratégie, la liste de
1002 résultats est considérée comme une nouvelle liste de requêtes à exécuter. Cette liste de résultats peut
1003 potentiellement contenir des fausses alarmes. En effet, si l'extension de requêtes a pour effet d'améliorer
1004 le rappel, il est primordial de vérifier qu'elle ne dégrade pas la précision. Afin d'éviter d'exécuter les
1005 requêtes « fausses alarmes », les auteurs de [4] proposent un seuillage adaptatif permettant d'avoir une
1006 sélection très précise des résultats de la requête initiale. Cette stratégie de seuillage adaptatif est basée
1007 sur une méthode a contrario [130].

1008 Ces méthodes d'extension requêtes [134, 137, 4] sont essentielles pour garantir un rappel important
1009 dans le cas de la recherche d'objets visuels. Elles entraînent par contre un sur-coût calculatoire qui
1010 peut atteindre des facteurs supérieurs à dix. Ces méthodes doivent donc être utilisées lorsqu'elles sont
1011 essentielles. Par exemple, elles ne sont pas essentielles lors des phases de découvertes d'objets visuels,
1012 mais elles le sont dans le cas où l'on veut réaliser des statistiques précises.

1013 2.7 Recherche d'objets visuels à grande échelle

1014 Le tableau 2.1 page 45 présente quelques unes des meilleures solutions actuelles de recherche
1015 d'images par le contenu. La performance de ces systèmes est évaluée par le score de mAP (mean Average
1016 Precision) obtenu sur la base Oxford Buildings⁴ (qui fait référence dans ce domaine), ainsi que sur la base
1017 BelgaLogos⁵ (qui permet l'évaluation de la recherche de petits objets).

1018 Ce tableau montre que la plupart des systèmes sont basés sur l'approche par sacs de mots visuels
1019 (BoW) [3], dont les mots visuels ont été créés avec un *clustering* par *K-Means* approximatif (AKM) [94].
1020 Bien que AKM rende possible la création de larges vocabulaires, ce type de méthode pose tout de même
1021 certains problèmes. Premièrement, la complexité d'AKM reste sur-linéaire en fonction de la taille de la
1022 base, contrairement aux méthodes de hachage. Deuxièmement, la finesse du partitionnement n'est
1023 pas réglable a posteriori. Il faut donc réeffectuer un *clustering* pour changer la taille du vocabulaire,
1024 contrairement aux méthodes de hachage qui peuvent calculer de longues clés et en utiliser une
1025 sous-partie. Troisièmement, les méthodes utilisant *K-Means* ont des performances très différentes selon
1026 que le partitionnement est effectué indépendamment ou non des données.

1027 On remarque que le descripteur SIFT [23] est le plus utilisé, mais que son calcul est effectué autour
1028 de points d'intérêt détectés par la méthode Hessian affine de Mikolajczyk [27], ou plus récemment par
1029 l'amélioration de Perdoch [28].

1030 Tous les systèmes étudiés emploient une vérification géométrique des appariements entre
1031 descripteurs locaux. La plupart emploie également une méthode d'extension de requêtes. On constate
1032 que l'emploi de ces méthodes améliore nettement les résultats. Malgré cette efficacité, l'extension de
1033 requête reste généralement très coûteuse en temps de calcul, la rendant souvent inappropriée à une
1034 utilisation *online*.

1035 Malgré de nombreux téléchargements de la base BelgaLogos (environ une centaine d'équipes de

4. <http://www.robots.ox.ac.uk/vgg/data/oxbuildings/>

5. <http://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/BelgaLogos.html>

1036 recherche à ce jour), peu de travaux ont été publiés. Seul Revaud et al. [139] en 2012 ont récemment
1037 battu le score original de Joly et Buisson en 2009. Ceci est probablement dû au fait que la plupart des
1038 approches sont basées sur l'utilisation de sacs de mots visuels. Cette approche consistant à créer une
1039 description globale des images à l'aide de descripteurs locaux, elle est de fait relativement inadaptée
1040 à la recherche de petits objets (représentant une très faible proportion des descripteurs). On constate
1041 d'ailleurs que Revaud et al. ont préféré utiliser la méthode Hamming Embedding [109] plutôt que des sacs
1042 de mots visuels.

1043 Un des travers de la méthode de Revaud et al. [139] est qu'elle requiert une base d'images
1044 « négatives » (images ne comportant aucunes des instances à rechercher) pour réaliser l'apprentissage
1045 de la *correlation-based burstiness*. Une telle base est en pratique difficile à construire dès lors que l'on
1046 souhaite ajouter de nouvelles instances à rechercher : il faut parcourir toutes les images afin de vérifier
1047 que ces instances ne sont pas présentes dans la base de négatifs. La méthode est donc efficace pour les
1048 logos considérés, mais elle n'est pas automatiquement générique.

1049 Toujours dans [139], les auteurs sont les premiers à soulever un problème qui, pour certains logos,
1050 limite fortement le rappel. Ce problème est celui des logos existants en deux versions (noir sur blanc ou
1051 blanc sur noir, ou plus généralement clair sur sombre et inversement). Ceci est le cas des logos Adidas,
1052 Nike ou Puma par exemple. Le descripteur SIFT n'étant pas invariant à l'inversion de couleur, Revaud et al.
1053 proposent donc de doubler chaque requête, c'est-à-dire d'effectuer la requête normale, et également la
1054 requête en inversion de couleur.

Système	Description	Partitionnement	Indexation & Recherche	Vérification géométrique	Extension de requêtes	mAP Oxford Buildings				mAP BelgaLogos	
						P. indep		P. dep		P. dep / P. indep	
						QE	QE	QE	QE	QE	QE
Philbin et al. [104] (2008)	Hessian affine [27] + SIFT [23]	AKM [94]	BoW + tf-idf + SA	RANSAC	Average expansion [134]	0,598	0,718	0,731	0,825		
Joly et al. [4] (2009)	SIFT [23]	LSH [63]	APMPLSH [103]	RANSAC	QE avec seuillage adaptatif a contrario	0,608	0,807			0.208	0.341
Arandjelovic et al. [138] (2012)	Perdoch [28] + RootSIFT [138]	AKM [94]	BoW + tf-idf	RANSAC	Discriminative QE		0,809		0,929		
Perdoch et al. [28] (2009)	Perdoch [28]	AKM [94]	BoW + tf-idf + SA	RANSAC	?	0,725	0,822	0,846	0,916		
Chum et al. [137] (2011)	Hessian affine [27] + SIFT [23]	AKM [94]	BoW + tf-idf	RANSAC	Contextual QE	0,785	0,82				
Notre méthode (2012)	SIFT [23]	RMMH [90]	APMPLSH [103] + HE	RANSAC	QE avec seuillage adaptatif a contrario			0,851	0,896	0,419	0,51
Revaud et al. [139] (2012)	Hessian affine [27] + SIFT [23]	K-Means	Liste inversée + HE [109]	RANSAC + Correlation-based burstiness	QE					0,414	

TABLE 2.1 – Comparaison des systèmes de recherche d'images par le contenu sur Oxford Buildings et BelgaLogos.

BoW = Bag of visual Words (Sacs de mots visuels), tf-idf = term frequency - inverse document frequency, SA = soft assignment, HE = Hamming Embedding, QE = Query Expansion (Extension de requêtes), ~~QE~~ = sans extension de requêtes, mAP = mean Average Precision, P. dep/indep = Partitionnement dépendant/indépendant des données

1055 Chapitre 3

1056 État de l'art des méthodes de découverte 1057 et fouille d'objets visuels

1058 Dans le chapitre précédent, nous avons détaillé les méthodes utilisées dans les systèmes de recherche
1059 d'images par le contenu. Comme nous l'avons vu, cette recherche vise à retrouver des objets similaires
1060 entre une image requête, et une collection d'images ou de vidéos, à la demande d'un utilisateur, ou
1061 éventuellement d'un système automatique.

1062 Contrairement à la recherche d'images, la découverte (ou fouille) d'images est un processus
1063 automatique, dans lequel l'utilisateur ne donne aucune information sur les objets à rechercher (exceptés
1064 quelques éventuels réglages). L'objectif est d'extraire une connaissance à partir de grandes quantités
1065 de données, trop grandes pour être rapidement analysées par un opérateur humain. Dans le domaine
1066 visuel, il s'agit donc de faire émerger des informations visuelles permettant à un utilisateur d'effectuer
1067 une analyse ou un traitement en un temps raisonnable.

1068 Les informations à extraire dépendent directement du cas d'utilisation. Ces cas peuvent être divers
1069 et variés, allant du résumé de collections à la reconstruction d'objets en 3D. On remarque que pour
1070 la plupart de ces cas d'utilisation, on porte une attention particulière aux objets dont la taille et la
1071 fréquence sont importantes. Techniquement, on peut considérer deux problèmes principaux. Le premier
1072 est celui de la découverte d'objets, qui consiste à détecter la présence d'un objet, c'est-à-dire à découvrir
1073 plusieurs (au moins deux) instances d'un même objet. Le deuxième est celui du regroupement (*clustering*)
1074 d'instances, qui vise à regrouper toutes les instances d'un même objet entre elles.

1075 Dans la section [4.1.2 page 56](#), nous formaliserons ces deux problèmes, que nous appellerons
1076 « Découverte » et « Fouille » d'objets $\{c, f\}$ -fréquents, où l'on caractérisera les objets par leur couverture
1077 spatiale c , et leur fréquence d'apparition f .

3.1 Méthodes de fouille d'objets visuels par regroupement de motifs visuels

Sivic et Zisserman [140] proposent un système capable de détecter les objets principaux à l'intérieur d'une vidéo, en mesurant la fréquence d'apparition des configurations spatiales des mots visuels. Ces mots visuels sont des SIFT ou des MSER, quantifiés par un *K-Means*. Ces descripteurs peuvent ensuite être regroupés entre voisins pour former des régions plus grandes nommées voisinages. Ces voisinages sont définis comme étant l'enveloppe convexe des KNN spatiaux d'un descripteur central. Les voisinages étant représentés par des sacs de mots visuels, ils peuvent être comparés en calculant leur produit scalaire. L'algorithme opère en trois étapes. Premièrement, les voisinages sont filtrés afin de ne garder que ceux dont la fréquence d'apparition (nombre d'images) est supérieure à un certain seuil. Pour accélérer cette étape, seuls les voisinages présentant le même mot visuel central sont comparés. Deuxièmement, les voisinages restants sont regroupés par un algorithme de *clustering* progressif. Pour finir, les *clusters* générés à l'étape 2 sont fusionnés en fonction de leur recouvrement spatial et temporel. Cette approche est adaptée à la découverte d'objets à l'intérieur d'un petit corpus (comme un film), mais absolument pas à de grandes bases. En effet, la complexité de la première étape de l'algorithme reste en $O(\sum_{i=1}^m n_i^2)$, avec m le nombre total de mots visuels, et n_i le nombre de fois que le mot visuel i apparaît dans la base.

Comme dans [140], Quack et al. [141] recherchent les configurations de mots visuels fréquentes. La différence est qu'ils emploient une stratégie de recherche d'ensembles de motifs fréquents (*frequent itemsets*) utilisant l'algorithme Apriori [142], à laquelle ils ajoutent la localisation spatiale des mots visuels. Concrètement, les mots visuels sont filtrés sur leur fréquence et leur stabilité temporelle, puis on crée des transactions à partir de chaque mot restant. Pour une transaction, on a donc un mot visuel « central » et des KNN spatiaux, comme dans [140]. Ces KNN sont classés en quatre quadrants spatiaux afin de stocker la localisation relative des KNN. Quack et al. [141] tirent également profit du mouvement des objets dans les vidéos pour réaliser une segmentation, et ainsi filtrer les mots visuels se déplaçant indépendamment dans le temps.

Yuan et al. [143, 144] emploient également une approche de type *frequent itemsets*. Ils proposent de mesurer la fréquence des *itemsets* relativement à une génération aléatoire d'*itemsets*. La localisation spatiale est également prise en compte, et les auteurs proposent de refaire un parcours de la base pour identifier les transactions comptées plusieurs fois au sein d'une même image. Le partitionnement des descripteurs PCA-SIFT est effectué par un *K-Means*. Ce *K-Means* peut être ré-effectué en utilisant une nouvelle métrique, apprise à partir de la phase d'identification des *frequent itemsets*. Pour finir, les *frequent itemsets* sont *clusterisés* à l'aide d'un algorithme de *normalized cut*.

Yuan et al [145] proposent un autre type d'approche, basé cette fois sur du hachage. Les descripteurs sont hachés avec LSH, ce qui permet ensuite d'effectuer des requêtes de type rayon dans l'espace des descripteurs. En combinant cette recherche avec une recherche de KNN dans l'espace image (spatial), les

1116 auteurs calculent un score de similarité avec d'autres images de la base, ce qui permet de découvrir
1117 des zones d'images fréquentes. Dans [146], Yuan et Wu effectuent plusieurs partitionnements spatiaux
1118 aléatoires des images afin de créer des zones d'images qui sont ensuite approximativement recherchées
1119 grâce à LSH. Un vote est ensuite appliqué pour déterminer les zones d'images les plus répétées.

1120 Tout comme [140], la complexité de ces méthodes de découverte d'*itemsets* fréquents n'est pas
1121 adaptée à de très grands corpus.

1122

1123 Dans [147, 148], Anjulan et Canagarajah segmentent leurs vidéos en shots et en extraient des *tracks*
1124 (descripteurs stables temporellement). Une méthode de regroupement basée sur la distance est ensuite
1125 appliquée pour rassembler les *tracks* appartenant aux mêmes instances d'objets. Cette méthode est ainsi
1126 adaptée à la découverte d'objets dans des vidéos, mais l'inclusion de l'information temporelle empêche
1127 l'algorithme d'être appliqué à un volume important de vidéos.

1128 3.2 Méthodes de construction de graphes d'appariement d'im- 1129 ages contenant des objets similaires

1130 Dans [82], Philbin propose de construire un graphe d'appariement complet, c'est-à-dire un graphe
1131 dont les nœuds sont les images et les arêtes représentent une similarité entre deux régions (vérifiées
1132 géométriquement) de ces images. Pour des raisons d'efficacité, la construction du graphe est effectuée à
1133 l'aide de mots visuels et d'une liste inversée. Chaque image est recherchée parmi les autres. Les 400
1134 meilleurs résultats de chaque requête sont ensuite vérifiés spatialement, et seules les images comptant
1135 au moins 20 appariements avec la requête sont conservées (reliées au nœud « requête » dans le graphe).
1136 On remarque que ce seuil minimum de 20 appariements ne permet pas la découverte de petits objets,
1137 pouvant être représentés par moins d'une dizaine de descripteurs visuels. La méthode n'étant pas
1138 particulièrement robuste, Philbin [82] propose d'utiliser des méthodes capables d'identifier des *clusters*
1139 au sein de ce graphe. Par exemple, détecter les composantes connexes (en complexité linéaire grâce à un
1140 algorithme de recherche en profondeur d'abord [149]), ou calculer des mesures de type PageRank [150]
1141 ou centralité [151]. Le principal problème de la construction d'un tel graphe est le temps de calcul. La
1142 complexité de l'algorithme est certes linéaire en nombre d'images, mais l'opération de recherche reste
1143 coûteuse, à moins de fortement dégrader la qualité au profit de la vitesse. De plus, cette méthode
1144 n'envisage pas la possibilité qu'il y ait plusieurs instances d'objets différents au sein d'une même images.

1145 3.3 Méthodes de découverte de concepts visuels fréquents par 1146 extraction de sujets latents

1147 Les modèles de sujets latents (*latent topic models*) issus du domaine textuel, tels que l'analyse
1148 sémantique latente probabiliste (pLSA) [152] et l'allocation de Dirichlet latente (LDA) [153] ont également
1149 été appliquées au domaine visuel [17, 154, 155], bien que l'objectif visé par ces travaux était de découvrir
1150 des catégories visuelles (voiture, chien, maison, ...) dans une collection d'images, plutôt que de découvrir

1151 des *clusters* d'instances d'objets. Ces méthodes se contentent donc de représenter les images comme des
 1152 sacs de mots visuels, sans inclure d'informations géométriques sur les descripteurs visuels. Ces modèles
 1153 représentent chaque image comme un mélange de K sujets, où chaque sujet correspond à une classe
 1154 d'objets.

1155 Dans [156], Philbin et al. introduisent un nouveau modèle génératif de sujets latents pour la
 1156 découverte d'instances d'objets, nommé gLDA (*Geometric LDA*). A la différence des modèles précédents
 1157 cités (pLSA, LDA), gLDA intègre des contraintes géométriques dans les sujets (position spatiale et forme
 1158 des descripteurs visuels), ainsi qu'une transformation géométrique entre les sujets et les documents.

1159 La principale limitation de ce type de modèles est que le nombre de sujets K doit être fixé par
 1160 l'utilisateur, ce qui est loin d'être évident lorsqu'on s'intéresse à une collection d'images inconnues. Le
 1161 choix du nombre de sujets est de plus un choix subjectif. Devant une photographie d'un bâtiment, doit-on
 1162 considérer que le bâtiment lui-même est un sujet, ou bien qu'il est composé de plusieurs sujets (fenêtres,
 1163 portes, etc.)? Les sujets eux-mêmes peuvent être ambigus, mélangeant plusieurs objets apparaissant
 1164 parfois ensemble.

1165 Le temps de calcul nécessaire à l'estimation des paramètres de ces modèles rend ce type d'approche
 1166 difficilement applicable à de grandes bases d'images.

1167 3.4 Approximation de graphes d'appariement d'instances d'ob- 1168 jets par hachage

1169 Dans ce type de méthode, les instances d'objets découvertes sont localisées dans les images (avec
 1170 une boîte englobante par exemple), en plus d'être détectées.

1171 Les solutions état de l'art existantes qui approximent un tel graphe d'appariement sont basées sur
 1172 min-Hashing.

1173 *Min-Hashing* [66] est un algorithme fréquemment employé dans la recherche textuelle pour trouver
 1174 des copies-proches [157, 158], et fonctionne en approximant l'intersection entre deux ensembles de
 1175 mots.

1176 Dans cette approche, chaque image est représentée sous forme d'un ensemble de mots visuels. Une
 1177 fonction de hachage aléatoire f_j assigne un nombre à chaque mot visuel, et impose ainsi un classement
 1178 spécifique à l'intérieur de chaque ensemble de mots visuels (typiquement chaque image). On peut alors
 1179 obtenir un *min-Hash* $m_j(A)$ dans chaque ensemble A en sélectionnant le mot visuel ayant la valeur
 1180 minimum, c'est-à-dire :

$$m_j(A) = \operatorname{argmin} f_j(A) \quad (3.1)$$

1181 Cette fonction a la propriété que la probabilité que deux ensembles \mathcal{A}_1 et \mathcal{A}_2 aient la même valeur de
 1182 *min-Hash* est égale au recouvrement de leur ensemble, c'est-à-dire le ratio de l'intersection et de l'union
 1183 de leurs ensembles :

$$\operatorname{ovr}_1(\mathcal{A}_1, \mathcal{A}_2) = \frac{|\mathcal{A}_1 \cap \mathcal{A}_2|}{|\mathcal{A}_1 \cup \mathcal{A}_2|} \in [0, 1] \quad (3.2)$$

1184 Cette mesure de similarité suppose que tous les mots ont la même importance. Il a été montré dans
 1185 [158] que la mesure de similarité pouvait être améliorée en utilisant une version pondérée. Soit $d_w \geq 0$
 1186 l'importance du mot visuel X_w . Le recouvrement pondéré des ensembles \mathcal{A}_1 et \mathcal{A}_2 est :

$$\text{ovr}(\mathcal{A}_1, \mathcal{A}_2) = \frac{\sum_{X_w \in \mathcal{A}_1 \cap \mathcal{A}_2} d_w}{\sum_{X_w \in \mathcal{A}_1 \cup \mathcal{A}_2} d_w} \in [0, 1] \quad (3.3)$$

1187 L'une des possibilités les plus utilisées pour choisir des poids d_w est l'approche TF-IDF (*Term Frequency*
 1188 *- Inverse Document Frequency*) [159].

1189 La probabilité que deux images aient la même valeur de *min-Hash* est égale à :

$$P\{m(\mathcal{A}_1) = m(\mathcal{A}_2)\} = \text{ovr}(\mathcal{A}_1, \mathcal{A}_2) \quad (3.4)$$

1190 Pour estimer le recouvrement des mots visuels entre deux images, de multiples fonctions *min-Hash*
 1191 indépendantes f_j sont utilisées. La fraction de fonctions *min-Hash* qui assignent une valeur identique
 1192 aux deux ensembles donne une estimation de la similarité entre ces deux images. Pour retrouver
 1193 efficacement les images ayant une forte similarité, les valeurs des fonctions *min-Hash* sont groupées
 1194 en s -tuples appelés *sketches*. On obtient de meilleurs taux de rappel en répétant k fois l'étape de
 1195 sélection des s -tuples. Un couple d'images est potentiellement appariable lorsqu'on obtient au moins une
 1196 collision entre leurs *sketches* respectifs, pour une même itération parmi les k itérations. La probabilité
 1197 qu'un couple d'images ait au moins un *sketch* en commun parmi les k générés est une fonction de leur
 1198 recouvrement :

$$P\{\text{collision}\} = 1 - (1 - \text{ovr}(\mathcal{A}_1, \mathcal{A}_2)^s)^k \quad (3.5)$$

1199 Plus on augmente la taille s des *sketches*, plus on diminue la probabilité de collision, mais plus on
 1200 augmente leur distinctivité (en pratique, il est conseillé de choisir $s \leq 3$). Et plus on augmente le nombre
 1201 de *sketches* k , et plus on augmente la probabilité de collision, sans autres conséquences particulières,
 1202 hormis le temps de calcul. Il est donc préférable d'augmenter k plutôt que s .

1203 *Geometric min-Hash* [160] apporte une amélioration dans la génération des *sketches* en intégrant
 1204 des contraintes géométriques, ou plus exactement des contraintes de voisinage. Les *sketches* sont
 1205 composés de deux parties : un descripteur central et un descripteur secondaire. Le descripteur central
 1206 est sélectionné par l'approche min-Hashing standard. La seule différence est que seuls les descripteurs
 1207 ayant des mots visuels uniques au sein de l'image sont considérés. Le descripteur secondaire est ensuite
 1208 sélectionné parmi les voisins du descripteur central, toujours par min-Hashing. L'idée sous-jacente est
 1209 que deux descripteurs proches spatialement ont beaucoup plus de chance de faire partie d'un même
 1210 objet (surtout si l'objet est petit), que deux descripteurs éloignés dans l'image.

1211 Les auteurs de GmH proposent deux applications de leur méthode de hachage : le *clustering* d'images
 1212 et la découverte de petits objets.

1213 Le *clustering* d'images fonctionne en détectant les composantes connexes au sein du graphe
 1214 d'appariement. Les appariements sont créés itérativement au fur et à mesure de la découverte de

1215 collisions entre les *sketches*. Les appariements potentiels sont vérifiés géométriquement dès lors qu'ils
1216 font intervenir au moins 20 descripteurs, comme cela était suggéré dans [82].

1217 La découverte de petits objets étant plus délicate (la probabilité de collision est très faible), la
1218 méthode de vérification des appariements est une cosegmentation [133], qui suppose d'avoir accès aux
1219 pixels, et qui est très coûteuse en temps de calcul.

1220 Le principal inconvénient de *Geometric min-Hashing* est qu'il faut générer beaucoup de *sketches* pour
1221 avoir une probabilité suffisante de toucher les objets ayant une faible couverture spatiale. La méthode est
1222 toutefois la seule à proposer une solution efficace et *scalable* pour découvrir des objets dans de très
1223 grandes bases d'images. Sa capacité à découvrir des petits objets est plus discutable, étant donné le
1224 manque d'information quant aux réglages des paramètres, et à l'absence d'évaluation quantitative, ou de
1225 temps de calcul.

1226

Deuxième partie

1227

Contributions

Chapitre 4

Fouille d'objets visuels fréquents par requêtage aléatoire

4.1 Définition formelle des problèmes de fouille et de découverte d'objets visuels fréquents

4.1.1 Notations

Soit I un ensemble de N_I images I_j , $j \in 1, \dots, N_I$, représentées par un ensemble X de N descripteurs locaux x_i , chacun étant extrait depuis une position $p_{ij} = (I_j, \chi_{ij}, \psi_{ij})$, où χ_{ij} et ψ_{ij} sont les coordonnées spatiales dans l'image.

Considérons maintenant un ensemble O d'objets O^m , chacun étant représenté par S_m instances O_s^m . Une instance O_s^m est associée à une zone unique A_s^m (dans une seule image), et contient un ensemble de descripteurs locaux :

$$X_s^m = \{x_i \mid p_{ij} \subset A_s^m\}_{1 \leq i \leq N}$$

Introduisons maintenant quelques définitions basiques qui nous aideront ensuite à formaliser les problèmes de découverte et fouille d'objets, et à modéliser une approche adaptée à la résolution de ces dits problèmes.

Définition - Couverture globale :

La couverture globale $c_X(O^m)$ d'un objet O^m est définie par :

$$c_X(O^m) = \frac{1}{N} \sum_{s=1}^{S_m} |X_s^m| \quad (4.1)$$

Elle mesure le pourcentage de descripteurs locaux dans X couverts par les instances d'un objet O^m donné.

1248 **Définition - Couverture moyenne :**

1249 La couverture moyenne $c(O^m)$ d'un objet O^m est définie par :

$$c(O^m) = \frac{1}{S_m} \sum_{s=1}^{S_m} \frac{1}{N_s^m} |\mathbf{X}_s^m| \quad (4.2)$$

1250 avec N_s^m le nombre de descripteurs locaux dans l'image incluant l'instance O_s^m .

1251 Elle mesure la taille moyenne d'un objet O^m (en pourcentage moyen de descripteurs locaux couverts
1252 par une instance de l'objet à travers toutes les images).

1253

1254 **Définition - Fréquence :**

1255 La fréquence $f(O^m)$ d'un objet O^m mesure le pourcentage d'images couvertes par une des instances de
1256 cet objet et est définie par :

$$f(O^m) = \frac{S_m}{N_I} \quad (4.3)$$

1257

1258

1259 **Définition - Objet $\{c, f\}$ -fréquent :**

1260 Un objet O^m est dit $\{c, f\}$ -fréquent si :

$$\begin{cases} c(O^m) = c \\ f(O^m) = f \end{cases}$$

1261 On remarque ici que si le nombre de descripteurs par image est assez stable, alors on peut considérer
1262 que :

$$c_{\mathbf{X}} \approx c.f$$

1263

1264

1265 4.1.2 Formalisation des problèmes de découverte et fouille d'objets fréquents

1266 A partir des définitions précédentes, il est maintenant possible de définir les deux problèmes suivants
1267 comme les deux principaux objectifs à atteindre par les méthodes de découverte et de fouille d'objets
1268 visuels.

1269 **Définition - Découverte d'objets :**

1270 Trouver au moins une instance O_s^m de tous les objets $\{c, f\}$ -fréquents O^m tels que :

$$\begin{cases} c \geq c_0 \\ f \geq f_0 \end{cases}$$

1271

1272

1273 **Définition - Fouille d'objets :**

1274 Trouver toutes les instances O_s^m de tous les objets $\{c, f\}$ -fréquents O^m tels que :

$$\begin{cases} c \geq c_0 \\ f \geq f_0 \end{cases}$$

1275

1276

1277 L'idée sous-jacente est que la complexité de ces deux problèmes dépend principalement des
1278 paramètres c et f . Nous suggérons donc d'inclure ces deux paramètres dans la formulation du problème,
1279 tout comme cela est fait dans les approches classiques de fouille de données (e.g. la fouille d'item-sets
1280 fréquents [161])

1281 4.2 Recherche itérative par échantillonnage pondéré adaptatif 1282 (RANSAS)

1283 La méthode de découverte proposée ici est un processus itératif que l'on nommera par la suite
1284 « RANSAS » (RANDOM Sample And Search model), dont chaque itération est composée de trois étapes
1285 principales. La première étape est celle de l'échantillonnage pondéré adaptatif d'une région locale
1286 d'image. La deuxième étape consiste en la recherche précise de la zone d'image sélectionnée. Enfin,
1287 la troisième étape est chargée de décider si oui ou non cette région et ses résultats devraient être
1288 considérés comme étant des instances d'un objet $\{c, f\}$ -fréquent. Étant donné que les régions ainsi
1289 appariées peuvent ne représenter qu'une partie d'un objet, nous parlons parfois de « germes » (*seeds*
1290 [160]), ou dans notre première publication sur le sujet [162] de « mots visuels géométriquement
1291 cohérents ». L'algorithme répète ces trois étapes T fois ($T \leq T_{\max}$, T_{\max} fixé par l'utilisateur), jusqu'à ce
1292 qu'un nombre d'objets $\{c, f\}$ -fréquents aient été découverts, ou qu'un temps limite ait été atteint.

1293 Au démarrage de l'algorithme, chaque descripteur local x_i est associé à une potentielle région
1294 requête R_i , définie comme un rectangle centré autour de la position $p_{i,j}$ et dont les dimensions H'_i et W'_i
1295 sont calculées à partir des dimensions de l'image I_j de la manière suivante :

$$H'_i = \sqrt{\beta} * H_j$$

$$W'_i = \sqrt{\beta} * W_j$$

1296 où β est un paramètre correspondant au pourcentage de la surface d'image couverte par une région
1297 requête candidate, et où W_j et H_j sont respectivement la largeur et la hauteur de l'image I_j . Par
1298 exemple, $\beta = 0.10$ signifie que les requêtes auront une surface correspondant à 10% de la surface de
1299 leurs images hôtes, soit une largeur et une hauteur correspondant à un tiers de celles de l'image. L'intérêt
1300 de ce paramètre est de pouvoir adapter la taille et la forme d'une requête à celle de l'image. Cette
1301 approche est une hypothèse naïve nécessaire afin de modéliser le problème dans un premier temps, et
1302 mériterait une plus grande attention dans le futur.

1303 Les trois étapes de l'algorithme RANSAS sont ensuite appliquées à chaque itération. Le schéma de la
 1304 figure 4.1 page ci-contre présente un récapitulatif du fonctionnement de RANSAS

1305 4.2.1 Échantillonnage pondéré adaptatif

1306 Pour éviter de requêter toutes les régions possibles tout en conservant une couverture maximum du
 1307 contenu des images, la stratégie proposée dans cette thèse repose sur un échantillonnage pondéré et
 1308 adaptatif ayant pour objectif de sélectionner, à chaque itération t , une région la plus pertinente possible
 1309 (en fonction du contenu de la base, et des itérations passées).

1310 Cette $t^{\text{ème}}$ région requête candidate, que l'on note R_q^t , est centrée autour d'un descripteur local x_q^t
 1311 sélectionné aléatoirement à l'aide d'une fonction de probabilité de masse $p_t(i)$, sur l'ensemble de tous
 1312 les descripteurs de la base. Un tel échantillonnage peut-être obtenu facilement depuis n'importe quelle
 1313 distribution p_t en utilisant la méthode de la transformée inverse [163] (également appelée transformée de
 1314 Smirnov). Cette méthode consiste à transformer un nombre aléatoire (uniforme) en intégrant la fonction
 1315 de probabilité de masse jusqu'à obtenir une valeur (surface) plus grande que le nombre transformé.

1316 L'échantillonnage est une méthode statistique visant à sélectionner une sous-partie d'un ensemble
 1317 d'individus dans une population, permettant d'estimer les caractéristiques de la population entière.
 1318 Lorsque tous les individus ont la même probabilité d'être sélectionnés, le problème est connu comme un
 1319 problème d'échantillonnage aléatoire *uniforme*.

1320 Dans les méthodes d'échantillonnage *pondéré* [164], les individus sont pondérés individuellement et
 1321 la probabilité de chaque individu est déterminée par son poids relatif.

1322 Dans les schémas conventionnels d'échantillonnage, *uniforme* ou *pondéré*, la sélection d'un individu
 1323 ne dépend pas des observations passées. Au contraire, dans un schéma d'échantillonnage *adaptatif* [165],
 1324 la stratégie consiste à adapter la sélection d'un individu en tenant compte des sélections précédentes.

1325 Notre méthode propose un échantillonnage *pondéré* et *adaptatif*. Elle débute à l'itération $t = 0$ avec
 1326 une fonction de probabilité de masse $p_0(i)$ couvrant l'ensemble des descripteurs (et donc l'ensemble
 1327 des régions requêtes candidates R_i). La distribution initiale peut être uniforme ou déterminée par une
 1328 connaissance a priori, comme discuté dans la section 4.3 page 62, et plus en détail dans les chapitres 5
 1329 page 71 et 6 page 81.

1330 Une fois appliquées les trois étapes de l'algorithme RANSAS, nous obtenons à l'itération $t = 0$ une
 1331 région requête R_q^0 et des régions résultats $R_m^0, m \in 1, \dots, M_0$ (si l'étape de décision en a retourné).

1332 A chaque itération, la fonction de probabilité de masse est mise à jour à l'aide d'une fonction f ,
 1333 prenant en paramètres les probabilités issues de l'itération précédente, la requête effectuée, ainsi que les
 1334 résultats :

$$p_t = f(p_{t-1}, R_q^{t-1}, \{R_m^{t-1}\})$$

1335 De même que dans les méthodes d'échantillonnage pondéré et adaptatif habituelles [164], la fonction
 1336 de probabilité de masse p_t est en pratique calculée en normalisant une fonction de pondération z_t . Nous
 1337 verrons dans la section 4.4 page 64 comment normaliser ces poids.

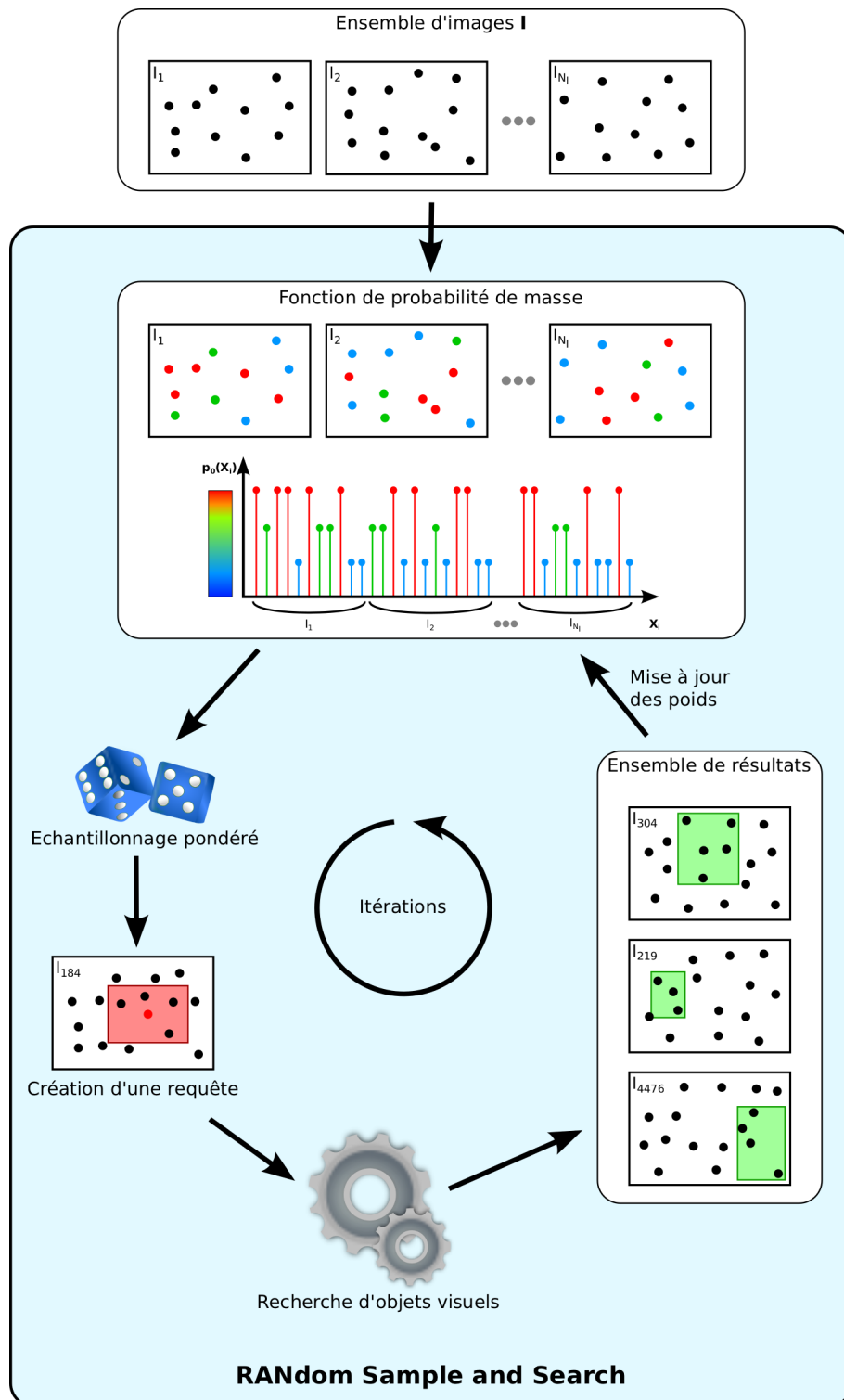


FIGURE 4.1 – Schéma récapitulatif du fonctionnement de l'algorithme RANSAS

1338 En pratique donc, la mise à jour des poids est effectuée de manière récursive :

$$z_t = g(z_{t-1}, R_q^{t-1}, \{R_m^{t-1}\})$$

1339 avec la fonction g définie telle que :

$$z_t(i) = \begin{cases} 0 & \text{si } \mathbf{x}_i = \mathbf{x}_q^{t-1} \\ \alpha_1 z_{t-1}(i) & \text{si } \mathbf{x}_i \in R_q^{t-1} \\ \alpha_2 z_{t-1}(i) & \text{si } \mathbf{x}_i \in \{R_m^{t-1}\}_m \\ z_{t-1}(i) & \text{sinon} \end{cases} \quad (4.4)$$

1340 La première condition garantit que le descripteur sélectionné pour créer la région requête candidate
1341 ne sera plus jamais sélectionné (à la manière d'un tirage sans remise).

1342 La deuxième condition diminue le poids des descripteurs appartenant à la région requête R_q^{t-1} , afin
1343 de diminuer leur probabilité d'être sélectionnés comme centres de nouvelles requêtes. Ceci permet
1344 d'éviter de créer de nouvelles requêtes ayant un trop fort recouvrement avec celles des itérations
1345 précédentes.

1346 La troisième condition vise à diminuer le poids des descripteurs appartenant à des régions résultats,
1347 toujours dans l'optique de créer de nouvelles requêtes différentes des précédentes. Cette condition a
1348 pour effet d'augmenter la probabilité de créer une requête à partir d'un objet différent, et non plus
1349 seulement d'une instance d'objet différente.

1350 Enfin, la dernière des conditions garantit que les poids des descripteurs qui ne sont pas intervenus lors
1351 de l'itération courante, ne seront pas modifiés.

1352 En pratique, α_1 et α_2 sont compris entre 0 et 1, avec $\alpha_1 < \alpha_2$.

1353 L'utilisation de faibles valeurs pour ces paramètres accélère le processus de découverte, en
1354 maximisant les chances de sélectionner des zones d'images nouvelles. Toutefois, une trop forte
1355 diminution de α_1 et α_2 provoque une baisse globale du rappel. En effet, requêter plusieurs instances d'un
1356 même objet est le principe à la base des méthodes d'extension de requêtes [4], ce qui améliore le rappel
1357 de l'objet considéré.

Les expérimentations ont permis de déterminer empiriquement des valeurs adaptées :

$$\alpha_1 = 0.01 \quad \alpha_2 = 0.05$$

1358 Les courbes de la figure 4.2 page suivante montrent l'effet de la mise à jour des poids. Nous
1359 constatons par exemple que pour couvrir 90% des descripteurs, il est nécessaire d'effectuer deux fois plus
1360 d'itérations lorsqu'on utilise un échantillonnage simple (non adaptatif).

1361 4.2.2 Recherche précise d'une région locale

1362 L'étape de recherche locale consiste à effectuer une recherche d'objets visuels à partir de la zone
1363 sélectionnée lors de l'étape d'échantillonnage. Comme nous l'avons vu dans la section 2.7 page 43, il
1364 existe dans l'état de l'art plusieurs méthodes de recherche d'images par le contenu d'un niveau état de

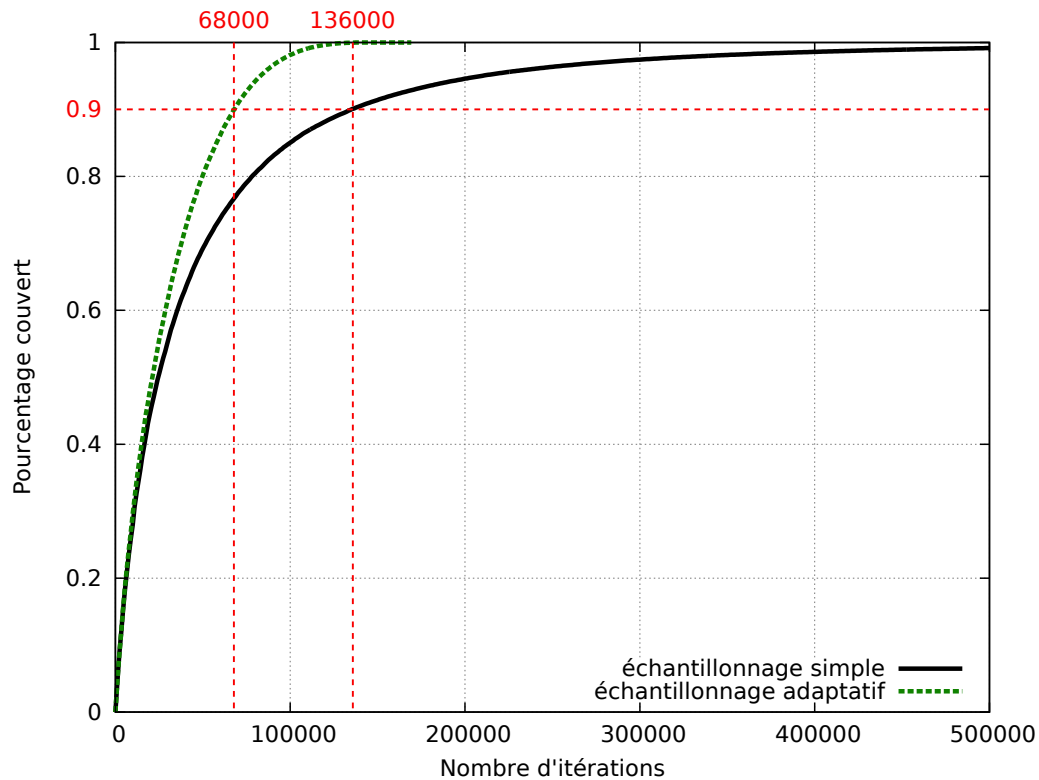


FIGURE 4.2 – Comparaison de la surface couverte (en pourcentage du nombre de descripteurs) en fonction du nombre d'itérations T de RANSAS, et du type d'échantillonnage (simple ou adaptatif).

1365 l'art pouvant convenir à cette phase. Nous discuterons les avantages et inconvénients de ces méthodes
 1366 dans la section 4.5 page 65, et nous verrons pourquoi la solution retenue est basée sur les travaux de Joly
 1367 et al. [103, 4, 90].

1368 4.2.3 Décision

1369 La dernière étape de RANSAS est chargée de décider si oui ou non un résultat de recherche de région
 1370 locale doit être conservé (dans le cadre du processus de découverte). Cette étape est très fortement
 1371 couplée à la précédente puisque elle consiste principalement à post-traiter les scores de similarité
 1372 (normalisation, seuillage).

1373 Finalement, après T requêtes candidates, l'algorithme produit un vocabulaire V de $|V| \leq T$
 1374 mots visuels géométriquement cohérents $v_{t,t}$, ayant une forte vraisemblance d'appartenir à un objet
 1375 $\{c, f\}$ -fréquent. Chaque mot visuel est associé à une région d'image R_q^t et représenté par un ensemble
 1376 de descripteurs locaux appartenant à R_q^t .

4.3 Modèle de coût de l'algorithme RANSAS

La fonction de probabilité de masse p_0 étant la condition initiale de l'algorithme d'échantillonnage pondéré adaptatif, la manière dont elle a été construite a un impact déterminant sur les performances globales du processus de découverte, c'est-à-dire sur le nombre T d'itérations requises pour découvrir les instances des objets $\{c, f\}$ -fréquents. Comme nous le verrons dans la section 7.2.2 page 99, une bonne initialisation peut permettre de diviser par 32 le nombre d'itérations nécessaires pour atteindre les performances obtenues avec une initialisation uniforme.

Il est bien sûr évident que la rapidité et l'efficacité du système effectuant la recherche précise ont une influence sur les performances de la méthode, mais il n'en demeure pas moins que la complexité globale de l'approche reste $O(T)$, quelle que soit la vitesse de l'algorithme de recherche utilisé (bien supérieure au temps de calcul de la fonction de probabilité de masse p_0). La modélisation du nombre d'itérations T nécessaires est plus difficile à modéliser car elle dépend du rappel et de la précision de l'algorithme de recherche en lui-même.

En prenant l'hypothèse d'une recherche « parfaite », nous parvenons tout de même à modéliser le comportement global du système. Considérons une base d'images avec un seul objet $\{c, f\}$ -fréquent O^m . Soit \mathbf{x}_t le descripteur candidat sélectionné lors de la phase d'échantillonnage dans \mathbf{X} , selon une fonction de probabilité de masse p_t . Tant que \mathbf{x}_t n'appartient pas à une instance O_s^m de O^m , p_t reste égal à p_0 (toujours si le système de recherche précise possède une précision parfaite).

D'un autre côté, si \mathbf{x}_t appartient à une instance O_s^m de O^m , le problème de la fouille d'objets est résolu (si le système de recherche précise possède un rappel parfait).

Ainsi, le nombre d'itérations requises T correspond au nombre de tirages successifs sans succès, c'est-à-dire ne tombant pas dans une instance O_s^m . Cette variable aléatoire suit une distribution binomiale négative $BR(1, 1 - c_0(O^m))$ où la probabilité d'échec est la probabilité qu'un descripteur échantillonné appartienne à une instance O_s^m de O^m , c'est-à-dire :

$$c_0(O^m) = \sum_{\mathbf{x}_i \in \{O_s^m\}} p_0(\mathbf{x}_i)$$

Le nombre d'itérations nécessaires estimé est donc égal à l'espérance de la loi binomiale négative $BR(1, 1 - c_0(O^m))$:

$$\hat{T} = \frac{1 - c_0(O^m)}{c_0(O^m)} = \frac{1}{c_0(O^m)} - 1 \quad (4.5)$$

et la complexité de l'algorithme est $O(\frac{1}{c_0(O^m)})$.

Nous nous référerons généralement à $c_0(O^m)$, comme la couverture statistique de l'objet O^m , étant donné qu'elle mesure le pourcentage de la fonction de probabilité de masse concerné par l'objet. Ceci montre que l'algorithme est fortement lié à la fonction de probabilité de masse p_0 .

Si on considère par exemple une fonction de probabilité de masse parfaite, c'est-à-dire :

$$p_0(\mathbf{x}_i) = 0, \forall \mathbf{x}_i \notin \{\mathbf{X}_s^m\}$$

$$p_0(\mathbf{x}_i) = \frac{1}{|\{\mathbf{X}_s^m\}|}, \forall \mathbf{x}_i \in \{\mathbf{X}_s^m\}$$

1410 avec $\{\mathbf{X}_s^m\}$ l'ensemble des descripteurs appartenant à l'objet O^m ,
 1411 alors, on a $c_0(O^m) = 1$, et la complexité de l'algorithme est $O(1)$.

1412

1413 D'un autre côté, si l'on considère une fonction de probabilité de masse uniforme, telle que :

$$p_0(\mathbf{x}_i) = 1/N, \forall \mathbf{x}_i \in \mathbf{X}$$

1414 avec N le nombre de descripteurs,

1415 alors la complexité devient $O(c_{\mathbf{X}}(O^m))$, où $c_{\mathbf{X}}(O^m)$ est la couverture globale de l'objet défini dans la
 1416 section 4.1.1 page 55.

1417 Étant donné que $c_{\mathbf{X}}(O^m) \approx c(O^m) \cdot f(O^m)$ si le nombre de points par image est globalement stable,
 1418 la complexité de l'algorithme de fouille d'objets $\{c, f\}$ -fréquents avec un échantillonnage uniforme tend
 1419 approximativement vers $O(\frac{1}{c \cdot f})$. Ceci illustre bien pourquoi la fouille de petits objets peu fréquents est
 1420 une tâche bien plus complexe que la fouille d'objets occupant une large portion des images ($c \approx 1$).

1421 La figure 4.3 présente des histogrammes de nombre d'instances en fonction du nombre de
 1422 descripteurs SIFT pour les bases BelgaLogos et Oxford Buildings. On voit nettement que la couverture
 1423 des objets de BelgaLogos est nettement inférieure à celle d'Oxford Buildings, ce qui fait des logos
 1424 de cette première des objets bien plus difficiles à fouiller que les bâtiments d'Oxford. Dans la base
 1425 Oxford Buildings, 20% des descripteurs appartiennent à des objets de la vérité terrain, contre seulement
 1426 0,5% dans la base BelgaLogos. Autrement dit, en sélectionnant au hasard un descripteur de la base
 1427 Oxford Buildings, on a une chance sur cinq de trouver un bâtiment, contre une chance sur deux-cents
 1428 pour BelgaLogos. Pour cette raison, nous nous focaliserons sur la découverte des instances issues de
 1429 BelgaLogos (cf. section 7.1 page 95), représentant un véritable défi.

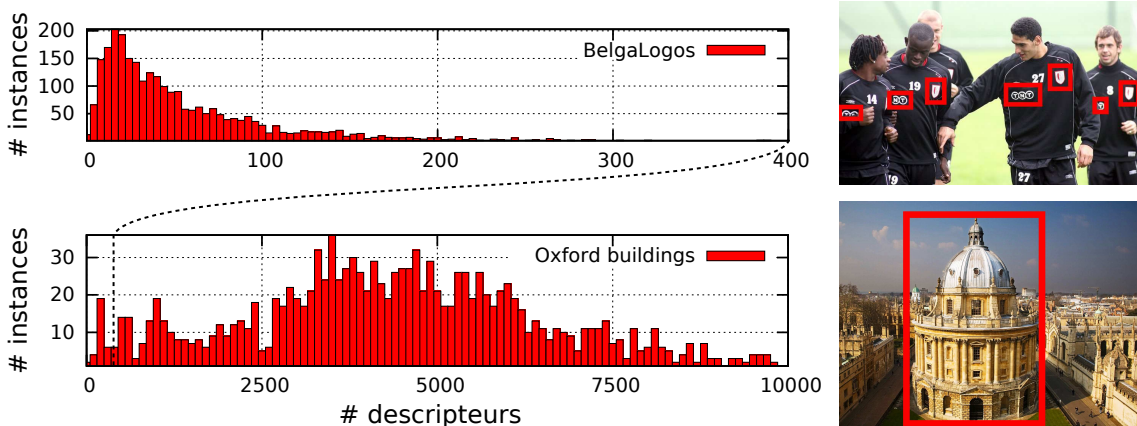


FIGURE 4.3 – Comparaison des tailles d'instances des bases BelgaLogos et Oxford Buildings

1430 Considérons maintenant un algorithme de recherche précise « réel » plutôt qu'un algorithme
 1431 « parfait », c'est-à-dire un algorithme ne retournant seulement qu'une fraction r des instances d'un

1432 objet O^m quand un descripteur requête x_t appartenant à une des instances est sélectionné par
 1433 l'échantillonnage.

1434 Alors, un rappel élevé sera obtenu seulement si un nombre suffisant μ d'instances du même
 1435 objet sont sélectionnées par l'échantillonnage. En faisant l'hypothèse que toutes les instances sont
 1436 indépendantes d'une requête à l'autre, la probabilité qu'une instance soit manquée après μ requêtes est
 1437 égale à $(1 - r)^\mu$. Si l'on souhaite que cette probabilité soit inférieure à une probabilité ϵ , alors on obtient
 1438 $\mu > \log(\epsilon) / \log(1 - r)$.

1439 Dans ce cas, le nombre d'itérations requises T suivra une distribution binomiale négative
 1440 $BR(\mu, 1 - c_0(O^m))$, au lieu de $BR(1, 1 - c_0(O^m))$. Et le nombre d'itérations nécessaires peut être
 1441 estimé par l'espérance :

$$\hat{T} = \frac{\log(\epsilon)}{\log(1 - r)} \left(\frac{1}{c_0(O^m)} - 1 \right) \quad (4.6)$$

1442 La complexité du processus global est donc toujours $O(\frac{1}{c_0(O^m)})$ pour des valeurs r et ϵ données. Pour
 1443 de relativement bonnes valeurs de rappel r et une assez faible différence de performances entre les
 1444 méthodes de recherche précise de l'état de l'art, on peut avancer que la fonction de probabilité de masse
 1445 p_0 a un bien plus fort impact sur la complexité que l'efficacité de l'algorithme de recherche précise.

1446 4.4 Conversion des poids en fonction de probabilité

1447 Après que tous les poids individuels z_0 aient été calculés, ils sont convertis en fonction de probabilité
 1448 de masse afin de pouvoir être utilisés pour la phase d'échantillonnage.

1449 La conversion est effectuée par :

$$p_0 = \frac{z_0}{z_{\max}} \cdot \frac{1}{p_z(z_0)} \quad (4.7)$$

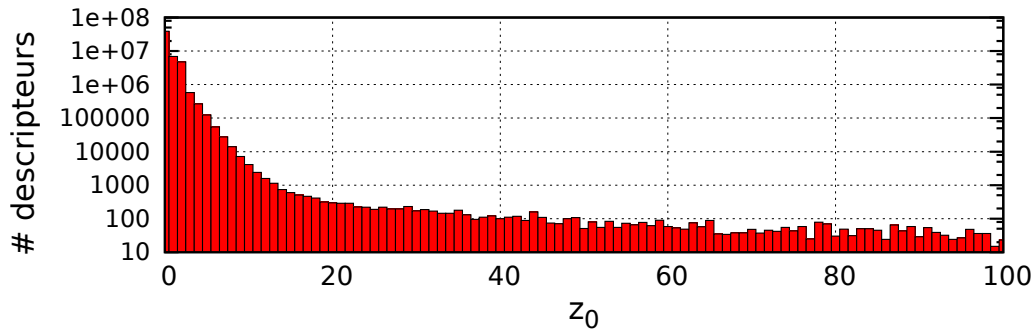
1450 où

$$z_{\max} = \max_{\mathbf{x}_i \in \mathbf{X}} z_0(\mathbf{x}_i)$$

1451 et $p_z(z)$ représente la probabilité que le poids z_0 soit égal à la valeur z dans l'ensemble des poids :

$$p_z(z) = \frac{\#\{\mathbf{x}_i \in \mathbf{X} \mid z_0(\mathbf{x}_i) = z\}}{N} \quad (4.8)$$

1452 Ceci permet de distribuer de façon égale les probabilités sur l'intervalle dans lequel varient les poids
 1453 z_0 . En effectuant une simple normalisation linéaire des poids z_0 , nous aurions conservé une masse de
 1454 probabilité très importante sur les poids les plus faibles. Ces faibles poids correspondent en effet à la très
 1455 grande majorité des descripteurs (qui n'appartiennent pas à des objets fréquents). Ce phénomène est
 1456 illustré dans la figure 4.4 page ci-contre qui montre un histogramme des poids z_0 (c'est-à-dire $N \cdot p_z(z_0)$)
 1457 sur la base FlickrBelgaLogos (présentée dans la section 7.1 page 95) utilisée dans nos expérimentations
 1458 (voir sections 5.3 page 77, 6.5 page 87 et 7.2 page 98).

FIGURE 4.4 – Histogramme des poids z_0 (en échelle logarithmique)

1459 4.5 Choix et implémentation de la méthode de recherche précise

1460 Pour la recherche de régions locales, nous avons utilisé un système de recherche d'images dont les
 1461 composants principaux sont décrits dans [103, 4, 90]. Les sections ci-après ont pour objectif de justifier le
 1462 choix des méthodes et des implémentations de chacun des composants du système de recherche précise
 1463 utilisé.

1464 4.5.1 Description des images

1465 L'objectif de cette thèse n'étant pas d'améliorer l'étape de description, nous avons utilisé une solution
 1466 de l'état de l'art. Notre choix s'est porté sur des points d'intérêt dits DoG et des descripteurs SIFT [23].
 1467 Cette solution a l'avantage d'être générique et répandue. Il est donc ainsi plus facile d'effectuer des
 1468 comparaisons, et de reproduire les contributions proposées dans cette thèse. L'utilisation du détecteur
 1469 Hessian affine [27, 28] à la place de DoG permettrait sans doute d'obtenir de meilleures performances
 1470 (au détriment du temps de description), mais nous n'avons pour l'instant pas poussé plus loin cette
 1471 investigation.

1472 4.5.2 Partitionnement des descripteurs SIFT

1473 Comme nous l'avons vu dans la section 2.7 page 43, et en particulier dans le tableau 2.1, les systèmes
 1474 états de l'art utilisent soit des méthodes de hachage comme RMMH [90], soit des méthodes de *clustering*
 1475 comme AKM [94]. L'avantage des méthodes de hachage par rapport aux méthodes de *clustering* est leur
 1476 rapidité. La complexité du hachage est linéaire par rapport au nombre de descripteurs et au nombre
 1477 de fonctions de hachage, tandis qu'elle est sur-linéaire par rapport au nombre de descripteurs pour les
 1478 méthodes de *clustering*. Le tableau 4.1 page suivante montre les temps de calcul pour partitionner deux
 1479 différents ensembles de vecteurs et met en évidence ce problème de sur-linéarité.

1480 L'idée fréquemment répandue qui justifie l'emploi de méthodes de *clustering* comme AKM, outre le
 1481 fait qu'elle propose un très bon partitionnement (une bonne minimisation des distances intra-clusters) et
 1482 un bon équilibrage des tailles de *clusters*, est que cette étape est effectuée « *offline* », une fois pour

Méthode	Paramètres	52M descripteurs		1G descripteurs	
		P. indep	P. dep	P. indep	P. dep
RMMH	128 bits	~ 647	647	~ 12,442	~ 12,442
	1,024 bits	~ 5,176	~ 5,176	~ 99,536	~ 99,536
AKM	1M mots	1,386	~ 41,580	~ 26,653	~ 799,615
	2M mots	1,680	~ 50,400	~ 32,307	~ 969,210
	4M mots	2,711	~ 81,330	~ 52,134	~ 1,564,020
	8M mots	5,297	~ 158,910	~ 101,865	~ 3,055,950

TABLE 4.1 – Comparaison des temps de calcul (en secondes) de AKM et RMMH.

La comparaison est effectuée sur une base de 52 millions de descripteurs SIFT, et sur une base d'un milliard de descripteurs. « P. dep / indep » signifie « partitionnement dépendant / indépendant des données ». On considère que le temps de calcul de RMMH est identique selon que l'on réalise son apprentissage sur les données partitionnées ou non, tandis que pour AKM, on considère le cas « dépendant » : on effectue le *clustering* sur la base elle-même (avec 30 itérations), ou « indépendant » : on possède un vocabulaire, et on effectue seulement l'étape d'affectation. Seuls les temps colorés en vert ont réellement été mesurés, les autres sont des estimations inférées à partir de la complexité des algorithmes utilisés et des temps réels en vert. On considère que les temps sont proportionnels au nombre de descripteurs, et au nombre d'itérations de AKM. Les mesures ont été effectuées sur un serveur équipé de deux processeurs hexa-cores (Intel X5660).

1483 toutes, et qu'elle peut être facilement distribuée sur de multiples serveurs de calcul. Ce temps ne devrait
 1484 donc pas être considéré comme problématique. Si l'argument peut être accepté pour un processus de
 1485 recherche d'images par le contenu (les temps de description, de partitionnement, d'indexation, etc. sont
 1486 minoritaires par rapport au temps passé à effectuer les nombreuses requêtes des nombreux utilisateurs),
 1487 il n'est plus valable dès lors que l'on se place dans le cas d'un processus de découverte d'objets visuels.
 1488 En effet, on cherche alors à obtenir un résultat le plus rapidement possible (relativement à la qualité
 1489 attendue), et ces étapes font parties intégrantes du processus.

1490 Pour toutes ces raisons, nous avons donc choisi d'utiliser RMMH pour partitionner nos descripteurs,
 1491 en tant que méthode de hachage état de l'art.

1492 4.5.3 Indexation et recherche des descripteurs SIFT

1493 Chacun des descripteurs x_{qm} composant une requête donnée Q est recherché dans l'ensemble des
 1494 descripteurs de la base grâce à la méthode de recherche approximative décrite dans APMP-LSH [103],
 1495 utilisant une stratégie *Multi-Probe* (recherches multiples dans une même structure d'index), combinée
 1496 avec un plongement dans l'espace de Hamming (*Hamming Embedding* [108, 109]).

1497 Le plongement dans cet espace a l'intérêt de réduire considérablement l'espace mémoire alloué aux
 1498 descripteurs locaux. Les expériences ont montré que des descripteurs SIFT représentés par 128 bits dans
 1499 l'espace de Hamming permettent d'obtenir des performances équivalentes à 128x32 bits dans l'espace
 1500 Euclidien. Nous obtenons donc un facteur de compression de 32.

1501 L'utilisation de la distance de Hamming permet de plus de réduire le temps de calcul en comparaison

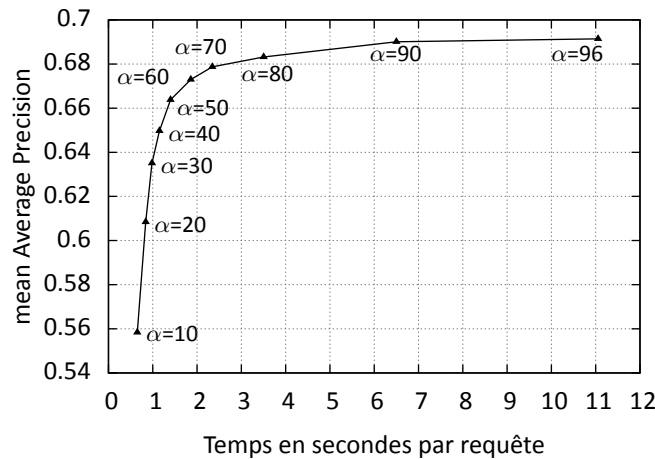


FIGURE 4.5 – Compromis entre qualité (mesurée par la mAP sur Oxford Buildings) et temps de calcul

1502 de la distance Euclidienne. Cette distance peut en effet être calculée à partir de deux opérations
 1503 extrêmement rapide : un XOR (OU exclusif) entre deux descripteurs, et le comptage du nombre de bits
 1504 égaux à 1, résultant de ce XOR. Cette deuxième opération peut être effectuée très efficacement par le
 1505 biais de l'instruction SSE4.2 « popcnt » qui divise par 4 le temps de calcul comparé à l'approche utilisant
 1506 des tables de correspondances.

1507 L'utilisation de APMP-LSH[103] a le gros avantage de permettre de régler la qualité de recherche α
 1508 (paramètre réglant le pourcentage de « vrais » plus proches voisins que l'on est statistiquement assuré de
 1509 retrouver). Ceci permet donc d'adapter le système pour utiliser un bon compromis qualité / temps de
 1510 calcul. La figure 4.5 montre la variation de la *mean Average Precision* et du temps de calcul obtenus sur la
 1511 base Oxford Buildings, en fonction du paramètre α . On constate qu'il n'est pas très rentable d'utiliser une
 1512 valeur de α supérieure à 60% ou 80%. Le temps de calcul indiqué n'a de valeur qu'à titre comparatif. De
 1513 plus, APMP-LSH garantit une évolution sous-linéaire du temps de calcul avec la croissance de la base de
 1514 descripteurs.

1515 Cette étape produit en sortie une liste de M_j descripteurs candidats x_{jm} pour chaque descripteur
 1516 requête x_{qm} .

1517 4.5.4 Vérification de la cohérence géométrique

1518 Pour chacune des images I_j contenant plus d'un nombre minimum de descripteurs candidats x_{jm}
 1519 (en général 3), nous calculons un score de cohérence géométrique $S_Q(I_j)$ en estimant le modèle d'une
 1520 transformation affine (A_j, B_j) à cinq degrés de liberté entre l'image requête et chaque image candidate
 1521 grâce à un algorithme de type RANSAC décrit dans [4].

1522 Le score de similarité d'une image I_j pour une requête Q est donné par le nombre d'*inliers* respectant

1523 le meilleur modèle de transformation (selon RANSAC) :

$$S_Q(I_j) = \sum_{m=1}^{M_j} \delta(\| \begin{bmatrix} \chi_{qm} \\ \psi_{qm} \end{bmatrix} - \mathbf{A}_j \begin{bmatrix} \chi_{jm} \\ \psi_{jm} \end{bmatrix} + \mathbf{B}_j \| \geq t) \quad (4.9)$$

1524 où $\delta(d \geq t)$ est égal à 1 si $d \geq t$, et 0 sinon. t est un seuil fixe gérant la tolérance sur l'erreur de
 1525 position ($t = 8$). M_j est le nombre de descripteurs candidats dans l'image I_j , $\begin{bmatrix} \chi_{qm} \\ \psi_{qm} \end{bmatrix}$ et $\begin{bmatrix} \chi_{jm} \\ \psi_{jm} \end{bmatrix}$ sont les
 1526 coordonnées spatiales des points d'intérêts du $m^{\text{ème}}$ appariement entre l'image requête I_q et l'image
 1527 candidate I_j .

1528 Afin de décider si la région requête R_q^t et ses résultats doivent être validés comme étant un mot visuel
 1529 géométriquement cohérent (c'est-à-dire une partie d'un potentiel objet $\{c, f\}$ -fréquent), nous filtrons les
 1530 résultats en normalisant les scores S_Q obtenus précédemment, par une technique a contrario [4].

1531 Cette technique de normalisation a contrario permet de contrôler précisément le risque de fausses
 1532 alarmes, en estimant leur distribution $\hat{N}_{fa}(S)$, avec la variable discrète $S = S_Q(I_l)$ et $I_l \in \mathcal{I}$. Selon
 1533 l'équation 4.9, S_Q dépend uniquement de l'ensemble des M_j appariements de descripteurs (c'est-à-dire
 1534 de leurs coordonnées spatiales). Les fortes valeurs de S_Q sont alors directement liées à la dépendance
 1535 statistique entre les positions spatiales des descripteurs requêtes et candidats. Aussi, nous définissons
 1536 notre modèle a contrario par la fonction de probabilité de masse $\hat{p}_{fa}(S)$ de la variable S sous l'hypothèse
 1537 \mathcal{H}_0^Q que $\begin{bmatrix} \chi_{qm} \\ \psi_{qm} \end{bmatrix}$ et $\begin{bmatrix} \chi_{jm} \\ \psi_{jm} \end{bmatrix}$ sont des variables aléatoires indépendantes pour chaque image I_j :

$$\hat{p}_{fa}(S) = \Pr[S_Q(I_l) = S \mid \mathcal{H}_0^Q]$$

1538 La fonction de distribution cumulée $\hat{N}_{fa}(S)$ est obtenue par :

$$\hat{N}_{fa}(S) = \sum_{s=0}^S \hat{p}_{fa}(s)$$

1539 Nous conservons finalement comme score normalisé $\hat{S}_Q(I)$ une estimation de la précision des
 1540 résultats selon $\hat{N}_{fa}(S)$:

$$\hat{S}_Q(I_l) = \frac{\#\{I_j \in \Omega, S_Q(I_j) > S_Q(I_l)\} - N \cdot \hat{N}_{fa}(S_Q(I_l))}{\#\{I_j \in \Omega, S_Q(I_j) > S_Q(I_l)\}}$$

1541 En pratique, nous estimons la fonction de probabilité de masse $\hat{p}_{fa}(S)$ pour chaque requête Q par
 1542 une simulation de Monte Carlo. Nous générons des coordonnées spatiales indépendantes $\begin{bmatrix} \chi_{qm} \\ \psi_{qm} \end{bmatrix}$ pour

1543 les descripteurs requêtes, et on conserve les coordonnées $\begin{bmatrix} \chi_{jm} \\ \psi_{jm} \end{bmatrix}$ des descripteurs candidats tels quels.

1544 Plus précisément, on affecte à un descripteur requête x_{qi} une nouvelle position spatiale $x_{qi'}$
 1545 sélectionnée aléatoirement parmi les autres points de la requête. Comparée à une génération aléatoire
 1546 purement uniforme des coordonnées spatiales, cette méthode a l'avantage de préserver une certaine
 1547 connaissance a priori de la distribution des points, comme les limites ou les orientations principales.

1548 Nous pouvons ensuite estimer $\hat{p}_{fa}(S)$ en comptant le nombre de résultats ayant un score S_Q
1549 supérieur à S .

1550 **4.5.5 Extension de requêtes**

1551 L'extension de requêtes est un processus très lent puisqu'il nécessite de multiplier le nombre de
1552 requêtes effectuées. Son but premier étant d'améliorer le rappel, elle n'est pas nécessaire pour le
1553 problème de la découverte des objets visuels fréquents. Elle pourrait l'être pour la fouille, afin de trouver
1554 un maximum d'instances de l'objet requêté, mais il est plus rentable de multiplier les requêtes issues de
1555 notre processus d'échantillonnage adaptatif et pondéré que d'augmenter le rappel individuel de chaque
1556 requête.

1557 Chapitre 5

1558 Calcul des scores de vraisemblance par 1559 des méthodes de l'état de l'art

1560 Comme nous l'avons vu au chapitre précédent, l'algorithme RANSAS nécessite d'être initialisé à l'aide
1561 d'une distribution a priori p_0 calculée à partir de poids ou scores de vraisemblance z_0 .

1562 Ce chapitre a pour objectif de proposer des méthodes de calcul de ces scores de vraisemblance à
1563 partir de méthodes de l'état de l'art. Nous proposerons tout d'abord des méthodes spécifiques à un type
1564 d'objet, puis des méthodes généralistes basées sur des heuristiques simples, un simple comptage des
1565 appariements, ou une solution plus sophistiquée basée sur Geometric min-Hashing.

1566 5.1 Calcul de scores de vraisemblance spécifiques à un type d'ob- 1567 jet

1568 Dans cette section sont présentées les scores de vraisemblance spécifiques à certains concepts
1569 visuels. Deux exemples illustratifs des possibilités offertes par ces méthodes sont proposés. Le premier
1570 génère des scores différents selon la position dans l'image, et le deuxième se focalise sur les visages
1571 détectés. On pourrait bien entendu proposer de nombreuses autres alternatives, dans la catégorie
1572 « détecteur » (détecteur de texte, détecteur de mouvement dans le cas des vidéos ...), ou dans la
1573 catégorie « a priori » (couleur, texture, ...).

1574 5.1.1 Scores de vraisemblance basés sur la position dans les images

1575 Le cadrage des photographies ou vidéos obéit généralement à certaines règles esthétiques : les objets
1576 d'intérêt sont placés au centre du cadre, sur des lignes placées sur les tiers de l'image, etc.

1577 L'idée de cette distribution spécifique à la position est donc d'utiliser ces règles afin que pour chaque
1578 descripteur, sa probabilité d'être échantillonné par l'algorithme RANSAS dépende directement de sa
1579 distance au centre de l'image ou au tiers le plus proche.

Si l'on prend l'exemple d'une distribution calculée par rapport au centre de l'image, on a :

$$z_0^P(\mathbf{x}_i) = \mathcal{K}_\sigma(\mathbf{C}_{I_j}, \mathbf{p}_{ij})$$

1580 où \mathbf{C}_{I_j} est le centre de l'image I_j , $\mathbf{p}_{ij} = (I_j, x_{ij}, \psi_{ij})$ est la position du point x_i et \mathcal{K} le noyau RBF
 1581 paramétré dans le cadre de nos expérimentations avec $\sigma_j^x = \frac{W_j}{8}$, $\sigma_j^\psi = \frac{H_j}{8}$.

La figure 5.1 présente un exemple d'un objet centré dans l'image.



FIGURE 5.1 – Score de vraisemblance basé sur la position centrale dans les images. Plus la couleur est chaude, et plus on est proche du centre de l'image

1582

1583 La figure 5.2 page ci-contre montre qu'en calculant un score de vraisemblance basé sur la position
 1584 centrale dans l'image, on améliore significativement la mAP par rapport à une distribution uniforme.
 1585 Cette expérimentation (initialement publiée dans [162]) a été réalisée sur la base Oxford Buildings. Cette
 1586 base se prête particulièrement à l'évaluation de cette méthode puisque les bâtiments sont généralement
 1587 centrés dans les photographies. Le nombre de mots $|V|$ est le nombre de mots visuels géométriquement
 1588 cohérents extraits par l'algorithme RANSAS. La mAP est calculée en effectuant les requêtes associées à la
 1589 vérité terrain d'Oxford Buildings, en utilisant une représentation des images par sacs de mots visuels,
 1590 dont les mots visuels sont les mots visuels géométriquement cohérents de RANSAS. Le calcul de distances
 1591 entre images se fait ensuite par un simple produit scalaire.

1592 5.1.2 Scores de vraisemblance basés sur la détection de visages

1593 Dans cette méthode, un détecteur de visages (celui d'OpenCV¹ dans le cadre de nos expérimentations)
 1594 est passé sur les images et nous affectons une plus forte probabilité aux zones détectées comme des
 1595 visages. Bien évidemment, n'importe quel type de détecteurs aurait pu être utilisé.

1596 La position et l'échelle des visages permettent de calculer la probabilité :

$$z_0^F(\mathbf{x}_i) = \sum_{l=0}^{N^f-1} \mathcal{K}_{\sigma_l^f}(\mathbf{p}_{ij}, \mathbf{p}_l^f)$$

1597 où \mathcal{K} est le noyau RBF, N^f est le nombre de visages détectés dans l'image I_j , \mathbf{p}_l^f et σ_l^f sont
 1598 respectivement le centre et l'échelle du $l^{\text{ème}}$ visage dans l'image courante I_j .

1599 La figure 5.3 page 74 montre un résultat de ce type de distribution dans une image.

1. <http://opencv.org>

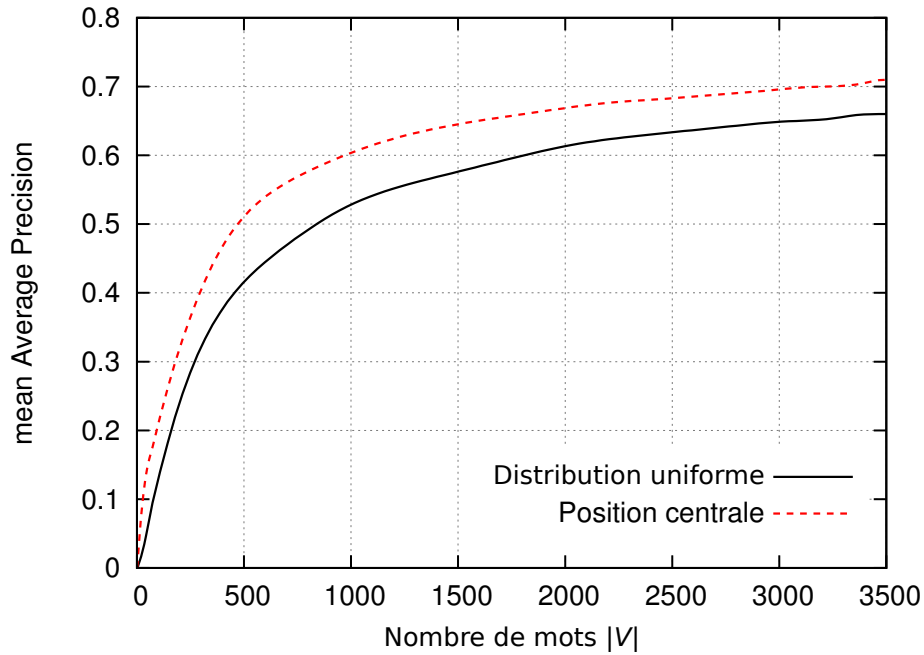


FIGURE 5.2 – Apport du score de vraisemblance « position centrale » par rapport à l'uniforme

1600 Pour observer l'effet du score de vraisemblance basé sur la détection de visages, nous avons à trois
 1601 reprises généré 600 mots visuels géométriquement cohérents sur la base BelgaLogos. Seuls les mots
 1602 composés d'un minimum de cinq instances d'objets ont été pris en compte. Nous avons ensuite compté le
 1603 nombre moyen de détections dans chaque catégorie (divers, logos, texture ou visage). Le tableau 5.1, issu
 1604 de notre première publication [162], montre qu'en utilisant une distribution uniforme, on détecte peu de
 1605 visages, et principalement des textures, tandis qu'avec une distribution basée sur la détection de visages,
 1606 on augmente nettement le nombre de visages obtenus.

	Distribution uniforme	Distribution « visage »
Divers	31	20
Logos	46	27
Textures	72	23
Visages	12	47

TABLE 5.1 – Nombre de détections par catégorie

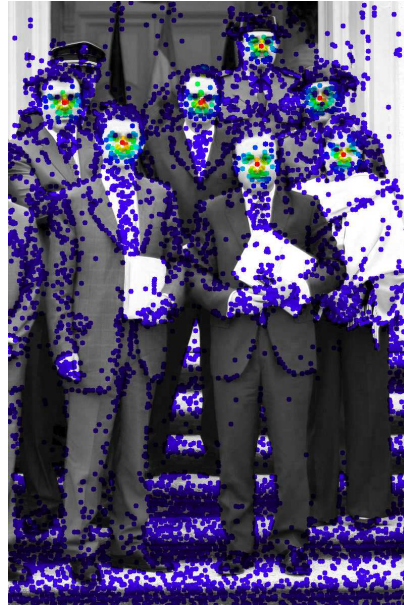


FIGURE 5.3 – Score de vraisemblance basé sur la détection de visages. Plus la couleur est chaude et plus on est proche d'un visage détecté.

1607 5.2 Calcul de scores de vraisemblance généralistes

1608 5.2.1 Calcul de scores de vraisemblance par des heuristiques simples

1609 Dans cette section sont présentés deux exemples de scores de vraisemblance généralistes. Le premier
 1610 est basé sur la densité spatiale des points d'intérêts, et le deuxième, sur la variabilité dans l'espace des
 1611 descripteurs.

1612 5.2.1.1 Scores de vraisemblance basés sur la densité spatiale des points d'intérêts

1613 Le principe de cette mesure de saillance est de mettre l'accent sur les centres des régions denses (en
 1614 termes de points d'intérêts), puisqu'on peut considérer qu'elles sont plus sujettes à contenir des objets
 1615 d'intérêts car plus informatives. Concrètement, nous mesurons pour chaque point d'intérêt x_i la densité
 1616 spatiale grâce ses points voisins, à l'aide d'une fenêtre de Parzen :

$$z_0^D(x_i) = \sum_{l=0}^{|\mathbf{X}_q|-1} \mathcal{K}_\sigma(\mathbf{p}_{ij}, \mathbf{p}_l)$$

1617 où \mathcal{K}_σ est typiquement un noyau RBF, et \mathbf{X}_q l'ensemble des points d'intérêts appartenant à l'image
 1618 I_j . Les expériences présentées dans la section 5.3.3 page 80 utilisent des valeurs différentes pour σ selon
 1619 qu'il s'agit des coordonnées horizontales ou verticales ($\sigma_x = \frac{W_i}{3}$ et $\sigma_y = \frac{H_i}{3}$). Un exemple du résultat de
 1620 cette méthode est donné dans la figure 5.4 page ci-contre, et des expérimentations quantitatives sont
 1621 menées dans la section 5.3.3 page 80.

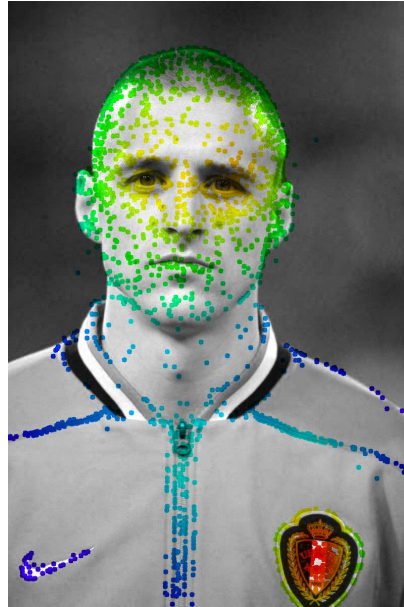


FIGURE 5.4 – Densité spatiale des points d'intérêts. Plus la couleur est chaude et plus la densité est forte.

1622 5.2.1.2 Scores de vraisemblance basés sur la variabilité dans l'espace des descripteurs

1623 L'un des problèmes fréquemment rencontrés dans les techniques d'appariement est la dégradation
 1624 des performances en présence de structures (motifs) répétées (comme certaines textures). Ce problème
 1625 vient de la forte probabilité d'obtenir des appariements multiples pour chacun des descripteurs [131,
 1626 166, 139]. Pour diminuer la probabilité de sélectionner de telles régions lors de l'étape d'échantillonnage,
 1627 nous proposons donc une mesure calculant la variabilité des descripteurs, dans l'espace des descripteurs.
 1628 Les descripteurs considérés pour la mesure de saillance de x_i appartiennent à la région candidate R_i ,
 1629 rectangle de dimension proportionnelle aux dimensions de l'image et centré autour de x_i . La variabilité
 1630 est calculée comme étant la moyenne des variances sur chaque dimension de l'espace des descripteurs,
 1631 dans la région spatiale considérée :

$$z_0^\delta(x_i) = \frac{1}{d} \sum_{l=0}^{d-1} \text{var}_{R_i}(x[l])$$

1632 où d est la dimension des descripteurs.

1633 Un exemple de cette mesure de saillance est illustré par la figure 5.5 page suivante et des
 1634 expérimentations quantitatives sont menées dans la section 5.3.3 page 80.

1635 5.2.2 Calcul du score de vraisemblance par comptage des appariements

1636 Plutôt que d'utiliser des mesures de saillance calculées au niveau image, comme vu précédemment, il
 1637 peut être beaucoup plus efficace d'employer des méthodes tenant compte de l'ensemble de la base
 1638 d'images. Le coût de calcul est bien entendu plus important qu'en travaillant à l'échelle d'une seule image,

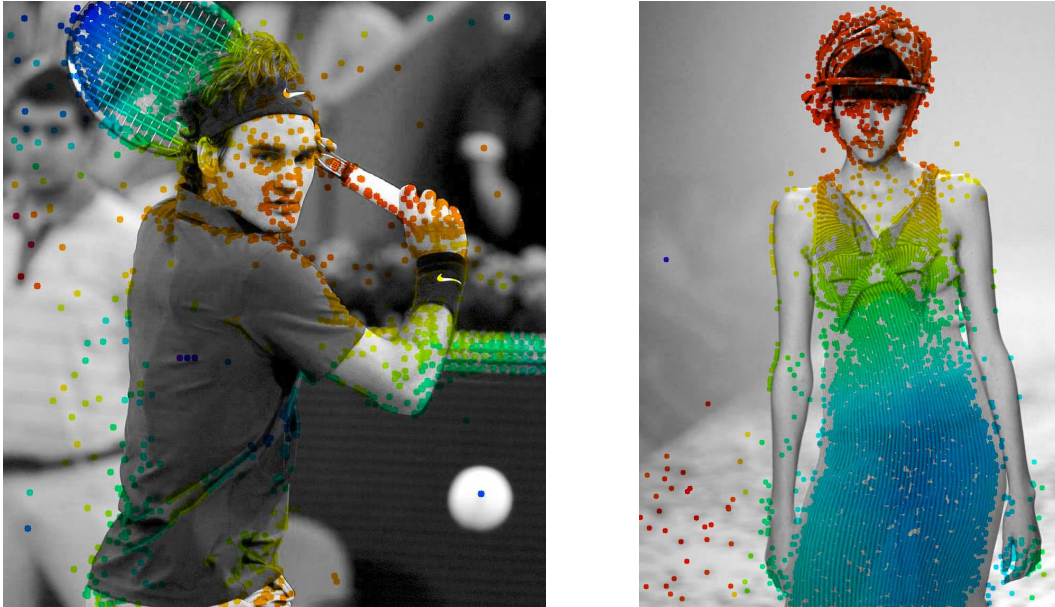


FIGURE 5.5 – Variabilité dans l'espace des descripteurs. Plus la couleur est chaude et plus la variabilité est importante.

1639 mais ce coût est compensé par la phase de découverte (recherche précise) qui est fortement accélérée
1640 (on a besoin de moins d'itérations).

1641 Le calcul des scores individuels $z_0(\mathbf{x}_i)$ qui après normalisation deviennent la fonction de probabilité
1642 de masse p_0 , peut donc être effectué à partir d'une méthode d'appariement de descripteurs. La méthode
1643 la plus couramment rencontrée dans la littérature utilise le principe des mots visuels (chaque mot visuel
1644 est un *cluster* de descripteurs locaux). Dans cette méthode, les descripteurs appartenant à un mot visuel
1645 sont appariés avec tous les autres descripteurs dans ce même mot visuel.

1646 Soit w le mot visuel à qui on a affecté le descripteur \mathbf{x}_i , et $|w|$ le nombre total de descripteurs
1647 affectés à w . On calcule le score individuel du descripteur \mathbf{x}_i par :

$$z_0^w(\mathbf{x}_i) = |w|$$

1648 La section 5.3.3 page 80 présente une évaluation quantitative de cette approche.

1649 5.2.3 Calcul du score de vraisemblance par Geometric min-Hashing

1650 L'algorithme Geometric min-Hashing [160] (pour la découverte de petits objets) est composé de
1651 différentes étapes :

- 1652 1. Création d'un vocabulaire (*clustering K-Means*) à partir des descripteurs ;
- 1653 2. Filtrage des descripteurs : on conserve les descripteurs dont les mots visuels sont uniques dans leur
1654 image, et ayant des voisins satisfaisant certaines contraintes géométriques ;
- 1655 3. Génération de multiples *sketches* (s-tuples de mots visuels) ;

1656 4. Co-segmentation des régions contenant des *sketches* ayant subi des collisions ;

1657 Afin de comparer la stratégie de GmH avec celle proposée précédemment, et notre propre méthode
1658 introduite au chapitre 6 page 81, nous verrons comment adapter Geometric min-Hashing pour générer
1659 des scores individuels de probabilité d'appartenance à un objet fréquent pour chaque descripteur. Nous
1660 reprendrons donc les trois premières étapes de l'algorithme original, puis nous proposerons une méthode
1661 pour générer les scores $z_0(\mathbf{x}_q)$.

1662 Dans les expérimentations présentées dans la section 5.3, le vocabulaire visuel a été créé par un
1663 *clustering K-Means*. Afin d'accélérer cette étape, l'algorithme AKM de Philbin et. al [94] a été employé.
1664 L'étape de filtrage des descripteurs sur leur unicité dans l'image et sur leurs voisins a été conservée telle
1665 qu'elle. Enfin la troisième et dernière des étapes conservées, la génération des *sketches* a simplement été
1666 réordonnée. Dans l'algorithme original, les auteurs généraient un *sketch* par image, et recommençaient
1667 cette étape autant de fois que nécessaire. Dans l'adaptation proposée, nous générons un nombre de
1668 *sketches* donné par image, et nous nous arrêtons lorsque toutes les images ont été parcourues. Ceci
1669 permet le calcul du score de vraisemblance, sans toutefois modifier les performances de GmH (à nombre
1670 de *sketches* équivalent).

1671 Les scores individuels $z_0(\mathbf{x}_i)$ sont générés comme étant le nombre de *sketches* collisionnant avec le
1672 ou les *sketches* construits à partir de \mathbf{x}_i .

1673 5.3 Expérimentations

1674 5.3.1 Protocole d'évaluation

1675 Dans le but d'évaluer la qualité des scores $z_0(\mathbf{x}_i)$, c'est-à-dire leur adéquation avec la vérité terrain,
1676 on définit deux mesures : la précision, et le rappel. On rappelle que les scores parfaits sont ceux qui après
1677 normalisation, donnent la fonction de probabilité de masse suivante :

$$1678 \begin{aligned} p_0(\mathbf{x}_i) &= 0, \forall \mathbf{x}_i \notin \{\mathbf{X}_s^m\} \\ p_0(\mathbf{x}_i) &= \frac{1}{|\{\mathbf{X}_s^m\}|}, \forall \mathbf{x}_i \in \{\mathbf{X}_s^m\} \end{aligned}$$

1679 avec $\{\mathbf{X}_s^m\}$ l'ensemble des descripteurs appartenant à des instances de la vérité terrain.

1680 Le rappel, pour un seuil η_0 donné, est le pourcentage d'instances dans la vérité terrain ayant au moins
1681 un descripteur local \mathbf{x} avec une probabilité telle que $p_0(\mathbf{x}) > \eta_0$.

1682 La précision est le pourcentage de descripteurs ayant une probabilité supérieure à η_0 , qui
1683 appartiennent à une instance de la vérité terrain.

1684 Les courbes de précision/rappel sont ensuite construites en faisant varier η_0 .

1685 Ces deux mesures d'évaluation sont totalement indépendantes de l'étape de recherche précise, et
1686 permettent ainsi de comparer directement la qualité des différentes méthodes de calcul des scores de
1687 vraisemblance. Contrairement aux mesures de précision/rappel habituellement utilisées, les mesures
1688 proposées ici opèrent donc à des niveaux différents : le rappel opère au niveau instance, et la précision au
1689 niveau descripteur.

1690 Les expérimentations suivantes ont été effectuées sur la base FlickrBelgaLogos², dont le détail est
1691 donné dans la section 7.1 page 95. Les raisons pour lesquelles cette évaluation utilise cette base sont
1692 également expliquées dans la même section 7.1 page 95.

1693 5.3.2 Évaluation de Geometric min-Hashing comme score de vraisemblance

1694 Cette section a pour objectif d'observer l'influence des deux principaux paramètres de l'algorithme
1695 Geometric min-Hashing : le nombre de *sketches* et la taille du vocabulaire.

1696 Il est à noter que les courbes de précision/rappel s'arrêtent prématurément. En effet, de très
1697 nombreux descripteurs étant ignorés lors du processus de génération des *sketches*, la plupart des
1698 descripteurs se retrouvent avec un score nul. Ceci est observé même avec un très grand nombre de
1699 *sketches* (100K), comparé au nombre utilisé dans le papier d'origine. Il serait donc trompeur de prolonger
1700 les courbes jusqu'au point de rappel parfait et de précision égale à celle obtenue lorsque le seuil η_0
1701 est inférieur à tous les scores (cette précision est égale au nombre de descripteurs contenus dans des
1702 instances de la vérité terrain, divisé par le nombre total de descripteurs de la base, soit environ 0,00355).

1703 5.3.2.1 Influence du nombre de *sketches*

1704 La figure 5.6 page suivante montre l'influence du nombre k de *sketches* sur la précision et le rappel.
1705 L'article original [160] prétend que 60 *sketches* sont suffisants pour effectuer un *clustering* de qualité sur
1706 la base Oxford Buildings. Cependant, sur la base BelgaLogos, on voit qu'il faut au minimum 100 000
1707 *sketches* pour atteindre des taux de précision/rappel de l'ordre de 20%. Le mode de génération des
1708 *sketches* (par min-Hashing) montre donc clairement ses limites puisqu'avec un tel nombre de *sketches*, on
1709 peut considérer que l'on a généré une bonne partie des *sketches* constructibles. Un vocabulaire de 1
1710 million de mots a été utilisé.

1711 5.3.2.2 Influence de la taille du vocabulaire

1712 La figure 5.7 page ci-contre montre l'effet de la taille du vocabulaire utilisé. On voit nettement que
1713 plus la taille augmente et plus on gagne en précision/rappel, en convergeant très doucement. On peut
1714 supposer qu'en continuant d'augmenter la taille du vocabulaire, on améliorerait également la précision,
1715 mais que le rappel finirait inexorablement par chuter au fur et à mesure que la taille des *clusters* tendrait
1716 vers 1. En pratique, l'utilisation d'un vocabulaire aussi grand est problématique. Le temps de calcul
1717 nécessaire pour effectuer le *clustering* avec 8 millions de mots, ou plus, devient réellement prohibitif (voir
1718 tableau 4.1 page 66). De plus, le vocabulaire n'est pas réutilisable pour la phase de recherche précise. On
1719 peut également remarquer que nous ne traitons ici que 50 millions de descripteurs, quand on aimerait
1720 potentiellement pouvoir en traiter plus d'un milliard. En pratique, un vocabulaire d'un million de mots est
1721 un bon compromis, à la fois pour le temps de calcul, et aussi pour la ré-utilisabilité lors de la phase de
1722 recherche précise.

2. <http://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/FlickrBelgaLogos.html>

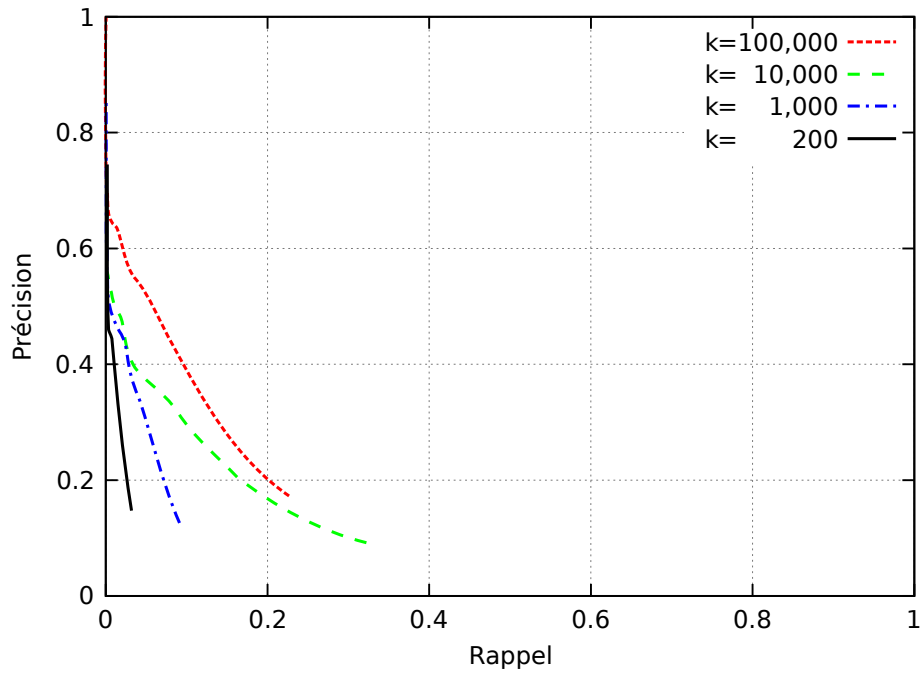


FIGURE 5.6 – Influence du nombre de *sketches* k sur la précision et le rappel. Un vocabulaire de 1 million de mots a été utilisé.

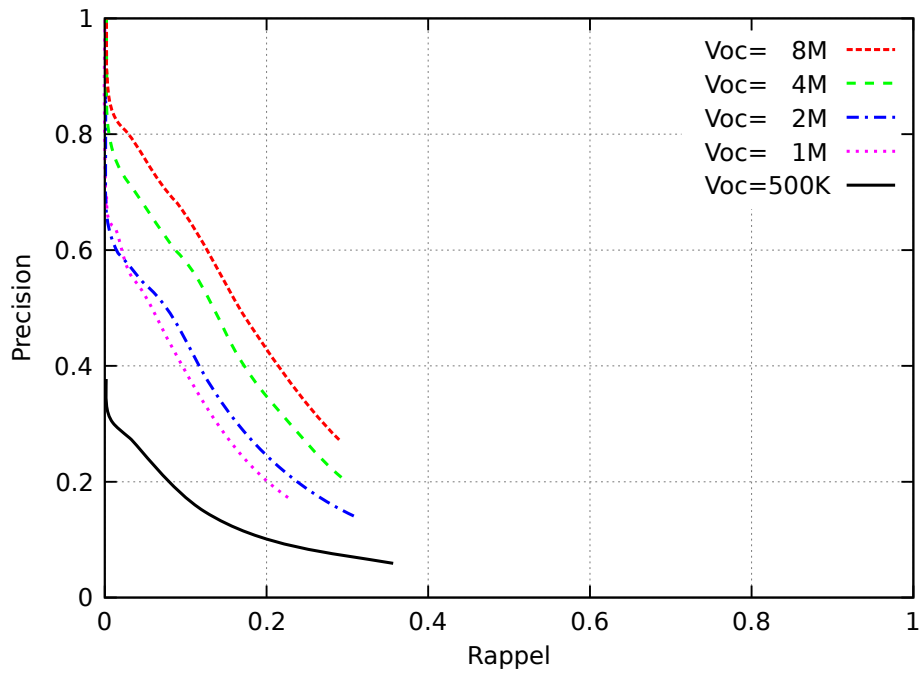


FIGURE 5.7 – Influence de la taille du vocabulaire (Voc) sur la précision et le rappel. 100 000 *sketches* ont été générés.

1723 5.3.3 Comparaison des performances des scores de vraisemblance généralistes

1724 Les courbes de la figure 5.8 montrent les performances des méthodes de calcul de scores de
 1725 vraisemblance par des heuristiques simples (voir section 5.2.1 page 74), par comptage des appariements
 1726 (voir section 5.2.2 page 75), ou par Geometric min-Hashing (voir section 5.2.3 page 76).

1727 On constate que l'heuristique simple se focalisant sur les zones denses (en termes de points d'intérêt)
 1728 obtient des performances similaires à celles de l'aléatoire, c'est-à-dire que peu importe le rappel, on
 1729 obtient une précision d'environ 0,00355 correspondant au nombre de descripteurs contenus dans des
 1730 instances de la vérité terrain, divisé par le nombre total de descripteurs de la base. Cette heuristique est
 1731 donc totalement inutile pour cette base d'images.

1732 L'heuristique se basant sur la variabilité des descripteurs obtient légèrement mieux, mais cette
 1733 amélioration n'est pas suffisante pour justifier son calcul. La méthode de calcul de score par comptage
 1734 des appariements améliore encore un peu les performances, en obtenant un compromis précision/rappel
 1735 égal à 10%/10%. Cette méthode souffre clairement d'un problème de robustesse (le simple appariement
 1736 des descripteurs ne permet pas d'obtenir des précisions élevées).

1737 Geometric min-Hashing résout partiellement ce problème en rendant les appariements plus précis
 1738 par l'utilisation de *sketches* (avec contraintes géométriques lors de la construction) au lieu d'appariements
 1739 individuels.

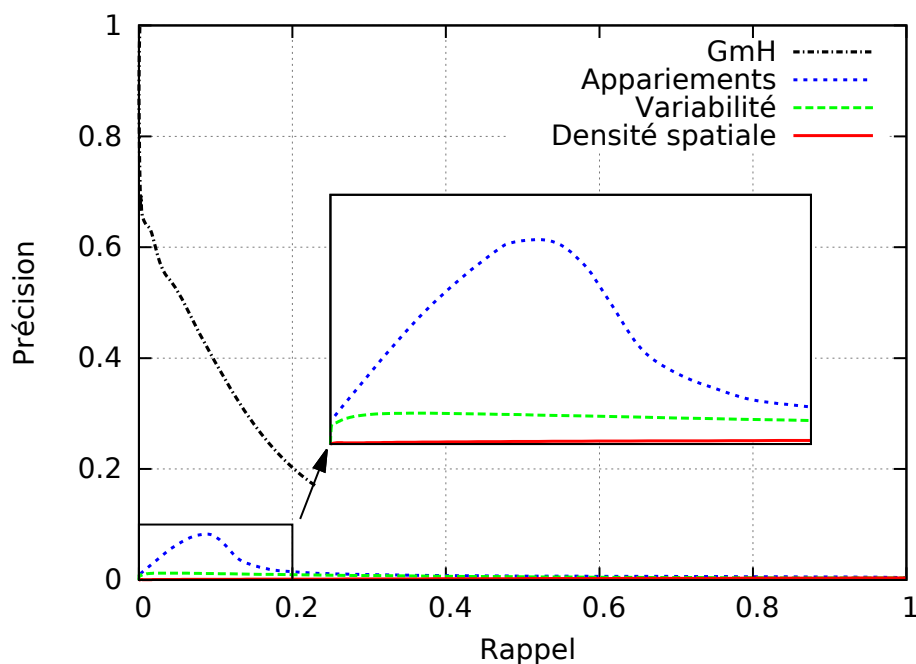


FIGURE 5.8 – Comparaison des performances des scores de vraisemblance générés par Geometric min-Hashing, par appariements de descripteurs (mots visuels), par variabilité des descripteurs, ou encore par densité spatiale des descripteurs. La courbe GmH a été obtenue avec un vocabulaire d'un million de mots visuels et 100 000 *sketches*.

1740 Chapitre 6

1741 Calcul des scores de vraisemblance par 1742 hachage visuel et ajout de contraintes 1743 géométriques faibles

1744 Comme nous l'avons vu précédemment, notre objectif est de calculer une fonction de probabilité de
1745 masse p_0 sur l'ensemble des descripteurs de la base \mathbf{X} , de manière à ce que $p_0(\mathbf{x}_i)$ reflète au mieux la
1746 vraisemblance qu'un descripteur local \mathbf{x}_i appartienne à un objet $\{c, f\}$ -fréquent. En d'autres mots, nous
1747 cherchons à maximiser $c_0(O^m)$ pour tous les objets $\{c, f\}$ -fréquents de la base (cf. section 4.3 page 62).

1748 Plutôt que d'utiliser des mesures de saillance calculées au niveau image, nous verrons qu'il peut être
1749 beaucoup plus efficace d'employer des méthodes tenant compte de l'ensemble de la base d'images. Le
1750 coût de calcul est bien entendu plus important qu'en travaillant à l'échelle d'une seule image, mais ce
1751 coût est compensé par la phase de recherche précise dont le nombre d'itérations T est fortement réduit.

1752 Le calcul des scores individuels $z_0(\mathbf{x}_i)$ repose principalement sur l'observation de la fréquence des
1753 collisions dans des tables de hachage, ce qui permet à l'algorithme présenté dans cette section de rester
1754 scalable et facilement distribuable si besoin.

1755 Cet algorithme est composé des trois étapes suivantes :

- 1756 1. Hachage visuel et filtrage des appariements candidats ;
- 1757 2. Extraction et hachage des informations de géométrie faible ;
- 1758 3. Calcul des scores individuels $z_0(\mathbf{x}_i)$ par comptage dans les tables de hachage de la géométrie
1759 faible ;

1760 Intuitivement, le score $z_0(\mathbf{x}_i)$ calculé pour chaque \mathbf{x}_i représente le nombre de correspondances
1761 visuelles qui sont à la fois dans le voisinage de p_{ij} et qui sont géométriquement cohérentes avec les
1762 appariements visuels directs de \mathbf{x}_i .

1763 6.1 Hachage visuel et filtrage des appariements candidats

1764 6.1.1 Construction de l'index visuel

1765 Comme nous l'avons vu au chapitre 2 page 21, il y a deux stratégies principales pour la recherche par
 1766 similarité de descripteurs locaux : un vocabulaire visuel associé à une liste inversée [82, 137], ou les
 1767 méthodes de hachage [4, 90]. Ces deux types de méthodes poursuivent le même but : réduire le coût de
 1768 calcul de la recherche, en partitionnant et en compressant les descripteurs.

1769 La plupart des méthodes de recherche par similarité visuelle sont basées sur l'utilisation d'un
 1770 vocabulaire visuel. Ce vocabulaire est généralement généré à l'aide d'un *clustering K-Means* [104, 134, 82,
 1771 137]. Comme nous l'avons vu dans la section 4.5 page 65, le principal inconvénient de ces méthodes
 1772 est que la génération du vocabulaire est extrêmement coûteuse, et en particulier pour un processus
 1773 de découverte. En effet, dans le cadre de la génération d'un index pour un moteur de recherche par
 1774 similarité visuelle, cette étape n'est d'ordinaire pas considérée comme coûteuse car elle est vue comme
 1775 une étape préliminaire effectuée une seule fois pour toute une base (*offline*). Le seul coût considéré est
 1776 alors celui des multiples recherches *online*. Au contraire, dans un processus de découverte, le but est de
 1777 faire le minimum possible de recherches dans la base. La génération de l'index devient alors un des coûts
 1778 principaux.

1779 Afin de diminuer ces coûts de calcul, nous avons donc choisi d'utiliser des méthodes de hachage pour
 1780 compresser et indexer les descripteurs visuels. Les méthodes de hachage ont par le passé prouvé qu'elles
 1781 pouvaient obtenir d'aussi bon résultats dans le domaine de la recherche d'objets visuels [4, 162].

1782 De plus, comme précisé dans la section 4.5 page 65, le temps de calcul nécessaire au hachage (128
 1783 bits) d'une base de $N = 10^9$ SIFT est inférieur à 4 heures avec un serveur à 12 cœurs, tandis qu'il en
 1784 faut le double pour effectuer une itération de *K-Means* (avec AKM). Cette indexation visuelle est donc
 1785 particulièrement efficace et adaptée à un processus à large échelle.

1786 Plutôt que d'utiliser des fonctions LSH classiques, la méthode employée ici est basée sur un algorithme
 1787 plus efficace : *Random Maximum Margin Hashing* (RMMH, [90]), une famille de fonctions de hachage
 1788 récemment introduite. La principale originalité de RMMH est d'apprendre des séparations aléatoires
 1789 des données, quelle que soit la proximité des échantillons d'apprentissage (sans aucune supervision).
 1790 Concrètement, la méthode fonctionne en apprenant un ensemble de classifieurs à partir d'une petite
 1791 partie de la base. Pour chaque fonction de hachage, M points d'apprentissage sont sélectionnés
 1792 aléatoirement dans \mathbf{X} et sont ensuite étiquetés aléatoirement (la moitié avec -1 et le reste avec 1). Si on
 1793 note x_j^+ les $\frac{M}{2}$ échantillons d'apprentissage positifs, et x_j^- les négatifs, chaque fonction de hachage est
 1794 ensuite calculée par un classifieur binaire $h(\mathbf{x})$ tel que :

$$h(\mathbf{x}) = \operatorname{argmax}_{h_\theta} \sum_{j=1}^{\frac{M}{2}} h_\theta(\mathbf{x}_j^+) - h_\theta(\mathbf{x}_j^-) \quad (6.1)$$

1795 En utilisant un SVM linéaire comme classifieur binaire, on obtient :

$$h(\mathbf{x}) = \operatorname{sgn} \left(\sum_{i=1}^m \alpha_i^* \mathbf{x}_i^* \cdot \mathbf{x} + b_m \right) \quad (6.2)$$

1796 où \mathbf{x}_i^* sont les m vecteurs supports sélectionnés par le SVM ($\mathbf{x}_i^* \in \{\mathbf{x}_j^+, \mathbf{x}_j^-\}$) qui estime α_i^* et b_m .

1797 Dans [90], les auteurs montrent que le paramètre M est en pratique assez stable sur différents types
1798 de descripteurs et différentes tailles de base. Par la suite, on notera $\mathbf{h}_l(\mathbf{x})$ la clé de hachage de longueur k
1799 produit par la concaténation de k fonctions de hachage binaire individuelles pour la $l^{\text{ème}}$ des L fonctions
1800 de hachage.

1801

1802 Une fois que l'index visuel a été créé, l'algorithme calcule individuellement les scores $z_0(\mathbf{x}_i)$ en
1803 traitant indépendamment toutes les images $I_j \in \mathcal{I}$ une par une (soit itérativement, soit en parallèle).
1804 Notons I_Q l'image traitée à la $Q^{\text{ème}}$ itération et \mathbf{X}_Q l'ensemble des descripteurs locaux lui appartenant.

1805 6.1.2 Filtrage visuel des appariements candidats

1806 L'utilisation « naïve » de l'index visuel génère un très grand nombre de collisions entre les descripteurs,
1807 qui sont autant d'appariements candidats. Afin de limiter ce nombre, et donc de limiter le temps de
1808 calcul, nous appliquons trois stratégies de filtrage pour tous les descripteurs $\mathbf{x}_q \in \mathbf{X}_Q$:

- 1809 – Filtrage sur la fréquence de collision intra-image ;
- 1810 – Filtrage sur la fréquence de collision inter-tables ;
- 1811 – Filtrage par les K plus proches voisins ;

1812 6.1.2.1 Filtrage sur la fréquence de collision intra-image

1813 Comme notifié dans [160], les descripteurs visuels qui sont uniques dans une image, sont bien plus
1814 propices à former des appariements que l'on peut considérer comme fiables. Cette observation permet
1815 de réduire une part importante des faux positifs produits par les textures et autres motifs répétés. Plutôt
1816 que de conserver seulement les descripteurs uniques, notre algorithme propose une approche plus
1817 souple qui limite la fréquence de collision intra-image à une valeur maximum ζ . De cette manière, on
1818 obtient un sous-ensemble \mathbf{X}'_Q de descripteurs candidats à l'étape suivante, à partir de \mathbf{X}_Q :

$$\mathbf{X}'_Q = \{\mathbf{x}_q \in \mathbf{X}_Q \mid f_Q(\mathbf{x}_q) < \zeta\} \quad (6.3)$$

1819 où la fréquence de collision intra-image f_Q est estimée par sa valeur moyenne à travers les L tables
1820 de hachage :

$$f_Q(\mathbf{x}_q) = \frac{1}{L} \sum_{l=1}^L \#\{\mathbf{x} \in \mathbf{X}_Q \mid \mathbf{h}_l(\mathbf{x}) = \mathbf{h}_l(\mathbf{x}_q)\} \quad (6.4)$$

1821 Empiriquement, ζ a été fixé à une valeur par défaut à $\zeta = 3$.

1822 6.1.2.2 Filtrage sur la fréquence de collision inter-tables

1823 Comme suggéré par la méthode Frequency-based LSH [100], l'algorithme filtre ensuite l'ensemble des
1824 descripteurs de la base ayant des collisions avec \mathbf{X}'_Q . Ce filtrage s'effectue sur la fréquence de collision

1825 à travers les L tables de hachage de l'index visuel. L'ensemble des appariements Y_Q résultant de ce
 1826 filtrage peut-être exprimé de la façon suivante :

$$Y_Q = \{(\mathbf{x}_q, \mathbf{x}_i) \in (\mathbf{X}'_Q, \mathbf{X}) \mid f_L(\mathbf{x}_q, \mathbf{x}_i) > \tau\} \quad (6.5)$$

1827 où $f_L(\mathbf{x}_q, \mathbf{x}_i)$ est le nombre de fois où \mathbf{x}_q et \mathbf{x}_i ont la même clé de hachage à travers les L tables de
 1828 hachage visuel.

1829 Comme suggéré dans [100], le seuil τ peut être calculé numériquement comme une fonction de L et
 1830 ϕ , où ϕ est une borne inférieure sur la fréquence de collision dans une seule fonction de hachage. Ce
 1831 seuillage basé sur la fréquence approxime une requête par rayon dans l'espace original des descripteurs
 1832 lorsque LSH est utilisé. Avec RMMH, il s'agit plus d'une approximation d'un seuil basé sur la densité.

1833 Nous verrons dans la section 6.5.1.1 page 88 comment nous paramétrons ce filtrage, grâce à
 1834 l'équation 6.6 donnée par Frequency-based LSH [100] :

$$\phi^* = \sum_{l=\tau}^L C_L^l \phi^l (1 - \phi)^{L-l} \quad (6.6)$$

1835 En estimant empiriquement une valeur optimale $\hat{\tau}$ pour un nombre de tables L , on peut alors obtenir
 1836 une valeur ϕ pour une probabilité ϕ^* fixée, grâce à la méthode de Newton de résolution d'équations. ϕ^*
 1837 représente la probabilité d'obtenir au moins τ collisions à travers les L tables.

1838 6.1.2.3 Filtrage par les K plus proches voisins

1839 Afin de réduire l'impact des descripteurs visuels ambigus qui sont très fréquents dans la base (comme
 1840 dans les zones de texte par exemple), nous réduisons encore le nombre d'appariements par un filtrage
 1841 par K plus proches voisins, calculé avec une distance de Hamming sur les L clés de hachage concaténées.
 1842 De cette façon, l'ensemble final des appariements pour l'image I_Q est donc donné par :

$$Z_Q = \{(\mathbf{x}_q, \mathbf{x}_m) \in Y_Q \mid d_H(\mathbf{h}(\mathbf{x}_q), \mathbf{h}(\mathbf{x}_m)) \leq r_H(\mathbf{x}_q, K)\} \quad (6.7)$$

1843 où $r_H(\mathbf{x}_q, K)$ est la distance de Hamming de la $K^{\text{ème}}$ plus proche clé de hachage de $\mathbf{h}(\mathbf{x}_q)$. K a été
 1844 fixé empiriquement à 200 dans les expérimentations suivantes, c'est-à-dire du même ordre de grandeur
 1845 que lors de l'étape de recherche précise, où K est généralement fixé à 300.

1846 6.2 Extraction et hachage d'attributs de géométrie faible

1847 Une fois que l'on a obtenu un ensemble Z_Q d'appariements de descripteurs visuels pour la $Q^{\text{ème}}$
 1848 image I_Q , les scores individuels $z_o(\mathbf{x}_q)$ sont calculés pour chaque descripteur \mathbf{x}_q de l'image, au moyen
 1849 d'un hachage des attributs de géométrie faible provenant des descripteurs.

1850 Cette approche est inspirée de celle de Jégou et al. [109] qui estiment la cohérence géométrique des
 1851 appariements, en se basant sur les métadonnées associées à chaque descripteur SIFT (orientation et
 1852 échelle).

1853 La méthode proposée ici diffère en trois points principaux :

- 1854 – Premièrement, plutôt que de voter dans des histogrammes mono-dimensionnels (un par attribut),
 1855 nous votons dans un histogramme multidimensionnel creux. On augmente alors la distinctivité ;
 1856 – Deuxièmement, en plus de l'échelle et de l'orientation, nous ajoutons les coordonnées spatiales
 1857 des descripteurs dans l'image. Ceci permet d'inclure une contrainte de voisinage spatial, et favorise
 1858 les petits objets au détriment des plus gros (copies, arrière-plans, ...);
 1859 – Dernièrement, le vote est effectué à l'échelle de la base entière, contrairement à la version originale
 1860 [109] qui s'appliquait uniquement aux paires d'images ;

1861 6.2.1 Création de vecteurs de géométrie faible

1862 Plus concrètement, pour chaque appariement visuel $(x_q, x_m) \in Z_Q$, nous créons le vecteur de
 1863 géométrie faible suivant :

$$\Delta_{q,m} = (\Delta\theta_{q,m}, \Delta\sigma_{q,m}, \chi_q, \psi_q) \quad (6.8)$$

1864 où (χ_q, ψ_q) représente la position de x_q dans l'image I_Q . $\Delta\sigma_{q,m} = \sigma_m - \sigma_q$ est la différence des
 1865 échelles caractéristiques des points d'intérêts x_q et x_m . $\Delta\theta_{q,m}$ est le facteur de rotation entre les angles
 1866 caractéristiques des points d'intérêts x_q et x_m , obtenu par :

$$\Delta\theta_{q,m} = \frac{\arctan(\sin(\theta_m - \theta_q), \cos(\theta_m - \theta_q))}{\pi} \quad (6.9)$$

1867 Afin de créer un espace de vote multidimensionnel à partir de l'espace initial des attributs de
 1868 géométrie faible $\Delta_{q,m}$, nous proposons d'employer une famille de fonctions LSH adaptée de celle des
 1869 fonctions LSH Euclidiennes classiques [63].

1870 Notons que l'espace des vecteurs $\Delta_{q,m}$ est tout d'abord normalisé par composante de manière à ce
 1871 que chaque attribut varie dans l'intervalle $[-1, 1]$.

1872 6.2.2 Hachage des attributs de géométrie faible

1873 Les vecteurs ainsi normalisés sont ensuite hachés avec L' fonctions LSH distinctes, chacune composée
 1874 de k' projections aléatoires de la forme :

$$h'(\Delta) = \left\lfloor \frac{a \cdot \Delta + b}{w} \right\rfloor \quad (6.10)$$

1875 avec a étant un vecteur de nombre aléatoires gaussiens, et b étant un nombre aléatoire sélectionné
 1876 dans $[-1; 1]$.

1877 La légère modification introduite dans cette famille de fonctions est d'avoir une valeur adaptative w
 1878 pour chaque projection aléatoire. Un hachage « L_2 -sensible » n'est en effet dans ce cas pas approprié
 1879 pour créer notre espace de vote. On aimerait plutôt garantir une dynamique uniforme sur chacun des
 1880 axes de projection. On introduit donc les contraintes suivantes :

$$\min_{\Delta \in [-1; 1]^d} \frac{a \cdot \Delta + b}{w} = -2^\gamma, \quad \max_{\Delta \in [-1; 1]^d} \frac{a \cdot \Delta + b}{w} = +2^\gamma \quad (6.11)$$

1881 Ce qui nous donne $w = \frac{\|a\|_1}{2^\gamma}$ où γ est le nombre de bits utilisés pour quantifier chaque axe de projection.

1882 On obtient donc finalement :

$$h'(\Delta) = \left\lfloor 2^\gamma \frac{\mathbf{a} \cdot \Delta + b}{\|a\|_1} \right\rfloor \quad (6.12)$$

$$h'(\Delta) = \left\lfloor 2^\gamma \frac{\mathbf{a} \cdot \Delta + b}{\|a\|_1} \right\rfloor \quad (6.13)$$

1883 Et on note $h'_l(\Delta)$ la clé de hachage de longueur k' produite par la concaténation de k' fonction de
1884 hachage individuelle pour la $l^{\text{ème}}$ fonction de hachage.

1885 6.3 Calcul des scores de vraisemblance par accumulation des col- 1886 lisions

1887 6.3.1 Vote dans les accumulateurs

1888 Le processus de vote fonctionne ensuite simplement en gérant un conteneur de type associatif H_l^Q
1889 pour chacune des L' fonctions de hachage (on peut typiquement utiliser en C++ un container `std::map`).
1890 L'intérêt d'utiliser une telle structure est qu'il permet d'économiser énormément d'espace mémoire,
1891 étant donné que l'accumulateur H_l^Q est extrêmement creux.

1892 La « valeur clé » utilisée pour insérer les appariements visuels (incrémenter) dans l'accumulateur est
1893 définie par la paire :

$$h''_l(\Delta_{q,m}) = (I_{q,m}, h'_l(\Delta_{q,m})) \quad (6.14)$$

1894 où $h'_l(\Delta_{q,m})$ est la clé de hachage du vecteur de géométrie faible (c'est-à-dire l'estimation de la
1895 transformation géométrique), et $I_{q,m}$ est l'identifiant de l'image contenant les descripteurs \mathbf{x}_m appariés
1896 à \mathbf{x}_q .

1897 L'idée est que tous les appariements $\Delta_{q,m}$ voisins dans l'image I_Q et cohérents géométriquement se
1898 retrouvent avec la même clé de hachage (même partie de l'espace haché).

1899 Les valeurs associées à ces clés dans l'accumulateur représentent le nombre d'occurrences dans la clé
1900 de hachage identifiée par $h''_l(\Delta_{q,m})$.

1901 Tous les L' accumulateurs H_l^Q sont vides au commencement de la $Q^{\text{ème}}$ itération de l'algorithme.

1902 Les accumulateurs sont incrémentés en parallèle au fur et à mesure que l'on parcourt les
1903 appariements $(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q$ un par un, de la manière suivante :

$$H_l^Q(h''_l(\Delta_{q,m})) += f_L(\mathbf{x}_q, \mathbf{x}_m) \quad (6.15)$$

1904 où on rappelle que $f_L(\mathbf{x}_q, \mathbf{x}_m)$ est le nombre de collisions dans l'index visuel (cf. équation 6.5
1905 page 84). L'idée d'utiliser un vote pondéré par f_L plutôt qu'une simple incrémentation unitaire est de
1906 favoriser les correspondances visuelles les plus sûres, c'est-à-dire celles dont la distance visuelle est la
1907 plus petite.

1908 6.3.2 Calcul des scores individuels

1909 Après que tous les appariements visuels $(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q$ aient été insérés dans les accumulateurs, les
 1910 scores de vraisemblance $z_0(\mathbf{x}_q)$ sont calculés en comptant le nombre de collisions à travers chacun des L'
 1911 accumulateurs :

$$z_0(\mathbf{x}_q) = \sum_{l=1}^{L'} \sum_{(\mathbf{x}_q, \mathbf{x}_m) \in \mathbf{Z}_Q} H_l^Q(\mathbf{h}_l''(\Delta_{q,m})) \quad (6.16)$$

1912 De cette façon, $z_0(\mathbf{x}_q)$ mesure le nombre d'appariements visuels dans \mathbf{Z}_Q qui sont à la fois dans le
 1913 voisinage de \mathbf{x}_q et géométriquement cohérents avec les correspondances directes de \mathbf{x}_q (avec les $\Delta_{q,m}$).

1914 Plus la valeur $z_0(\mathbf{x}_q)$ est forte, plus la probabilité que \mathbf{x}_q appartienne à un objet fréquent est élevée.

1915 6.4 Solution alternative : ajout de contraintes géométriquement 1916 faibles à GmH

1917 Notre algorithme de vote sur les attributs de géométrie faible peut se généraliser à d'autres types
 1918 d'appariements visuels en entrée. Nous proposons ainsi une variante en aval de l'utilisation d'un
 1919 algorithme de type GmH.

1920 Dans ce cas, on obtient non plus des appariements individuels entre un vecteur \mathbf{x}_q et
 1921 un vecteur \mathbf{x}_m , mais des appariements entre *sketches* (groupes de s appariements individuels),
 1922 c'est-à-dire des appariements entre des *sketches* requêtes $\{\mathbf{x}_{q_1}, \mathbf{x}_{q_2}, \dots, \mathbf{x}_{q_s}\}$ et des *sketches* candidats
 1923 $\{\mathbf{x}_{m_1}, \mathbf{x}_{m_2}, \dots, \mathbf{x}_{m_s}\}$.

1924 Ces appariements entre *sketches* sont obtenus de la manière décrite dans le papier original de GmH
 1925 [160] ainsi que dans la section 3.4 page 50. Dans chaque image, nous créons k *sketches*, composés de
 1926 $s = 2$ min-Hash, dont le deuxième est choisi parmi les voisins du premier respectant certaines contraintes
 1927 géométriques.

1928 Afin d'ajouter nos contraintes géométriquement faibles à GmH, notre proposition est de créer pour
 1929 chaque paire de *sketches* appariés $\{\mathbf{x}_{q_1}, \mathbf{x}_{q_2}, \dots, \mathbf{x}_{q_s}\}$ et $\{\mathbf{x}_{m_1}, \mathbf{x}_{m_2}, \dots, \mathbf{x}_{m_s}\}$, s vecteurs de géométrie
 1930 faible $\{\Delta_{q_1, m_1}, \Delta_{q_2, m_2}, \dots, \Delta_{q_s, m_s}\}$. Ces s vecteurs sont ensuite utilisés individuellement pour voter
 1931 dans les accumulateurs, c'est-à-dire sans aucune autre modification de notre algorithme de base.

1932 Cette solution sera évaluée par la suite en comparaison de notre algorithme original.

1933 6.5 Expérimentations

1934 Le protocole d'évaluation des algorithmes présentés dans ce chapitre est le même que celui décrit au
 1935 chapitre précédent dans la section 5.3.1 page 77. Nous utilisons donc la précision et le rappel calculés sur
 1936 la base FlickrBelgaLogos, présentée dans la section 7.1 page 95.

1937 **6.5.1 Étude paramétrique**

1938 **6.5.1.1 Paramétrage du hachage des descripteurs visuels**

1939 Les deux principaux paramètres du hachage des descripteurs visuels sont le nombre de tables de
 1940 hachage (RMMH) L , et le seuil sur la probabilité de collision ϕ . Pour $L = 64$, le plus grand nombre de
 1941 tables testé, on estime empiriquement un seuil optimal $\hat{\tau} = 7$. Ainsi, en fixant une probabilité raisonnable
 1942 $\phi^* = 0.5$, et par la méthode de Newton appliquée à l'équation 6.6 page 84, on obtient $\phi \approx 0.10$. Ceci
 1943 permet maintenant de déduire le paramètre $\hat{\tau}$ estimé en fonction du nombre de tables de hachage L . Le
 1944 tableau 6.1 présente les résultats de ce calcul. En pratique, pour des raisons de temps de calcul, nous
 1945 avons préféré utiliser des valeurs de τ légèrement plus élevées (voir tableau 6.1) qui permettent un
 1946 filtrage plus fort, sans diminuer significativement les performances.

L	1	2	4	8	16	32	64
$\hat{\tau}$	1	1	1	1	2	4	7
τ	1	2	2	2	3	4	7

TABLE 6.1 – Réglage du paramètre $\hat{\tau}$ optimum théorique en fonction de L , et τ utilisé en pratique.

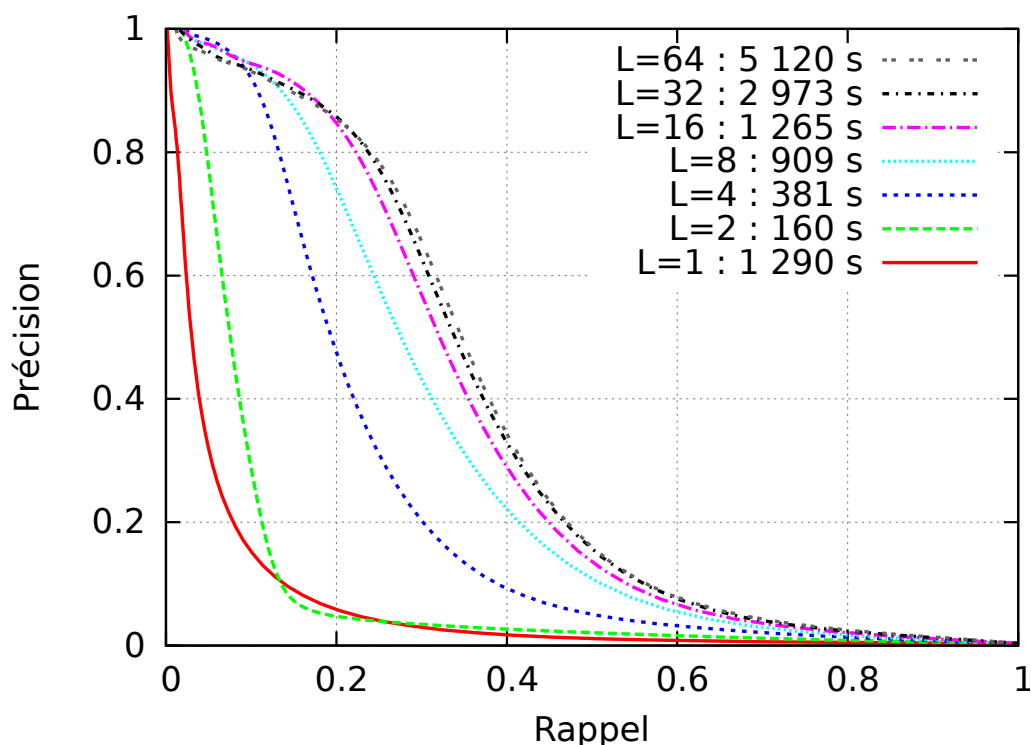


FIGURE 6.1 – Réglage du nombre de tables de hachage L

1947 Les résultats présentés dans la figure 6.1 montrent qu'utiliser plus de 16 tables de hachage visuel n'est

1948 pas très rentable, étant donné que la progression en termes de précision/rappel devient très faible, alors
 1949 que le temps de calcul augmente linéairement.

1950 Il faut noter que le temps nécessaire pour calculer les scores de vraisemblance avec une seule table
 1951 de hachage est bien plus important qu'avec deux tables. Ceci s'explique par l'impossibilité de filtrer sur la
 1952 fréquence de collision inter-tables de hachage f_L lorsque $L = 1$.

1953 **6.5.1.2 Longueur des clés de hachage des attributs de géométrie faible**

1954 Dans cette expérimentation, nous étudions l'effet du nombre de fonctions de hachage k' utilisées
 1955 pour calculer les clés de hachage des attributs de géométrie faible, ainsi que le nombre de bits γ utilisés
 1956 pour la quantification scalaire de chaque projection.

1957 Pour améliorer la lisibilité du graphique de la figure 6.2, les courbes ont été groupées manuellement
 1958 en quatre groupes (identifiés par quatre couleurs), selon leurs performances. Les meilleurs compromis
 1959 sont obtenus avec $\gamma \in [3, 5]$ et $k' \in [3, 10]$. On remarque que l'utilisation d'une forte valeur pour γ est
 1960 plutôt défavorable, étant donné que cela diminue la probabilité d'obtenir un nombre de collisions
 1961 significatif. Au contraire, une faible valeur de γ peut être contrebalancée par un plus grand nombre de
 1962 fonctions de hachage k' , afin d'augmenter la distinctivité de l'histogramme. Dans les expérimentations
 1963 suivantes, nous utiliserons les paramètres de la courbe noire ($k' = 6$ et $\gamma = 4$), pour un total de 24 bits
 1964 par clé de hachage.

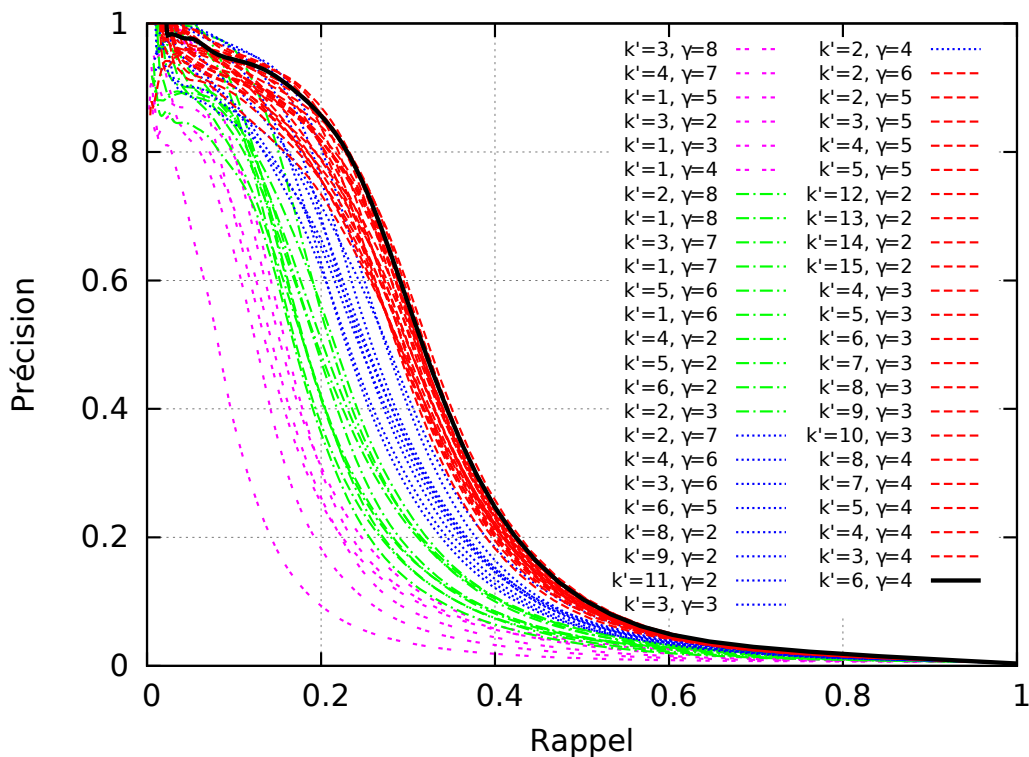


FIGURE 6.2 – Effets de la longueur des clés de hachage de la géométrie faible

1965 **6.5.1.3 Nombre d'accumulateurs**

1966 L'évaluation présentée dans la figure 6.3 compare la qualité obtenue avec différentes valeurs de L' ,
 1967 c'est-à-dire le nombre d'accumulateurs (histogrammes). On peut également observer les performances
 1968 obtenues avec une simple grille multidimensionnelle (à la place du hachage par projections aléatoires),
 1969 utilisant le même nombre de bits. On remarque que les histogrammes basés LSH fonctionnent bien mieux
 1970 qu'avec la grille multidimensionnelle.

1971 En regardant la valeur de L' , on voit nettement qu'il n'est pas rentable d'utiliser plus d'un
 1972 accumulateur, contrairement au cas des espaces à très grandes dimensions.

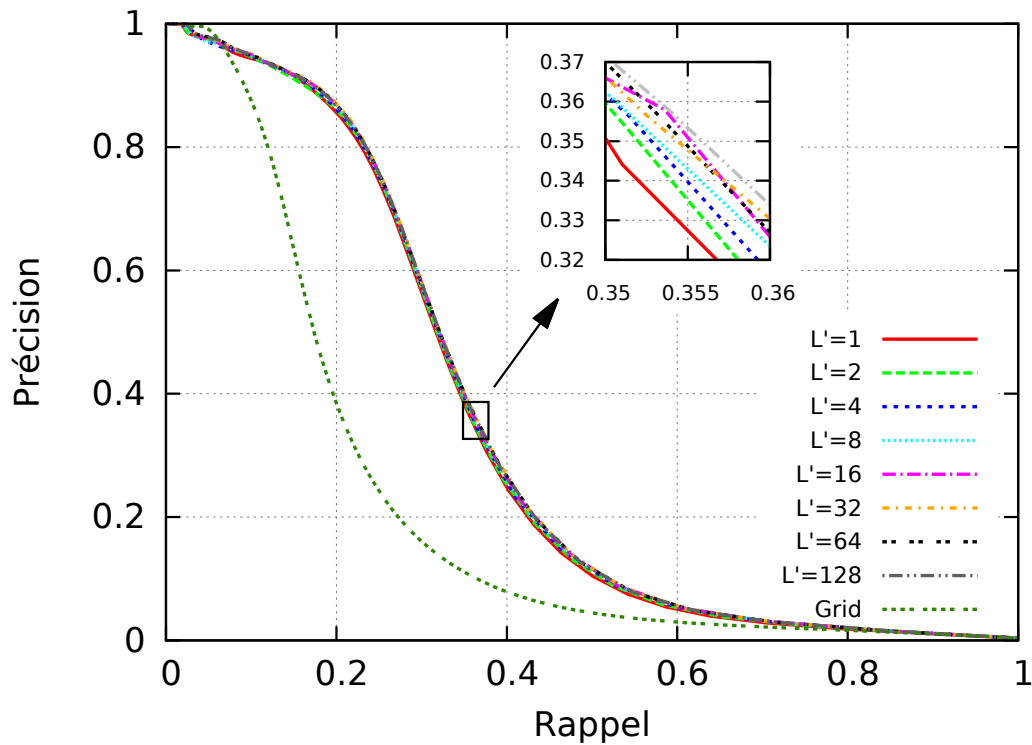


FIGURE 6.3 – Influence du nombre d'accumulateurs

1973 **6.5.2 Apport des attributs de géométrie faible**

1974 Cette évaluation vise à mesurer la contribution des différents attributs du vecteur de géométrie faible
 1975 Δ . On peut voir dans la figure 6.4 page ci-contre que l'utilisation de la géométrie faible améliore très
 1976 significativement la précision et le rappel. La courbe 0,0,0,0 mesure les performances en l'absence de
 1977 géométrie faible, en utilisant simplement le hachage visuel et les trois règles de filtrage des appariements
 1978 visuels. Les autres courbes ont été obtenues à partir d'un sous-ensemble d'attributs de géométrie faible
 1979 (angle : θ , échelle : σ , position : χ et ψ). On observe que parmi les différents attributs, ce sont les deux
 1980 attributs de position spatiale qui procurent le gain le plus significatif. On voit que tous les attributs
 1981 apportent de l'information, et que cette information est complémentaire.

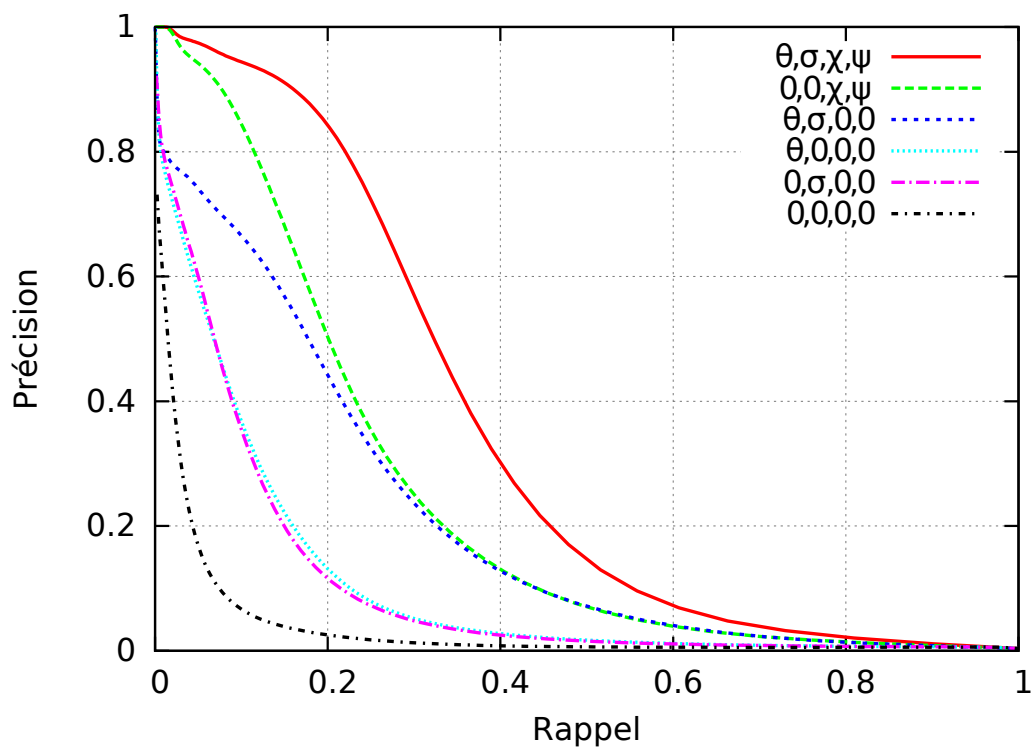


FIGURE 6.4 – Contribution des attributs de Géométrie Faible $(\theta, \sigma, \chi, \psi)$ sur la précision et le rappel, pour la méthode à base de hachage visuel.

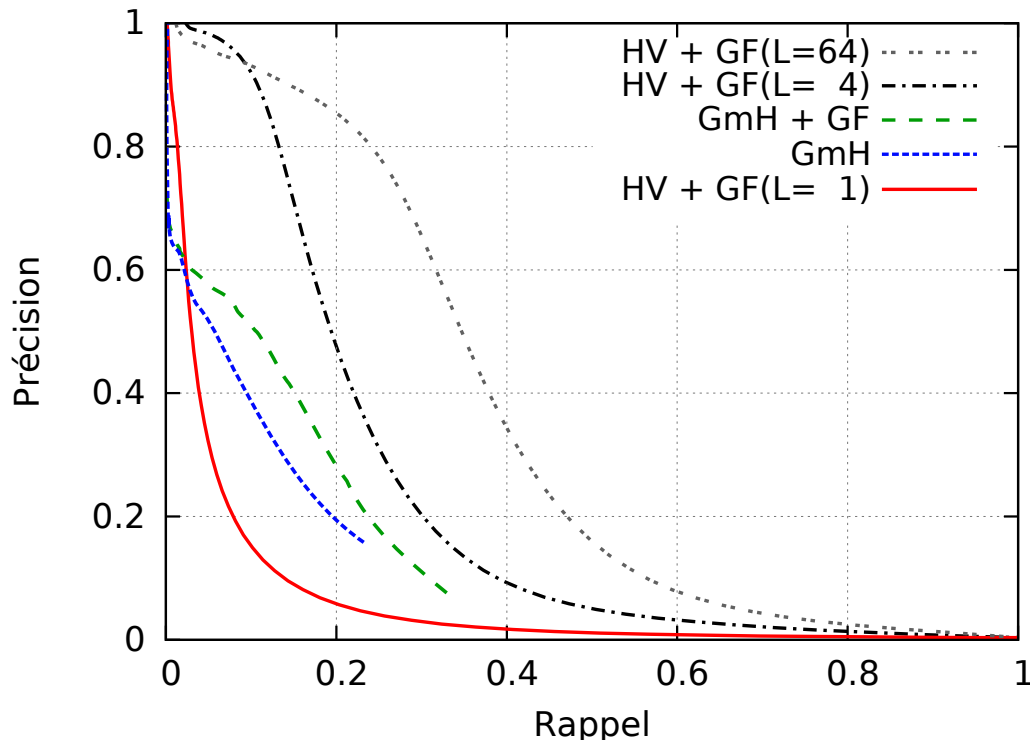


FIGURE 6.5 – Contribution de la géométrie faible sur les méthodes de calcul de score de vraisemblance. HV : Hachage Visuel, GF : Géométrie Faible, L : nombre de tables de hachage visuel, GmH : Geometric min-Hashing.

1982 6.5.3 Comparaison de notre méthode à GmH

1983 Dans la figure 6.5, nous comparons les performances de notre méthode « Hachage Visuel (HV) +
 1984 Géométrie Faible (GF) » avec la méthode état de l'art « Geometric min-Hashing (GmH) » présentée dans la
 1985 section 3.4 page 50, avec ou sans la contribution du vote sur la géométrie faible (cf. section 6.4 page 87).

1986 On constate que notre méthode fonctionne bien mieux que Geometric min-Hashing, lorsqu'on utilise
 1987 notre hachage des attributs de géométrie faible avec $L = 4$ tables de hachage visuel. Le gain apporté par
 1988 la géométrie faible à GmH reste trop faible pour justifier pleinement son emploi. On peut supposer que
 1989 ceci vient du fait que GmH utilise déjà une contrainte de voisinage pour générer ses *sketches*, donc le
 1990 hachage sur la position devient redondant et seul le gain sur l'angle θ et l'échelle σ est visible.

1991 On remarque que l'on peut facilement augmenter le nombre L de tables de hachage visuel afin
 1992 d'améliorer encore le compromis précision/rappel.

1993

Troisième partie

1994

Expérimentations et Applications

1995

Chapitre 7

1996

Découverte et fouille de logos

1997

7.1 Le corpus FlickrBelgaLogos

1998

L'évaluation des performances des algorithmes de découverte et fouille d'objets visuels est un problème plus complexe que celui de la recherche d'objets visuels. En effet, la recherche d'objets visuels peut être évaluée par la réalisation d'une vérité terrain construite sur la base d'une liste de requêtes. Plus précisément, cela consiste, pour chaque requête de la liste (une requête étant une zone d'image), à fournir la liste des images (ou zones d'images) dont on considère qu'elles sont visuellement similaires à la requête.

2004

Ce type de vérité terrain n'est malheureusement pas adapté aux problèmes de découverte et fouille d'objets visuels. En effet, il est dans ce cas nécessaire de disposer d'une vérité terrain couvrant la totalité des objets visuels se répétant dans la collection d'images, alors qu'on ne dispose que d'une vérité terrain partielle (construite sur la base d'une liste de requêtes). En outre, il faut remarquer que dans les images naturelles, il est très difficile de définir une liste exhaustive de tous les objets visuels répétés. Cela supposerait pour le créateur de la vérité terrain, qu'à chaque image, il soit capable de déterminer si l'un des multiples objets visuels composant l'image se répète ne serait-ce qu'une seule fois dans tout le reste de la collection d'images.

2012

A notre connaissance, il n'existait pas de vérité terrain adaptée à notre problème au début de ce travail.

2014

La construction d'une telle vérité terrain nous paraissant trop difficile à réaliser, nous avons préféré adapter la collection d'images plutôt que d'adapter la vérité terrain. L'idée était de contrôler les répétitions d'objets visuels tout en conservant une collection d'images naturelles. Pour atteindre cet objectif, nous avons introduit des répétitions dans une collection de 10 000 images n'en contenant pas, les objets visuels introduits étant des logos de la base BelgaLogos, et les images « sans répétitions » venant de Flickr.

2019

Logo	Illustration	# Instances		Logo	Illustration	# Instances	
		Total	OK			Total	OK
Adidas		1043	147	Mercedes		279	86
Adidas-text		178	63	Nike		2242	235
Airness		120	11	Peugeot		7	5
Base		248	162	Puma		800	157
BFGoodrich		308	86	Puma-text		80	27
Bik		270	65	Quick		253	57
Bouigues		32	14	Reebok		66	18
Bridgestone		105	31	Roche		2	2
Bridgestone-text		201	64	Shell		236	123
Carglass		65	18	SNCF		10	7
Citroen		242	78	Standard-Liege		381	98
Citroen-text		331	197	StellaArtois		28	20
CocaCola		73	40	TNT		183	102
Cofidis		90	45	Total		96	78
Dexia		626	235	US_President		14	14
ELeclerc		20	15	Umbro		659	153
Ferrari		213	77	Veolia		77	12
Gucci		4	2	VRT		18	10
Kia		242	141	Somme totale		9842	2695

TABLE 7.1 – Logos présents dans la vérité terrain de la base BelgaLogos

2020 L'introduction de ces logos s'est faite en copiant et collant les instances (chaque pixel de leur rectangle
2021 englobant) depuis les images de BelgaLogos vers celles de Flickr, à des coordonnées spatiales aléatoires,
2022 mais sans autres transformations géométriques du type rotation ou changement d'échelle. Les instances
2023 originales possèdent en effet déjà des angles et des échelles variés, mais surtout naturels.

2024

2025 La vérité terrain originale de BelgaLogos ne fournissant pas les coordonnées spatiales des instances,
2026 mais seulement les noms des images, un travail préliminaire de complètement de la vérité terrain a dû
2027 être effectué. Nous avons pour cela fourni manuellement les coordonnées des rectangles englobants
2028 de chacune des 9 842 instances appartenant à 37 objets visuels (logos) différents. Chacune de ces 9
2029 842 instances a ensuite été évaluée par trois utilisateurs afin de la classer (par un vote majoritaire)
2030 en « conservée » ou « rejetée » selon qu'ils étaient capables d'identifier aisément l'objet visuel ou non,
2031 sans avoir accès au contexte de l'image puisque seul le contenu du rectangle englobant l'instance était
2032 présenté. Au final, seules 2 695 instances ont été conservées. Le tableau [7.1 page ci-contre](#) présente
2033 la liste des logos annotés dans la vérité terrain, avec le nombre total d'instances, et le nombre (OK)
2034 d'instances conservées.

2035

2036 Afin de garantir l'absence de répétitions (ou tout du moins les minimiser) dans les images
2037 de Flickr, celles-ci ont été sélectionnées aléatoirement dans 10 000 régions différentes du
2038 globe (c'est-à-dire une image par région) grâce aux informations de géolocalisation associées aux
2039 photographies. Ces cases géographiques sont des carrés de 2,5° de longitude et latitude. Nous
2040 garantissons également que les 10 000 images proviennent de 10 000 utilisateurs différents (grâce
2041 aux identifiants Flickr). Ces contraintes permettent ainsi de minimiser la probabilité d'obtenir des
2042 répétitions d'objets visuels. Nous minimisons principalement les répétitions architecturales (grâce à
2043 la géographie), et d'autres objets divers, car en multipliant les utilisateurs, nous augmentons la di-
2044 versité des centres d'intérêts, et nous évitons les répétitions d'objets personnels (vêtements, véhicules, ...).

2045

2046 Nous sommes toutefois bien conscients des limites de la méthode, qui par exemple ne tient
2047 pas compte du phénomène de la mondialisation économique et culturelle, qui provoque l'apparition
2048 de logos « Coca-Cola » ou « McDonald's » aux quatre coins du globe (ou en l'occurrence aux 10 000 régions).

2049

2050 Une solution pour identifier ces répétitions involontaires pourrait se baser sur la construction
2051 d'un graphe d'appariement entre toutes les images d'une collection (cf. [3.2 page 49](#)), en requêtant
2052 exhaustivement les images avec notre système de recherche d'objets visuels. Les images représentées par
2053 les nœuds isolés pourraient alors être considérées comme n'ayant aucune similarité avec le reste de
2054 la collection. Nous pourrions également conserver une seule image par composante connexe. Même
2055 si cette solution pourrait s'avérer idéale pour notre évaluation, nous l'avons rejetée pour une raison
2056 simple : elle n'est pas objective. En effet, en usant de cette approche, nous diminuerions le nombre de
2057 faux positifs lors de l'évaluation de notre méthode, mais probablement moins que lors de l'évaluation de
2058 méthodes concurrentes, utilisant des composants différents (description, indexation, ...).

2059 La figure 7.1 présente un échantillon des images de la base FlickrBelgaLogos.

2060



FIGURE 7.1 – Un échantillon des 10 000 images de la base FlickrBelgaLogos

2061 7.2 Vitesse de convergence en fonction des différents scores de 2062 vraisemblance

2063 L'objectif de cette section est d'analyser l'influence des scores de vraisemblance sur la vitesse de
2064 convergence de l'algorithme RANSAS, c'est-à-dire d'analyser les performances de l'ensemble du processus
2065 de découverte, et non plus seulement la pertinence des scores de vraisemblance. Comme nous l'avons vu
2066 précédemment, le calcul des scores de vraisemblance a pour but de focaliser la création de requêtes dans
2067 les zones à forte probabilité de contenir des objets $\{c, f\}$ -fréquents. Les premières requêtes issues de
2068 l'échantillonnage pondéré ont donc de grandes chances de découvrir des objets $\{c, f\}$ -fréquents, tandis
2069 que ces chances s'amenuisent au fur et à mesure des échantillonnages successifs, en raison du caractère
2070 adaptatif de la pondération (cf. section 4.2.1 page 58). Autrement dit, plus le nombre T d'itérations de
2071 RANSAS augmente, et moins l'influence des scores de vraisemblance est mesurable.

2072 Nous verrons donc dans la section 7.2.2 page ci-contre que les meilleurs scores de vraisemblance
2073 permettent de découvrir plus rapidement des objets $\{c, f\}$ -fréquents, mais que les performances
2074 finissent par converger après un grand nombre d'itérations.

2075 7.2.1 Protocole d'expérimentations

2076 Les serveurs utilisés dans ces expériences sont équipés de 2 processeurs hexa-cores (Intel X5660). Les
2077 temps de calcul sont donnés en temps absolu. Les mesures utilisées dans les expérimentations suivantes
2078 sont le « rappel objet » (pour évaluer le problème de la découverte d'objets), le « rappel instance » (pour

2079 évaluer le problème de la fouille d'objets), et la « précision instance ». Ces mesures sont données en
 2080 fonction du nombre d'itérations T de RANSAS. Elles dépendent en grande partie des performances de
 2081 notre système de recherche d'objets visuels (cf. 4.5 page 65), mais les performances relatives sont
 2082 déterminées par les différents scores de vraisemblance utilisés.

2083 Le rappel instance est défini comme le nombre d'instances d'objets $\{c, f\}$ -fréquents uniques
 2084 retrouvées dans au minimum un groupe de résultats de l'étape de recherche de RANSAS, à travers tous
 2085 les résultats obtenus au moment de l'itération considérée, divisé par le nombre d'instances de la vérité
 2086 terrain.

2087 Le rappel objet concerne quant à lui le pourcentage d'objets $\{c, f\}$ -fréquents uniques retrouvés, de la
 2088 même manière que précédemment.

2089 La précision instance mesure le nombre d'instances d'objets $\{c, f\}$ -fréquents uniques retrouvées dans
 2090 au minimum un groupe de résultats de l'étape de recherche de RANSAS, divisé par le nombre de résultats
 2091 retournés au moment de l'itération considérée.

2092 La base d'images utilisée est FlickrBelgaLogos, présentée dans la section 7.1 page 95.

2093 La recherche d'objets a été paramétrée avec :

- 2094 – SIFT : octave -1 , produisant 52 millions de descripteurs ;
- 2095 – RMMH : noyau « produit scalaire », $M = 40$, et 128 bits par descripteur ;
- 2096 – APMP-LSH : 300 KNN, $\alpha = 0.80$;
- 2097 – Vérification géométrique : seuillage a contrario égal à 0.95 ;

2098 7.2.2 Évaluation

2099 Dans les graphiques des figures 7.2a, 7.2b page suivante et 7.3 page 101, on observe l'augmentation
 2100 du rappel (objet ou instance) et la diminution de la précision en fonction du nombre d'itérations de
 2101 RANSAS (en échelle logarithmique), et pour différents scores de vraisemblance, ainsi que dans le cas d'un
 2102 échantillonnage uniforme, et d'un score de vraisemblance parfait.

2103 Afin de réaliser un échantillonnage uniforme, nous utilisons un score de vraisemblance tel que sa
 2104 fonction de probabilité de masse soit égale à :

$$p_0(\mathbf{x}_i) = 1/N, \forall \mathbf{x}_i \in \mathbf{X}$$

2105 où \mathbf{X} est l'ensemble des N descripteurs de la base.

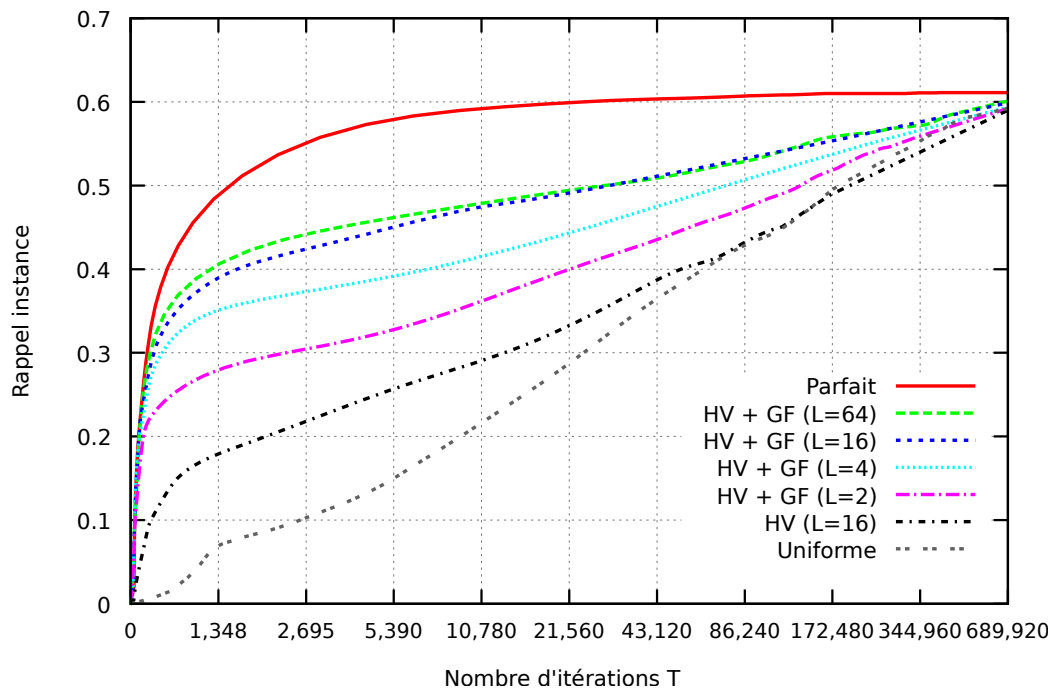
2106

2107 Tandis que le score de vraisemblance parfait donne la fonction de probabilité de masse suivante :

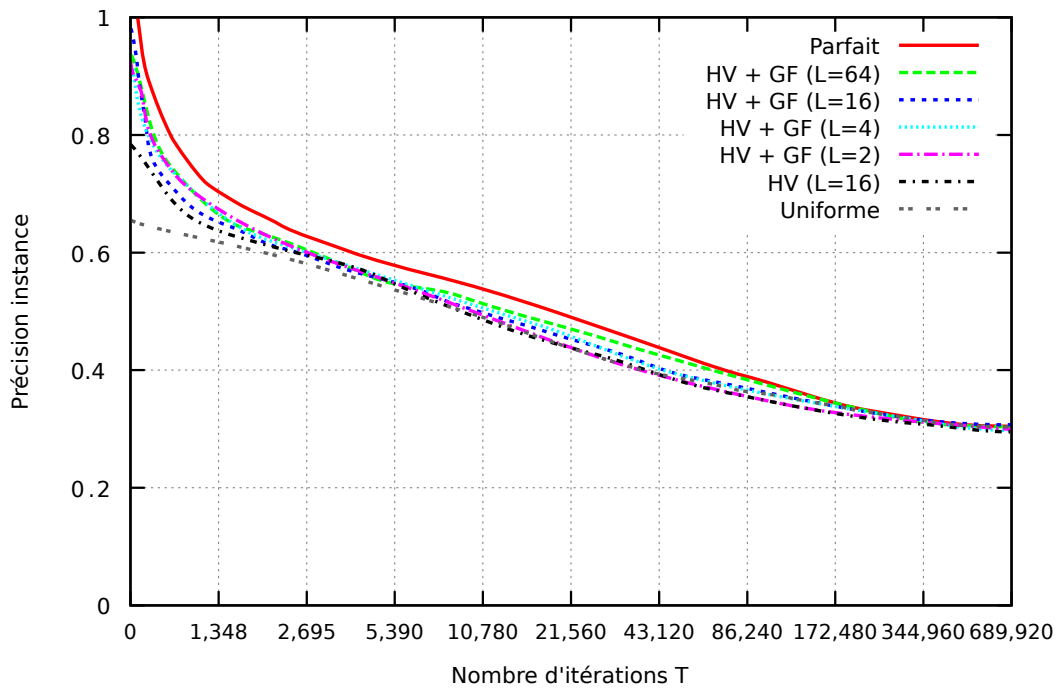
$$p_0(\mathbf{x}_i) = 0, \forall \mathbf{x}_i \notin \{\mathbf{X}_s^m\} \text{ et } p_0(\mathbf{x}_i) = \frac{1}{|\{\mathbf{X}_s^m\}|}, \forall \mathbf{x}_i \in \{\mathbf{X}_s^m\}$$

2108 avec $\{\mathbf{X}_s^m\}$ l'ensemble des descripteurs appartenant à des instances de la vérité terrain.

2109 La courbe « Uniforme » représente le rappel minimum que l'on devrait obtenir avec n'importe quel
 2110 score de vraisemblance. En effet, un score donnant une courbe inférieure à celle-ci signifierait que ce
 2111 score défavorise les instances de la vérité terrain au profit du reste de la collection, ce qui serait



(a) Rappel instance



(b) Précision instance

FIGURE 7.2 – Rappel et de la précision instance en fonction du nombre d'itérations de RANSAS
 HV : Hachage Visuel, GF : Géométrie Faible, L : nombre de tables de hachage visuel.

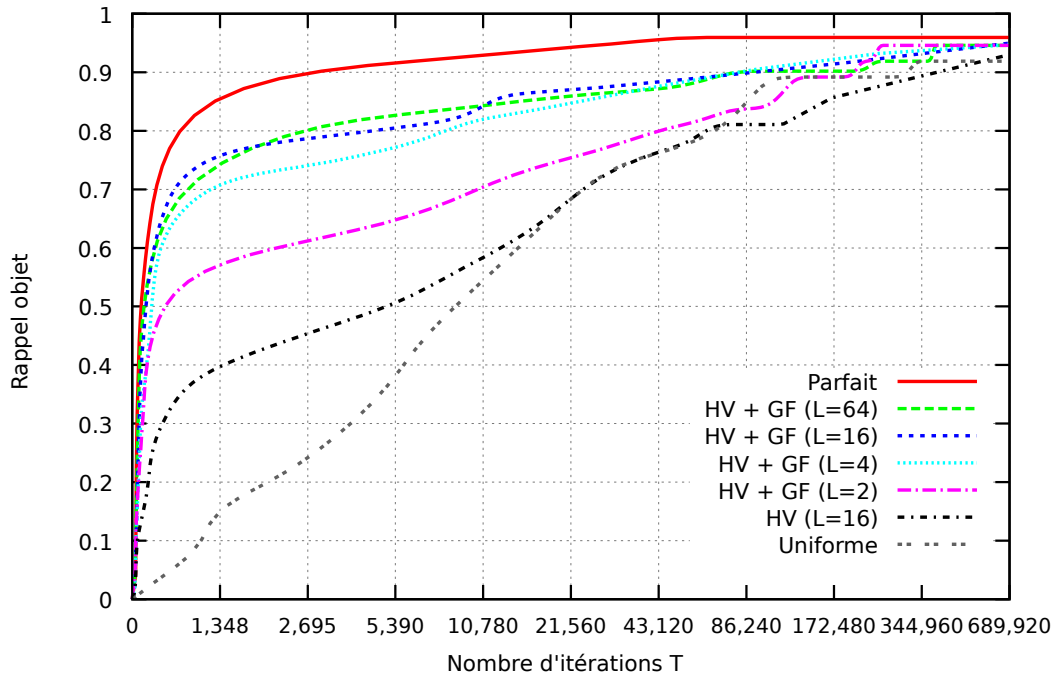


FIGURE 7.3 – Rappel objet en fonction du nombre d'itérations de RANSAS
 HV : Hachage Visuel, GF : Géométrie Faible, L : nombre de tables de hachage visuel.

2112 exactement l'inverse de ce que l'on recherche. A contrario, on ne peut espérer créer de scores de
 2113 vraisemblance donnant une courbe supérieure à la courbe « Parfait », tout du moins pas sans modifier la
 2114 méthode de recherche d'objets, ou ses paramètres. Ces deux courbes définissent donc l'intervalle dans
 2115 lequel peuvent évoluer les courbes de rappel obtenues à partir de nos différents scores de vraisemblance.

2116 Les courbes montrent tout d'abord que notre score de vraisemblance (avec typiquement $L = 16$)
 2117 est bien plus proche du score « Parfait », que du score « Uniforme ». Ceci prouve donc que les scores
 2118 $z_0(x_i)$ quantifient bien la vraisemblance qu'un descripteur x_i appartienne à un objet $\{c, f\}$ -fréquent de
 2119 la vérité terrain.

2120 On constate également que toutes les courbes finissent par converger au bout d'un grand nombre
 2121 d'itérations (environ 600 000), à environ 91% d'objets découverts, et 61% d'instances découvertes.
 2122 Cette convergence montre les performances limites de la méthode de recherche d'objets. Avec autant
 2123 d'itérations, on a probablement requêté la plupart des instances d'objets de la vérité terrain, et il est donc
 2124 inutile de poursuivre plus loin l'expérimentation.

2125 De plus, on observe que l'augmentation du rappel est bien plus forte lorsqu'un petit nombre
 2126 d'itérations ont été effectuées. Ainsi, après seulement 1 348 itérations (soit la moitié du nombre
 2127 d'instances de la base FlickrBelgaLogos), nos meilleurs scores de vraisemblance ont permis la découverte
 2128 de plus de 70% des objets, et de la moitié des instances retrouvables (par notre méthode), tandis qu'il faut
 2129 environ 32 fois plus d'itérations pour obtenir de telles performances avec un échantillonnage uniforme.

2130 On peut noter que la précision instance diminue au fur et à mesure des itérations de RANSAS, et

2131 converge finalement pour tous les scores de vraisemblance vers 30% de précision. Ceci s'explique par le
2132 fait qu'à chaque itération, on diminue les poids (scores) des descripteurs correspondant à la requête
2133 et aux résultats. A l'initialisation, les scores les plus forts sont ceux représentant les descripteurs des
2134 instances les plus faciles à retrouver (par notre méthode). Après un grand nombre d'itérations, les scores
2135 les plus forts représentent en majorité des images sans instances (de la vérité terrain), ou des instances
2136 d'objets difficiles à retrouver. La précision de ces dernières requêtes est donc beaucoup plus faible. La
2137 convergence vers 30% correspond à la précision minimale atteinte par notre système de recherche
2138 d'objets visuels lorsqu'on requête principalement des zones ne comportant pas d'instances de la vérité
2139 terrain.

2140

2141 Le tableau [7.2 page suivante](#) donne un aperçu du compromis temps/qualité de notre méthode
2142 comparé à un échantillonnage uniforme. On observe qu'avec tous les coûts de calcul inclus (hormis la
2143 description, qui est une constante), notre méthode permet l'obtention d'un facteur d'accélération de la
2144 découverte pouvant aller jusqu'à 18 (dans nos expérimentations). Ce tableau montre qu'il faut choisir
2145 un nombre L de tables de hachage visuel en fonction du rappel souhaité. Pour un rappel (objet ou
2146 instance) faible, on obtient les meilleurs facteurs d'accélération avec peu de tables (typiquement 2 ou
2147 4). Par contre, pour obtenir un rappel fort, mieux vaut prendre environ 16 tables. On remarque que
2148 l'utilisation de 64 tables n'est jamais pertinente (toujours dans cette expérimentation). Le temps de calcul
2149 du score de vraisemblance est en effet dans ce cas tellement élevé qu'il est préférable de faire davantage
2150 d'itérations de RANSAS avec un moins bon score de vraisemblance. Au total, il faut environ une quinzaine
2151 d'heures pour découvrir 90% des objets de la vérité terrain, et moins de neuf heures pour découvrir 50%
2152 des instances.

2153 Des exemples de requêtes issus de l'échantillonnage de RANSAS et de certains de leurs résultats sont
2154 présentés dans la figure [7.4 page 104](#).

	U	HV (L=16)	HV+GF (L=2)	HV+GF (L=4)	HV+GF (L=16)	HV+GF (L=64)	
Temps de hachage	627s	1,567s	634s	641s	1,567s	7,292s	
Temps de calcul du score de vraisemblance	0s	1,128s	163s	320s	1,280s	5,120s	
Temps total	U	HV (L=16)	HV+GF (L=2)	HV+GF (L=4)	HV+GF (L=16)	HV+GF (L=64)	
Rappel instance	0.20	8,858s	4,482s x2.0	919s x9.6	1,045s x8.5	2,925s x3.0	12,491s x0.7
	0.30	23,588s	14,753s x1.6	2,874s x8.2	1,282s x18.4	3,073s x7.7	12,597s x1.9
	0.40	55,825s	47,731s x1.2	19,676s x2.8	7,471s x7.5	4,113s x13.6	13,225s x4.2
	0.50	159,973s	158,105s x1.0	120,289s x1.3	60,146s x2.7	30,751s x5.2	37,066s x4.3
Temps total	U	HV (L=16)	HV+GF (L=2)	HV+GF (L=4)	HV+GF (L=16)	HV+GF (L=64)	
Rappel objet	0.30	4,078s	3,054s x1.3	901s x4.5	1,039s x3.9	2,916s x1.4	12,467s x0.3
	0.40	5,724s	3,780s x1.5	936s x6.1	1,108s x5.2	2,932s x2.0	12,480s x0.5
	0.50	7,854s	7,111s x1.1	1,136s x6.9	1,163s x6.8	2,982s x2.6	12,520s x0.6
	0.60	14,227s	14,828s x1.0	2,515s x5.7	1,237s x11.5	3,063s x4.6	12,635s x1.1
	0.70	20,661s	23,081s x0.9	9,898s x2.1	1,830s x11.3	3,248s x6.4	12,916s x1.6
	0.80	65,831s	56,691s x1.2	37,338s x1.8	7,987s x8.2	7,110s x9.3	14,407s x4.6
	0.90	281,205s	199,128s x1.4	184,419s x1.5	55,234s x5.1	96,441s x2.9	78,261s x3.6

TABLE 7.2 – Temps et accélération (en vert) obtenus avec différents scores de vraisemblance.

U : Uniforme, HV : Hachage Visuel, GF : Géométrie Faible, L : nombre de tables de hachage visuel. Le temps de hachage est celui nécessaire au hachage des descripteurs locaux. Il faut au minimum 128 bits pour la recherche d'objets. Ensuite, pour chaque valeur de rappel considérée, on donne le temps total (hachage + calcul du score + recherche d'objets). Les facteurs d'accélération sont donnés en comparaison de ceux obtenus avec un échantillonnage uniforme.

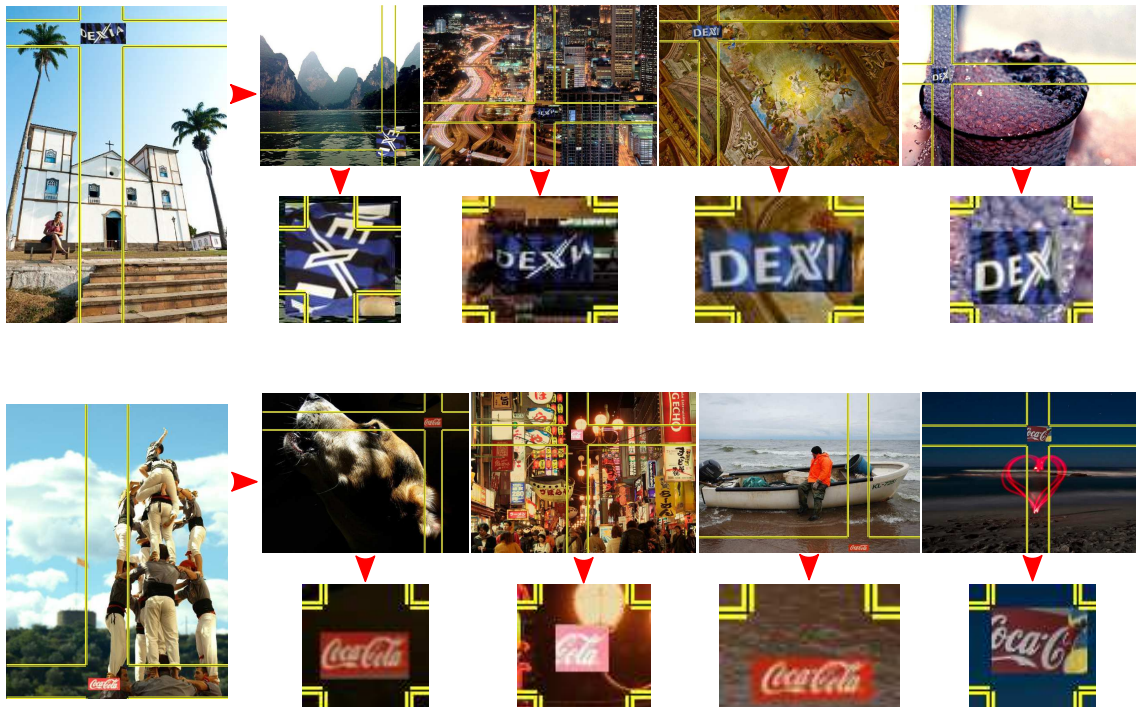


FIGURE 7.4 – Les deux premières requêtes issues de l'échantillonnage pondéré par le score de vraisemblance HV + GF (L=64), et quatre de leurs résultats de recherche

2155 7.3 Classification des instances par un algorithme de regroupe- 2156 ment

2157 Notre méthode de découverte et fouille d'objets propose en sortie une liste de résultats (rectangles
2158 englobant les instances dans les images) pour chaque requête issue de la phase d'échantillonnage. Afin
2159 d'améliorer le rappel instance, nous avons vu (cf. figure 7.2a page 100) qu'il était nécessaire d'effectuer
2160 un certain nombre d'itérations, afin de rechercher plusieurs instances de chaque objet visuel.

2161 Cette méthode génère donc en sortie des groupes d'instances représentant des sous-parties d'objets
2162 $\{c, f\}$ -fréquents. Le nombre de ces groupes peut rapidement devenir difficile à analyser pour un
2163 opérateur humain, il est donc souhaitable de lui proposer de rassembler les différents groupes d'instances
2164 appartenant à un même objet $\{c, f\}$ -fréquent.

2165 Chaque instance découverte est reliée à au moins une autre instance, et on connaît leur score de
2166 similarité. Il est donc possible de construire un graphe d'appariement entre instances.

2167 La construction d'un tel graphe permet ensuite d'identifier des groupes d'instances plus complets, par
2168 l'extraction des composantes connexes. Si le graphe présente trop peu de composantes connexes, il faut
2169 alors envisager un processus de regroupement (*clustering*) plus complexe.

2170 Dans la section suivante, nous détaillerons une solution envisageable pour résoudre ce problème,
2171 utilisant l'outil MCL-edge [167] de Stijn van Dongen, implémentant l'algorithme MCL (*Markov Cluster*

2172 *algorithm*) [168].

2173 7.3.1 L'algorithme MCL

2174 Selon Peyronnet [169] :

2175 L'algorithme MCL (*Markov Cluster algorithm*) est un algorithme de partitionnement
2176 rapide et capable de prendre en entrée de très grands graphes [millions de nœuds et
2177 centaines de millions d'arêtes]. Il est basé sur la simulation de flots stochastiques dans un
2178 graphe. Il a été mis au point par Stijn van Dongen au Centre for Mathematics and Computer
2179 Science (CWI).

2180 Les *clusters* naturels dans un graphe sont caractérisés par la présence d'un grand nombre
2181 d'arcs entre les membres de ce *cluster*, et on peut s'attendre à ce que le nombre de chemins
2182 de longueur supérieure entre deux nœuds arbitraires dans le *cluster* soit grand. En particulier,
2183 ce nombre devrait être grand, relativement aux paires de nœuds appartenant à des *clusters*
2184 naturels différents. Vu sous un autre angle, une marche aléatoire dans le graphe va peu
2185 fréquemment aller d'un *cluster* naturel à un autre. L'intuition derrière l'algorithme correspond
2186 donc à ce que va faire un être humain pour repérer des regroupements de nœuds : trouver les
2187 « gros » blocs de nœuds fortement connectés, groupes qui sont eux-mêmes peu liés entre eux.

2188
2189 L'algorithme MCL trouve la structure en *cluster* d'un graphe grâce à une procédure
2190 mathématique de *bootstrapping*. Le processus calcule de manière déterministe les
2191 probabilités des marches aléatoires dans le graphe, et utilise deux opérateurs transformant
2192 un ensemble de probabilités dans un autre. Cela est rendu possible par l'utilisation du
2193 langage des matrices stochastiques (aussi appelées matrices de Markov), qui formalise le
2194 concept mathématique de marches aléatoires dans un graphe. Informellement, une matrice
2195 stochastique donne les probabilités de transition entre deux nœuds d'un graphe.

2196
2197 L'algorithme MCL simule des marches aléatoires à l'intérieur d'un graphe en alternant
2198 deux opérations appelées expansion et inflation. L'expansion consiste à prendre la puissance
2199 d'une matrice stochastique en utilisant le produit matriciel usuel (c'est-à-dire la mise au carré
2200 d'une matrice). L'inflation consiste à prendre la puissance d'Hadamard d'une matrice, suivie
2201 d'une étape de mise à l'échelle, de telle sorte que la matrice résultante soit à nouveau
2202 stochastique, c'est-à-dire que les éléments de la matrice (sur chaque colonne) correspondent
2203 à des valeurs de probabilités. Une matrice colonne stochastique est une matrice non-négative
2204 avec la propriété que chacune de ses colonnes ait une somme égale à 1. Étant donné une
2205 telle matrice M , et un nombre réel $r > 1$, la matrice colonne stochastique résultant de
2206 l'inflation de chacune des colonnes de M avec le coefficient de puissance r est écrit $G_r(M)$,
2207 et G_r est appelé l'opérateur d'inflation de coefficient de la puissance r . En écrivant $S_{r,j}(M)$
2208 pour la sommation de toutes les entrées de la colonne j de M pris à la puissance r (la

2209 somme est effectué après la mise à la puissance), $G_r(M)$ est défini pour chaque élément par
2210 $G_r(M_{ij}) = M_{ijr}/S_{r,j}(M)$ Chaque colonne j de la matrice stochastique M correspond à
2211 un nœud j du graphe stochastique associé à M . L'élément de la ligne i dans la colonne j
2212 (c'est-à-dire l'élément M_{ij}) correspond à la probabilité d'aller du nœud j au nœud i . On peut
2213 observer que pour des valeurs de $r > 1$, l'inflation change les probabilités associées avec la
2214 collection de marches aléatoires partant d'un nœud particulier (correspondant à la colonne
2215 d'une matrice) en favorisant les marches les plus probables par rapport aux marches moins
2216 probables. L'expansion correspond à calculer des marches aléatoires de longueur supérieure,
2217 ce qui signifie des marches aléatoires avec beaucoup d'étapes. Il associe des nouvelles
2218 probabilités à toutes les paires de nœuds, où un nœud est le point de départ et l'autre est la
2219 destination. Puisque les chemins de longueur supérieure sont plus courants à l'intérieur d'un
2220 *cluster* qu'entre des *clusters* différents, les probabilités associées avec les paires de nœuds
2221 appartenant au même *cluster* vont, en général, être relativement grandes car il y a beaucoup
2222 de chemins allant d'un nœud à l'autre. L'inflation va donc avoir pour effet d'augmenter les
2223 probabilités de marches intra-cluster et va diminuer les marches inter-cluster. Ceci est obtenu
2224 sans connaissance à priori de la structure des *clusters*. C'est simplement le résultat de la
2225 présence d'une structure en *cluster*. Finalement, itérer les expansions et inflations résulte en
2226 la séparation du graphe en plusieurs segments. Il y a plus de chemins entre ces segments,
2227 et la collection des segments résultants est simplement interprétée comme un *clustering*.
2228 L'opérateur d'inflation peut être altéré en utilisant le paramètre r . Augmenter ce paramètre
2229 a pour effet de rendre l'opérateur d'inflation plus fort, et cela augmente la granularité des
2230 *clusters*.

2231
2232 Informellement, exprimé dans le langage des flux stochastiques, on peut voir que
2233 l'expansion provoque la dissipation du flux à l'intérieur des *clusters* alors que l'inflation
2234 élimine le flux entre les différents *clusters*. L'expansion et l'inflation représentent les
2235 différentes forces qui s'alternent jusqu'à ce qu'un état d'équilibre soit atteint. Un état
2236 d'équilibre prend la forme d'une matrice doublement idempotente, c'est-à-dire d'une matrice
2237 qui ne change pas avec d'autres applications des opérateurs d'expansion ou d'inflation. Le
2238 graphe associé à une telle matrice consiste en différentes composantes connexes orientées.
2239 Chaque composante est interprétée comme un *cluster*, et a une forme d'étoile, avec un
2240 attracteur au centre, et des arcs allant de tous les nœuds de cette composante vers
2241 l'attracteur. En théorie, des systèmes attracteurs possédant plus d'un nœud attracteur
2242 peuvent apparaître, mais cela ne change pas l'interprétation en *clusters*. De plus, il peut
2243 exister des nœuds qui sont connectés à différentes étoiles, ce qui est canoniquement
2244 interprété comme des superpositions de *clusters*, ou en d'autres termes, certains nœuds
2245 peuvent appartenir à plusieurs *clusters*. Cette propriété de la méthode est d'ailleurs sans
2246 doute un avantage dans le cadre de la cartographie d'un système d'information, car on peut
2247 imaginer l'existence de nœuds qui sont connectés à de nombreux *clusters*, mais qui ne

2248 doivent pas appartenir à un seul de ces *clusters*. Pour ce qui est de la convergence, il peut être
2249 prouvé que le processus simulé par l'algorithme converge en un temps quadratique vers l'état
2250 d'équilibre. En pratique, l'algorithme commence à converger de manière visible après peu
2251 d'itérations (environ une dizaine). La convergence globale est quelque chose de très difficile à
2252 prouver, mais on peut conjecturer que le processus converge toujours si le graphe d'entrée
2253 est symétrique. A noter qu'il n'y a pas de garantie mathématique à l'heure actuelle pour la
2254 convergence. Une chose très importante à propos de l'algorithme est le fait qu'il récupère la
2255 structure en *clusters* via la trace laissée par cette structure sur le processus du flux.

2256
2257 D'autres avantages de cet algorithme sont :

- 2258 – il n'est pas mis en faute par des arcs liant des *clusters* différents ;
- 2259 – il est très rapide et supporte bien les fortes charges (plusieurs centaines de milliers de
2260 nœuds, la limite étant la multiplication de matrices) ;
- 2261 – il a un paramètre naturel influençant la granularité des *clusters* ;
- 2262 – les mathématiques associées à l'algorithme montrent qu'il y a une relation intrinsèque
2263 entre le processus simulé et la structure en *clusters* du graphe ;
- 2264 – sa formulation est simple et élégante ;

2265
2266 D'après la définition de l'algorithme MCL, on peut voir qu'il est basé sur un paradigme
2267 différent de tous les autres algorithmes basés sur des systèmes de liens. En effet, son
2268 fonctionnement est probabiliste, et sa terminaison basée sur une notion de convergence
2269 d'un processus aléatoire.

2270 Dans notre cas, les nœuds du graphe sont les images dans lesquelles on a détecté des instances, et
2271 on ajoute une arête pour chaque appariement entre deux instances.

2272 7.3.2 Protocole d'évaluation

2273 Le regroupement des groupes d'instances donnés par RANSAS a pour objectif principal de faciliter le
2274 travail de l'utilisateur. Ce travail consiste à fusionner des groupes entre eux (parce qu'ils représentent le
2275 même objet visuel), ou à nettoyer le groupe (extraire les instances qui ne correspondent pas à l'objet
2276 visuel identifié). Il est donc préférable que le nombre de groupes (*clusters*) soit relativement faible
2277 (quelques centaines de préférence, mais pas moins que le nombre d'objets visuels attendus, 37 en
2278 l'occurrence), et surtout que les groupes soient les plus purs possibles.

2279 Le seul paramètre réglable dans MCL-edge [167] est l'inflation. Afin de mesurer l'effet de ce paramètre
2280 sur la pureté et le nombre des *clusters* produits, nous proposons de mesurer l'évolution de la pureté
2281 moyenne (AvgPurity) en fonction du nombre de *clusters* lorsqu'on fait varier l'inflation dans l'intervalle
2282 conseillé par Van Dongen, c'est-à-dire entre 1.01 et 6.

2283 La pureté moyenne (AvgPurity) est définie comme étant la moyenne de la pureté de chaque *cluster*,

2284 telle que définie dans [170] par :

$$Purity(\mathbf{C}_j) = \frac{1}{|\mathbf{C}_j|} \max_{k=1,\dots,c} |\mathbf{C}_{j,k}| \quad (7.1)$$

2285 où $|\mathbf{C}_j|$ est la taille du $j^{\text{ème}}$ *cluster*, et $|\mathbf{C}_{j,k}|$ la taille du sous groupe de \mathbf{C}_j dont les instances
2286 appartiennent au $k^{\text{ème}}$ des c objets de la vérité terrain ($c = 37$).

2287 7.3.3 Évaluation

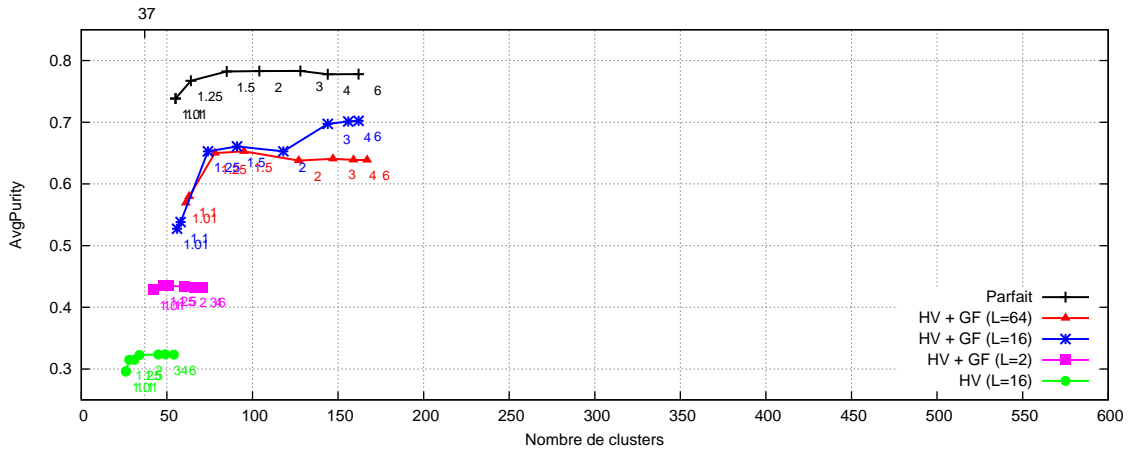
2288 La figure 7.5 page ci-contre montre la pureté moyenne des *clusters* en fonction du nombre de *clusters*
2289 produits par MCL. On peut lire la valeur d'inflation sur les courbes. La figure 7.5 page suivante présente
2290 trois graphiques, afin de présenter les résultats sur des graphes construits avec 1 000, 10 000 ou 100 000
2291 itérations de RANSAS. Il est ici nécessaire de rappeler que plus nombreuses sont les itérations et plus le
2292 graphe d'appariement est complet. Chaque graphique présente une courbe par score de vraisemblance
2293 utilisé dans RANSAS. On compare ici le score « parfait » (cf. section 4.3 page 62) et les scores calculés à
2294 partir de Hachage Visuel (HV), complétés ou non par le vote sur la Géométrie Faible (GF).

2295 On constate que les performances globales des différents scores de vraisemblance convergent
2296 lorsqu'on utilise le graphe d'appariement construit avec 100 000 itérations de RANSAS. Comme on l'a dit
2297 précédemment dans la section 7.2.2 page 99, on est dans ce cas limité par les performances du système
2298 de recherche d'objets, et le score de vraisemblance n'a que peu d'influence.

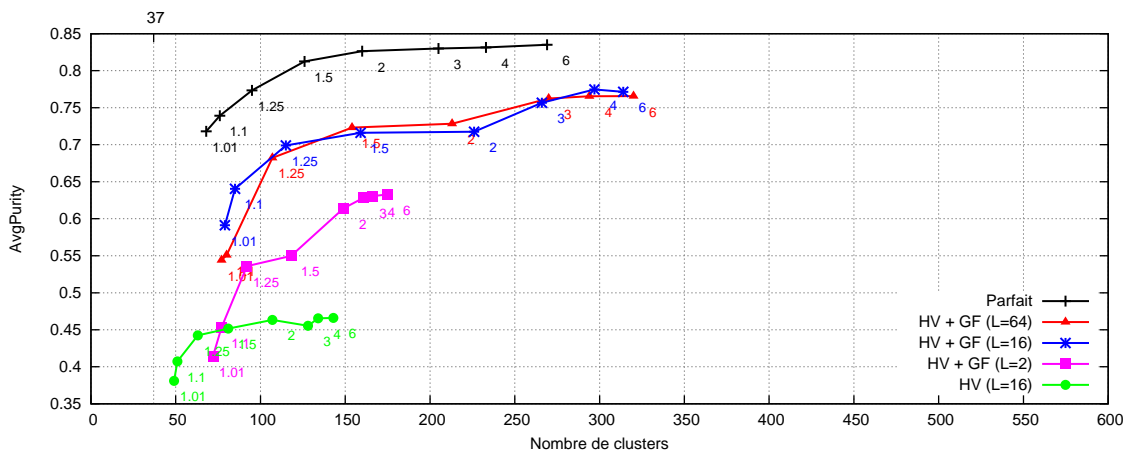
2299 On observe par contre sans surprise que pour seulement 1 000 itérations de RANSAS, on obtient déjà
2300 de bonnes performances avec nos meilleurs scores de vraisemblance (HV + GF et $L = \{16; 64\}$) : environ
2301 70 *clusters* (deux fois plus que dans la vérité terrain) pour une pureté moyenne de 0.65, tandis que les
2302 autres scores ne parviennent pas à obtenir plus de 0.45.

2303

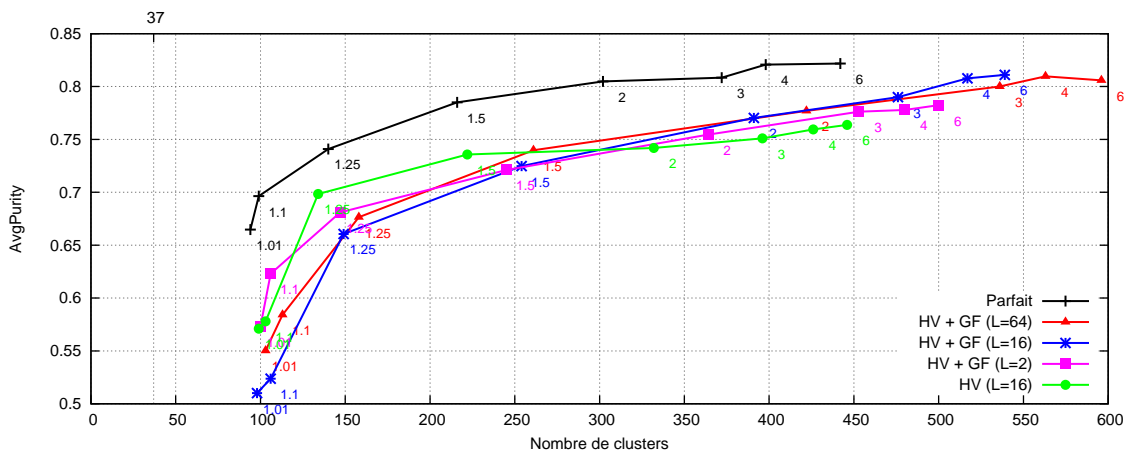
2304 Le temps de calcul mis par MCL-edge pour regrouper les instances est totalement négligeable
2305 comparé au reste du processus (description, indexation, RANSAS). Le regroupement est en effet effectué
2306 en moins d'une dizaine de secondes, même avec une faible valeur d'inflation (1.01) qui nécessite plus
2307 d'itérations pour converger. D'après Van Dongen, MCL-edge peut supporter des millions de nœuds et
2308 des centaines de millions d'arêtes, ce qui est une taille globalement similaire à celles des graphes
2309 d'appariement construits sur des collections multimédias traitées par les systèmes état de l'art de
2310 recherche d'objets visuels.



(a) 1K itérations



(b) 10K itérations



(c) 100K itérations

FIGURE 7.5 – Pureté moyenne (AvgPurity) en fonction du nombre de *clusters* obtenus en faisant varier le paramètre d'inflation de MCL entre 1.01 et 6, pour différents scores de vraisemblance, et différents nombre d'itérations de RANSAS (1K, 10K et 100K).

2311 Chapitre 8

2312 Découverte d'événements saillants dans 2313 des médias d'actualité

2314 Dans ce chapitre, nous proposons une expérimentation dans une grande collection de vidéos issues
2315 de la télévision. Cette expérimentation a pour objectif de montrer l'intérêt de l'approche transmédia dans
2316 la découverte d'évènements (médiatiques) saillants, et également d'évaluer les performances de notre
2317 méthode de découverte d'objets visuels appliquée à plus de 2 000 heures de vidéos. Nous proposons
2318 également une autre approche transmédia appliquée sur un corpus hétérogène constitué à partir de
2319 nombreuses sources de type différent.

2320 Un évènement médiatique a plusieurs définitions possibles, selon que l'on se place du point de vue du
2321 sociologue, du journaliste, ou de l'informaticien.

2322
2323 Selon Marie-Luce Viaud (coordinatrice du projet ANR OTMedia¹ No. 2010CORD015) :

2324 Un évènement médiatique en tant qu'entité [objet visuel] détectable est constitué d'énoncés
2325 [visuels] qui relatent une même « occurrence du réel » et son importance médiatique est liée
2326 au nombre des énoncés [visuels] qui le constituent.

2327 Parmi les évènements médiatiques français de l'année 2012, on trouve par exemple :

- 2328 – Mort du journaliste Gilles Jacquier à Homs en Syrie ;
- 2329 – Cinq Oscars pour le film « The Artist » ;
- 2330 – Débat de l'entre-deux-tours aux élections présidentielles ;
- 2331 – ArcelorMittal annonce la fermeture définitive des hauts fourneaux de Florange ;

2332 8.1 Détection d'évènements télévisuels

2333 8.1.1 Création d'une grande collection de journaux télévisés

2334 Afin d'évaluer la capacité de notre méthode à traiter une grande collection multimédia, nous
2335 avons construit une nouvelle base composée de 1 809 080 images-clés, extraites de 2 051 heures de

1. www.otmedia.fr

2336 vidéos. Nous l'avons nommée « FrenchTVFrames » dans notre dernière publication [171]. L'extraction des
 2337 images-clés est réalisée de manière à ce que l'on garde au moins une image pour chaque plan, voire
 2338 plusieurs si le contenu du plan est très variable (si la caméra bouge par exemple). On extrait donc bien
 2339 plus d'images dans les plans des reportages que dans ceux sur le plateau du journal télévisé. En moyenne,
 2340 on garde environ une image toutes les quatre secondes.

2341 Les vidéos ont été choisies parmi sept chaînes de télévision Française : TF1, France 2, Euronews,
 2342 BFMTV, France 24, i>TELE et LCI (cf. figure 8.1), entre le 10 juin 2011 et le 28 mars 2012, soit une période
 2343 d'environ neuf mois. Seule une heure par jour a été conservée pour chaque chaîne, correspondant à celle
 2344 proposant le plus de journaux télévisés dans la journée.

2345 Nous avons décrit cette base avec environ 549 millions de points d'intérêt SIFT.



FIGURE 8.1 – Chaînes de télévision incluses dans la collection FrenchTVFrames

2346 8.1.2 Approche transmédia pour la découverte d'objets informatifs

2347 La découverte d'éléments fréquents dans les vidéos issues de la télévision pose un problème bien
 2348 connu [172, 173] : les principales répétitions sont les publicités, les génériques d'émission, les décors de
 2349 plateaux, etc. Hors, ceci n'est pas l'objectif de notre expérimentation, qui vise à obtenir des objets visuels
 2350 du type « évènement médiatique ».

2351

2352 Afin de défavoriser les évènements peu informatifs, et de mettre en avant les évènements
 2353 médiatiques saillants, nous avons émis deux postulats :

- 2354 – Premièrement, un évènement médiatique est généralement relayé par la plupart des sources
 2355 d'information (la plupart des chaînes de télévision en l'occurrence), contrairement aux génériques
 2356 d'émission et aux décors de plateaux qui sont presque exclusivement « mono-source ». Quant aux
 2357 publicités, nous avons pu vérifier qu'elles n'apparaissent pas sur un grand nombre de chaînes ;
- 2358 – Deuxièmement, un évènement médiatique est condensé dans le temps. Il est traité par les médias
 2359 pendant quelques jours, avant de disparaître, tandis que les publicités sont diffusées pendant au
 2360 moins plusieurs semaines, et les génériques et les décors d'émissions pendant plusieurs mois ou
 2361 années ;

2362

2363 La stratégie que nous avons mise en place repose donc sur un classement des objets découverts.

2364 Concrètement, chaque objet détecté s'est vu affecté un score S tel que :

$$S = \frac{\#\{C\} \cdot \hat{f}}{f_d} \quad (8.1)$$

2365 où $\#\{C\}$ est le nombre de chaînes (uniques) dans lesquelles on a détecté une instance de l'objet
 2366 considéré, \hat{f} est l'estimation de la fréquence f de l'objet (nombre de détections) et f_d est la fréquence

2367 temporelle (nombre de jours différents parmi les instances de l'objet découvert).

2368 Ainsi, nous privilégions les objets possédant le plus d'instances et ayant été découverts parmi un
2369 maximum de chaînes, tout en minimisant le nombre de jours de diffusion.

2370

2371 La figure 8.1 page suivante donne des illustrations des meilleurs évènements découverts, par
2372 catégorie de surface (petit, moyen ou gros objet) dans les 2 051 heures de télévision française, selon le
2373 classement obtenu grâce à l'équation 8.1 page ci-contre.

2374 On constate que chaque objet visuel découvert est bien l'un des évènements médiatiques majeurs de
2375 la période couverte par notre collection de vidéos.

2376 Le calcul du score de vraisemblance a été effectué avec $L = 8$ tables de hachage visuel et ajout de
2377 contraintes de géométrie faible, et la recherche d'objets visuels a été paramétrée avec $\alpha = 0.90$.

2378 Cette expérimentation a été effectuée avec un serveur équipé de deux processeurs hexa-cores (Intel
2379 X5660). L'étape de hachage visuel a duré environ 4 heures, le calcul du score de vraisemblance a pris 17
2380 heures, et utilisé 70GB de RAM. RANSAS a ensuite effectué 50 000 itérations en 18 heures. Au total,
2381 le processus complet aura donc mis 39 heures pour découvrir des objets dans plus de 1.8 millions d'images.

2382

2383 8.2 Découverte d'évènements transmédias

2384 Dans cette section, nous présentons le travail [174] proposé au Grand Challenge d'ACM Multimedia
2385 2012.

2386 Ce travail s'inscrit dans le cadre du projet ANR OTMedia² No. 2010CORD015), dont l'objectif est de
2387 collecter et d'analyser les documents multimédias issus de nombreuses sources réparties sur plusieurs
2388 types de médias d'information.

2389 Concrètement, quatre catégories de médias ont été collectés et indexés par notre système :

2390 1. **Agences de presse** : La totalité des flux multimédias venant de l'AFP a été collectée. L'AFP est la
2391 plus ancienne agence de presse dans le monde et une des trois plus importantes actuellement.
2392 Chacune des dépêches multimédias inclut au minimum une image, un texte, et un ensemble de
2393 métadonnées ;

2394 2. **Sites web d'information** : Environ 1 500 sites internet français ont été collectés à partir de leurs flux
2395 RSS. Les sites qui ont été sélectionnés sont des pure players (12), des sites de radios (14), de TVs
2396 (17), de presse (71), des blogs (339), ainsi que les sites des personnalités politiques comme les
2397 sénateurs, députés, candidats à la présidentielle (646), des partis politiques (263), des syndicats
2398 (19) et des institutions (62) ;

2399 3. **Journaux d'informations** : 7 journaux nationaux ont été collectés numériquement (La Croix,
2400 Liberation, Le Monde, 20 Minutes, Direct Matin, les Echos, Le Canard Enchaîné) ;

2. www.otmedia.fr




<p>Sommet du G20 <i>Novembre 2011</i> Petit objet < 33% de la surface de l'image</p> $\left. \begin{array}{l} \#\{C\} = 6 \\ \hat{f} = 15 \\ f_d = 2 \end{array} \right\} S = 45$	<p>Décès de Kim Jong-il <i>Décembre 2011</i> Objet moyen < 66% de la surface de l'image</p> $\left. \begin{array}{l} \#\{C\} = 6 \\ \hat{f} = 8 \\ f_d = 1 \end{array} \right\} S = 48$	<p>The Artist <i>Février 2012</i> Gros objet > 66% de la surface de l'image</p> $\left. \begin{array}{l} \#\{C\} = 7 \\ \hat{f} = 39 \\ f_d = 8 \end{array} \right\} S = 34$
		

TABLE 8.1 – Meilleurs évènements découverts (plus gros scores S), par catégorie (petit, moyen, gros), dans les 2 051 heures de télévision Française.

2401 4. **Journaux télévisés** : Les informations et reportages de 10 chaînes de télévision ont été obtenues, à
 2402 partir de 5 chaînes publiques, 3 chaînes généralistes et 2 chaînes d'information ;

2403

2404 L'objectif que nous recherchons ici est la découverte d'évènements médiatiques transmédias. Tout
 2405 comme dans dans les vidéos issues de la télévision, les images et vidéos du web et de la presse
 2406 présentent de très nombreux objets visuels très fréquents et non informatifs, tel que des publicités, des

2407 bannières, et autres éléments graphiques divers. De même que dans la section précédente, nous partons
2408 donc du principe que les évènements médiatiques importants sont relayés par la plupart des médias
2409 d'information, et que nous pouvons les mettre en évidence par une approche transmédia, c'est-à-dire en
2410 ne considérant que les appariements visuels entre des médias de types différents.

2411 Une image issue d'une agence de presse, et relayée par plusieurs chaînes de télévision, journaux
2412 et sites web, est ainsi sujette à représenter un évènement médiatique saillant, qu'une image très
2413 fréquemment répétée par un seul média.

2414

2415 Afin d'identifier ces évènements médiatiques transmédias, nous proposons de réaliser une
2416 découverte d'objets visuels avec RANSAS, puis un *clustering* MCL, pour chaque semaine calendaire de
2417 la collection constituée. En pratique, nous construisons donc un index pour chaque semaine, dans
2418 lequel nous effectuons la découverte. Les résultats de la découverte sont filtrés afin de ne conserver
2419 que les appariements transmédias. Le *clustering* MCL est ensuite effectué pour rassembler les groupes
2420 d'instances découverts. Pour finir, on trie les *clusters* selon un score de diversité calculé à partir du nombre
2421 de supports détectés. Plus concrètement, nous avons tout d'abord affecté un poids à chacun des médias
2422 collectés. Ceci a été réalisé en pondérant de façon identique les quatre catégories de médias, c'est-à-dire
2423 0,25. Pour une catégorie de médias donnée, le poids est ensuite distribué uniformément entre tous les
2424 flux lui correspondant. Le poids du flux AFP est ainsi égal à 0,25, tandis que chacune des dix chaînes de
2425 télévision a un poids égal à un dixième de 0,25, soit 0,025. A partir de cette pondération, nous calculons
2426 le score de diversité d'un *cluster* en sommant les poids des médias détectés à l'intérieur de celui-ci.

2427 La figure 8.2 page suivante montre un *cluster* filtré par le tri sur notre score de diversité des médias,
2428 et trois *clusters* correspondant aux meilleurs scores (meilleurs évènements transmédias) de la première
2429 semaine de septembre 2011.

2430 La figure 8.3 page 117 présente une interface de navigation dans les meilleurs évènements
2431 transmédias détectés semaine par semaine.

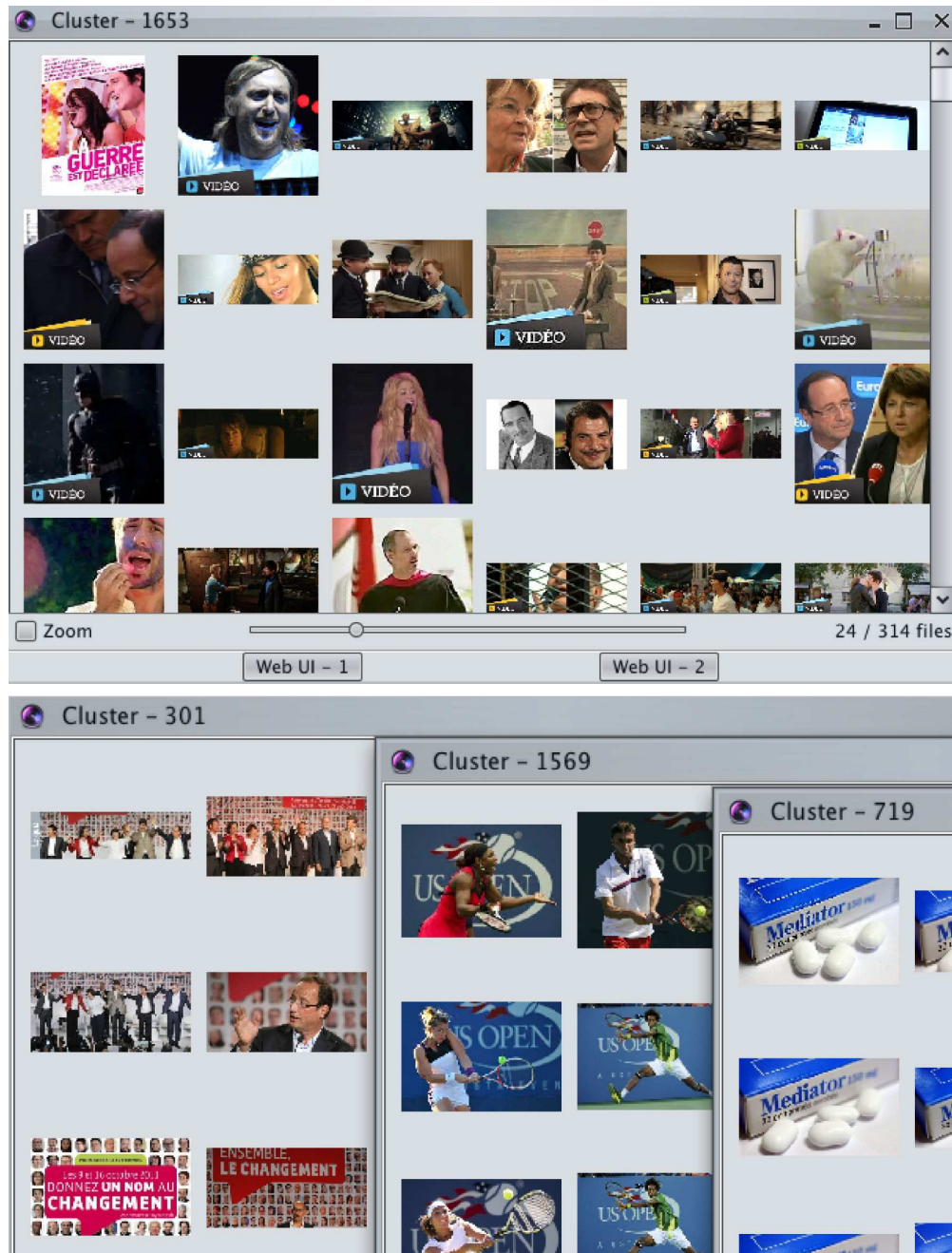


FIGURE 8.2 – Un *cluster* filtré par le tri sur le score de diversité (en haut), et les trois meilleurs *clusters* (événements transmédiés) de la première semaine de septembre 2011, représentant les primaires socialistes, l'US Open, et le scandale du Médiator.

Chapitre 9

Suggestion de requêtes visuelles

Nous avons vu dans la section 2.7 page 43 que les systèmes de recherche d'objets visuels sont aujourd'hui capables d'atteindre des performances appréciables, y compris pour la recherche de petits objets comme des logos.

Du point de vue de l'utilisateur, cette recherche d'objets se limite généralement à la création d'une requête locale (fenêtre autour de l'objet), puisque le système se charge ensuite automatiquement d'effectuer la recherche. Cette approche souffre toutefois de deux problèmes liés à la perception de l'utilisateur :

- Premièrement, quand aucune (ou très peu) autre instance de l'objet recherché n'existe dans la collection d'images, le système retourne principalement des faux positifs, et l'utilisateur est déçu par les résultats. En effet, celui-ci est rarement en mesure de déterminer à l'avance si l'objet est fréquent ou non ;
- Deuxièmement, quand l'utilisateur sélectionne un objet déformable, ou très altéré par les conditions de prises de vue, une mauvaise résolution..., il peut également être déçu des mauvais résultats, n'ayant pas conscience des capacités limites du système utilisé. Les utilisateurs sont ainsi fréquemment tentés de rechercher des visages, des fleurs, etc. ;

Une solution envisageable pour limiter la déception de l'utilisateur est de diminuer le taux de fausses-alarmes, en utilisant par exemple une méthode de seuillage adaptatif comme la méthode *contrario* de [4]. L'utilisateur est toutefois toujours déçu si le système ne lui retourne aucun résultat.

Nous proposons dans ce chapitre une solution alternative, qui consiste à suggérer directement à l'utilisateur, les requêtes retournant des résultats pertinents.

Une première approche de suggestion de requêtes visuelles avait déjà été proposée par Zha et al. dans [175]. Leur méthode est une extension des méthodes de suggestion de requêtes textuelles, qui sont maintenant fréquemment utilisées par les moteurs de recherche. L'argument des auteurs était que les suggestions basées sur le texte ne suffisent pas toujours à prédire précisément la requête de

2461 l'utilisateur. En ajoutant un ensemble d'images représentatives aux suggestions textuelles, l'utilisateur
 2462 peut alors spécifier plus précisément son intention. Leur méthode est principalement basée sur un calcul
 2463 de similarités visuelles globales, en complément de méthodes de recherche textuelle.

2464 Notre méthode diffère en deux points principaux :

- 2465 – Nos suggestions sont purement visuelles (bien que l'on pourrait employer les informations
 2466 textuelles pour affiner nos suggestions) ;
- 2467 – Les suggestions représentent des instances d'objet, et non pas des images entières ou des concepts
 2468 visuels (catégories), et sont de ce fait plus précises ;

2469 9.1 Principe de la suggestion de requêtes visuelles

2470 Dans [176], nous avons proposé, en collaboration avec Amel Hamzaoui, de suggérer des requêtes
 2471 uniquement si elles contiennent des résultats dans la collection d'images. Ceci requiert d'abord de
 2472 découvrir des *clusters* d'objets dans la collection, et de sélectionner ensuite les plus pertinents en
 2473 fonction du nombre d'occurrences et de l'intention de l'utilisateur. Le *clustering* est effectué par un
 2474 nouvel algorithme de *clustering* par voisins-partagés (*shared-neighbours*) de graphe biparti, utilisé pour
 2475 rassembler les instances d'objets découvertes par RANSAS. L'algorithme MCL utilisé dans le chapitre
 2476 précédent n'est en effet pas directement applicable lorsque les images peuvent contenir plusieurs
 2477 instances de plusieurs objets visuels.

2478
 2479 Plus concrètement, le concept que nous introduisons dans ce chapitre est le suivant : plutôt que de
 2480 laisser l'utilisateur sélectionner une région à rechercher, nous proposons que le système se charge de lui
 2481 suggérer automatiquement les régions contenant des instances d'objets $\{c, f\}$ -fréquents. Lorsque un
 2482 utilisateur clique sur une région suggérée, le système retourne les images contenant les autres instances
 2483 de l'objet appartenant au *cluster* découvert par notre méthode. Du point de vue utilisateur, ce concept de
 2484 suggestion est très différent de la méthode standard de sélection libre d'une requête. On peut ainsi
 2485 surnommer notre concept de suggestion d'objet visuel par l'expression « lien visuel » (ou même hyperlien
 2486 visuel, par analogie avec les hyperliens textuels).

2487
 2488 Cette approche a également un intérêt pour le système lui-même qui est moins surchargé par
 2489 des requêtes « inutiles », et qui peut répondre beaucoup plus rapidement puisque les résultats sont
 2490 pré-calculés, et stockés en mémoire ou sur disque.

2491
 2492 Ces hyperliens visuels peuvent être employés dans de nombreux scénarios. Nous nous focaliserons
 2493 dans la section 9.3 page 123 sur deux d'entre eux : la « suggestion d'objets visuels au survol de la souris »,
 2494 et la « suggestion d'objets visuels comme facétisation d'une requête textuelle ».

2495 9.2 Algorithme de *clustering* de graphe biparti

2496 Comme le montre la figure 9.1 page suivante, certaines images peuvent présenter plusieurs instances
 2497 différentes n'appartenant pas au même objet. L'utilisation d'un algorithme de *clustering* de graphe dont
 2498 les nœuds sont les images n'est donc pas adapté, il est ainsi préférable que les nœuds soient les instances
 2499 découvertes. Ceci justifie donc l'emploi d'un algorithme de *clustering* de graphe biparti, comme celui
 2500 proposé par Hamzaoui [177].

2501 Bien que les germes (mots visuels géométriquement cohérents) découverts par RANSAS
 2502 correspondent bien à des motifs visuels fréquents dans la collection, ils ne peuvent pas toujours être
 2503 considérés comme des objets complets pour plusieurs raisons :

- 2504 – Par construction, un germe ne couvre (spatialement) qu'une sous-partie d'un objet ;
- 2505 – Le rappel n'étant pas parfait, un germe correspond à un sous-ensemble des instances de l'objet
 2506 découvert ;
- 2507 – Plus un objet est fréquent dans la collection, plus on découvre de germes ;

2508
 2509 La construction précise et complète d'un modèle d'objet nécessite de regrouper tous les germes
 2510 appartenant à un même objet. Ceci ne peut être effectué à partir du contenu visuel des germes, étant
 2511 donné que deux germes avec des contenus différents peuvent très bien être deux sous-parties d'un
 2512 même objet. Une alternative à cela est de regrouper les germes fortement corrélés entre eux (c'est-à-dire
 2513 apparaissant des les mêmes images), ce qui peut être formulé comme un problème de *clustering* biparti.
 2514 La figure 9.1 page suivante illustre la méthode proposée pour regrouper les germes représentant le même
 2515 objet.

2516 Soit $G = (X; E) = (I, S; E)$ le graphe biparti résultant de la découverte d'objets visuels, avec :

$$I = \{I_i\}_{i \in [1, N_I]}$$

2517 l'ensemble des vertex représentant les images de la collection, et :

$$S = \{S_j\}_{j \in [1, |S|]}$$

2518 l'ensemble des germes (ou mots visuels géométriquement cohérents) découverts par RANSAS,
 2519 $X = I \cup S$ et $I \cap S = \emptyset$

2520 Chaque arête orientée $e_{i,j} \in E$ a un point de départ dans S et un point d'arrivée dans I , et un poids
 2521 $w_{i,j}$ correspondant au score d'appariement fourni par RANSAS ($w_{i,j} = 0$ signifie qu'aucune arête ne
 2522 connecte le germe S_j à l'image I_i).

2523 L'avantage de cette représentation bipartite est qu'elle permet de formuler notre objectif de *clustering*
 2524 de germes comme un problème de *co-clustering*. On vise en effet à trouver des *clusters* d'objet
 2525 $O_n = (S^n, I^n)$ avec $S^n \subset S$ étant le sous-ensemble des germes modélisant un objet donné, et $I^n \subset I$
 2526 étant le sous-ensemble des images contenant des instances d'objet. Un *cluster* d'objet idéal est celui
 2527 dont les germes apparaissent dans les mêmes images. Il est important de remarquer l'avantage sur les
 2528 méthodes de découverte d'objets créant des graphes d'appariement entre images [178, 179, 180, 148] :

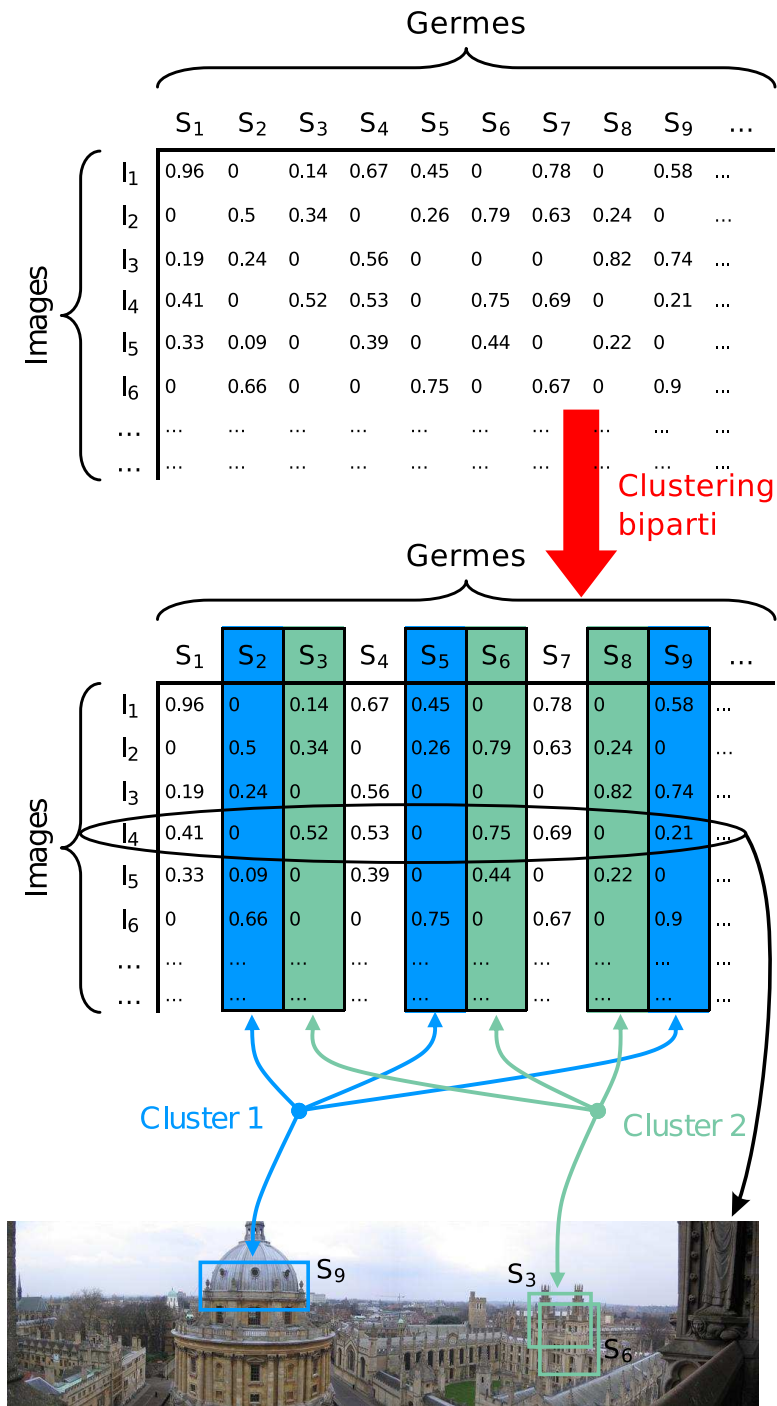


FIGURE 9.1 – Illustration de la méthode proposée pour suggérer des requêtes visuelles dans l'image I_4 . Les germes S_2, S_5 et S_9 appartiennent à un premier cluster (en bleu), tandis que S_3, S_6 et S_8 appartiennent à un second (en vert).

2529 une image donnée peut être affectée à plusieurs clusters d'objets (lorsqu'elle contient des instances de
 2530 différents objets). De plus, chaque cluster d'objet est composé d'un ensemble unique de germes associés

2531 aux régions appariées et localisées spatialement.

2532

2533 Résoudre le problème de *clustering* biparti n'est pas un problème trivial. Plusieurs travaux ont proposé
 2534 des techniques spectrales appliquées au *clustering* de documents textuels [181, 182]. Ces méthodes sont
 2535 utiles pour partitionner des graphes bipartis en un nombre prédéfini de *clusters* équilibrés, mais ne sont
 2536 pas adaptées à notre problème. Le nombre d'objets à découvrir, tout comme le nombre de germes à
 2537 regrouper peut en effet être très variable. Les résultats sont de plus très sensibles aux paramètres
 2538 utilisés. C'est pourquoi nous utiliserons donc l'algorithme de *clustering* de graphe biparti proposé par
 2539 Hamzaoui [177], et inspiré des méthodes de *clustering* par plus proches voisins partagés (*Shared Nearest*
 2540 *Neighbours*). Le principe de la méthode de Hamzaoui [177] est de regrouper les éléments non pas grâce à
 2541 leur similarité, mais grâce au degré de ressemblance de leurs voisinages respectifs.

2542 9.3 Scénarios de suggestion de requêtes visuelles

2543 Pour chacun des deux scénarios de suggestion de requêtes visuelles décrits plus haut, nous répondons
 2544 aux questions suivantes : Que suggère-t-on ? Comment affichons nous les suggestions ? Que doit-on
 2545 retourner lorsque l'utilisateur clique sur un objet suggéré ?

2546 *Suggestion d'objets visuels au survol de la souris :*

2547 Pour chaque image $I_j \in I$, on suggère des requêtes dont le nombre est égal au nombre de *clusters*
 2548 ayant I_j dans leur ensemble d'images duales. Chaque suggestion est représentée par une fenêtre
 2549 rectangulaire calculée à partir de l'ensemble de toutes les régions appartenant au *cluster* et à l'image
 2550 concernés. Cette fenêtre rectangulaire est obtenue en conservant les zones couvertes par au minimum
 2551 deux régions détectées, comme cela est illustré par la figure 9.2 page suivante. Ceci permet d'être plus
 2552 résistant aux *outliers* qui agrandissent par erreur les rectangles englobant des germes découverts. Dans
 2553 les zones non concernées par une suggestion, l'image est légèrement opacifiée afin de guider l'utilisateur
 2554 vers les zones contenant des instances d'objet découvertes. Au survol de la souris sur une suggestion, le
 2555 contour de la fenêtre est coloré en vert. Lorsqu'un utilisateur clique sur l'une des requêtes suggérées,
 2556 on retourne une liste d'images triées selon leur intersection avec l'objet sélectionné. L'intersection est
 2557 définie comme le nombre de germes correspondant à la fois au *cluster* et à l'image. Par exemple, dans
 2558 la figure 9.1 page ci-contre, l'intersection de l'image I_2 avec le *cluster* 2 est égale à 3, tandis que
 2559 l'intersection de l'image I_4 avec le même *cluster* est égale à 2. La figure 9.4 page 126 montre une capture
 2560 d'écran de ce scénario.

2561 *Suggestion d'objets visuels comme facétisation d'une requête textuelle :*

2562 Pour ce scénario, on suppose qu'un moteur de recherche textuel a retourné un sous-ensemble
 2563 d'images $I_x \subset I$. On sélectionne ensuite comme requêtes suggérées les M meilleurs *clusters* de la
 2564 base ayant la plus grande intersection entre les images (dans leur représentation duale) et la liste de



FIGURE 9.2 – Génération des fenêtres rectangulaires pour la suggestion d'objets visuels au survol de la souris

2565 résultats textuels (c'est-à-dire les *clusters* représentant les objets les plus fréquents dans la liste de
2566 résultats). Chaque requête suggérée est affichée au dessus de l'interface de recherche par une miniature
2567 représentative de l'objet. Cette miniature est construite en cherchant d'abord l'image ayant la plus grande
2568 intersection avec le *cluster*, c'est-à-dire en comptant le nombre de germes. Une fois la meilleure image
2569 sélectionnée, on recadre l'objet d'intérêt dans cette image avec la même procédure que celle décrite dans
2570 le scénario précédent. La figure 9.3 page ci-contre illustre ce processus de construction des miniatures
2571 sur trois *clusters* de la base Oxford Buildings. Lorsque l'utilisateur clique sur l'une de ces suggestions,
2572 on retourne, comme précédemment, une liste d'images triées selon leur intersection avec l'objet. La
2573 figure 9.5 page 127 montre une capture d'écran de ce scénario.

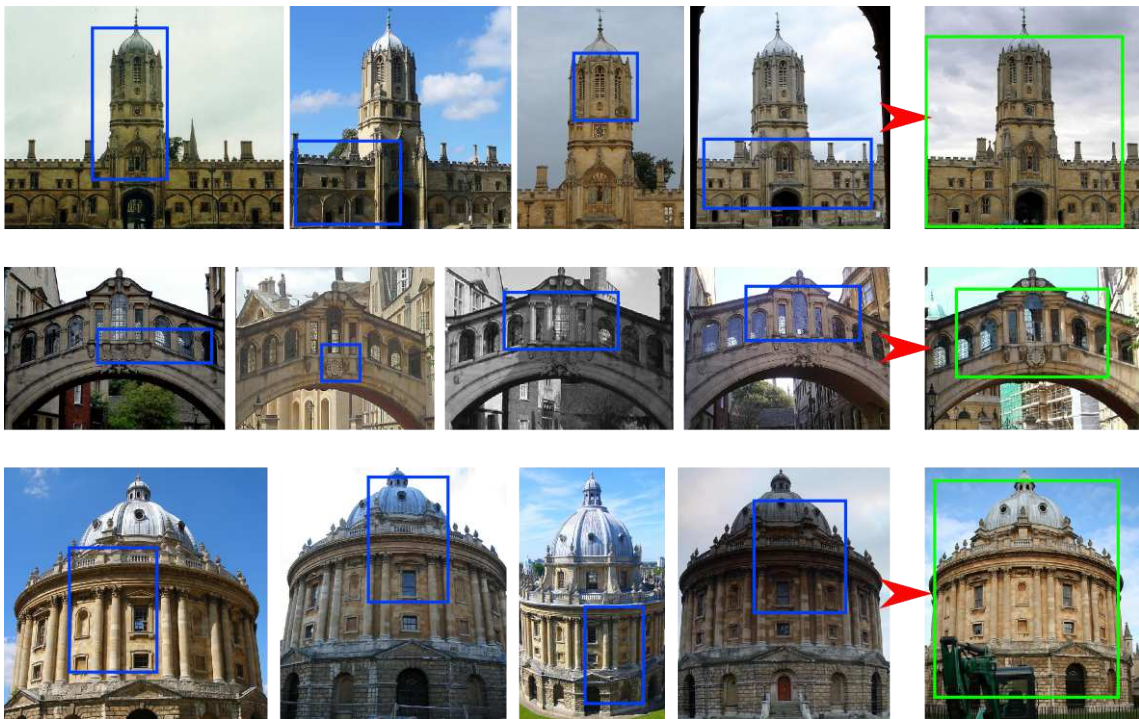


FIGURE 9.3 – Génération des miniatures par recadrage pour la suggestion d'objets visuels en complément d'une requête textuelle

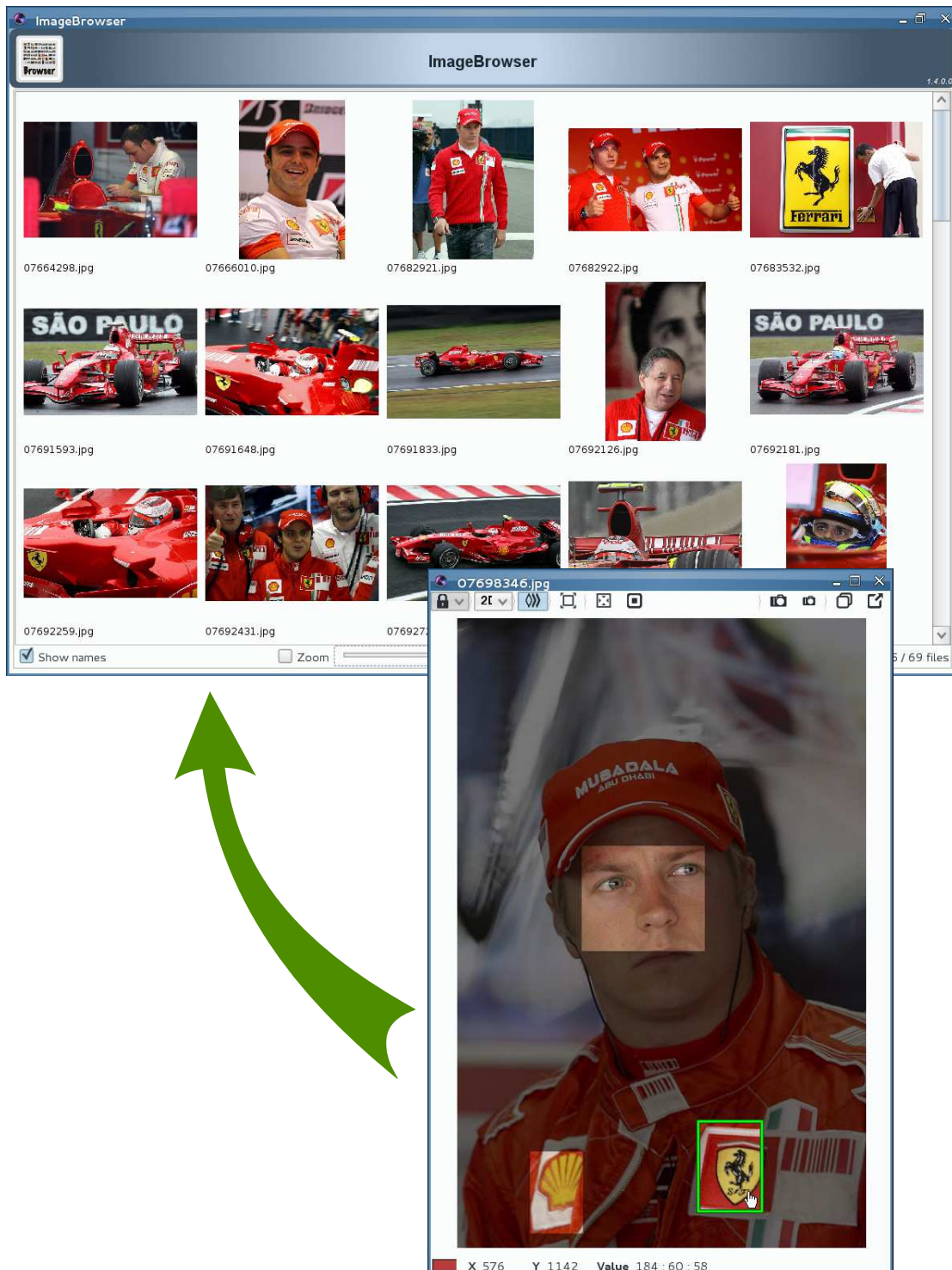


FIGURE 9.4 – Capture d'écran de l'interface de suggestion d'objets visuels au survol de la souris



Interface en construction

FIGURE 9.5 – Capture d'écran de l'interface de suggestion d'objets visuels comme facétisation d'une requête textuelle

2574

Chapitre 10

2575

Conclusion

2576

Les travaux de cette thèse traitent du problème de la découverte et de l'exploitation d'objets visuels fréquents dans des collections multimédias.

2578

2579

2580

Dans cette conclusion, nous commencerons par synthétiser les contributions. Nous dresserons ensuite un bilan des analyses menées à partir des expérimentations, et enfin nous réfléchirons aux perspectives envisageables.

2581

10.1 Synthèse des contributions

2582

2583

2584

2585

2586

La première contribution de cette thèse a été de fournir un formalisme aux problèmes de découverte et de fouille d'instances d'objets visuels fréquents. Ces problèmes n'avaient en effet jamais été clairement définis dans la littérature. Cette modélisation nous a permis entre autres choses de mettre en évidence le lien étroit qui existe entre la taille des objets à découvrir et la complexité du problème à traiter.

2587

2588

2589

2590

2591

2592

La deuxième contribution présentée dans cette thèse est l'algorithme RANSAS (*RAN*d*om* *S*ample *A*nd *S*earch), qui est une méthode générique de résolution des problèmes de fouille et découverte. RANSAS repose d'une part sur un processus itératif d'échantillonnage d'objets candidats et d'autre part sur une méthode efficace d'appariement d'objets rigides à large échelle. L'idée est de considérer l'étape de recherche d'instances proprement dite comme une simple boîte noire à laquelle il s'agit de soumettre des régions d'images ayant une probabilité élevée d'appartenir à un objet fréquent de la base.

2593

2594

2595

2596

2597

2598

2599

Une première approche étudiée dans la thèse consiste à simplement considérer que toutes les régions d'images de la base sont équiprobables, avec comme idée conductrice que les objets les plus instanciés sont ceux qui auront la couverture spatiale la plus grande et donc la probabilité la plus élevée d'être échantillonnés. En généralisant cette notion de couverture à celle plus générique de couverture probabiliste, il est alors possible de modéliser la complexité de notre méthode pour tout score de vraisemblance donné en entrée, et de montrer ainsi l'importance de cette étape.

2600

La troisième contribution principale de la thèse s'attache à construire un score de vraisemblance

2601 s'approchant au mieux de la distribution parfaite, tout en restant *scalable* et efficace. Cette dernière
2602 repose sur une approche originale de hachage à deux niveaux, permettant de générer efficacement un
2603 ensemble d'appariements visuels dans un premier temps, et d'évaluer ensuite leur pertinence en fonction
2604 de contraintes géométriques faibles.

2605

2606 Afin d'évaluer les problèmes de découverte et fouille d'objets $\{c, f\}$ -fréquents, nous avons proposé
2607 une nouvelle collection d'images disposant d'une vérité terrain adaptée.

2608

2609 Enfin, nous avons également présenté un nouveau concept de suggestion de requêtes visuelles
2610 permettant d'assurer à l'utilisateur que ses requêtes auront toujours des résultats, ainsi que d'améliorer
2611 les résultats d'une recherche d'images par le texte en suggérant des objets identifiés comme fréquents au
2612 sein de la collection.

2613 10.2 Analyse et bilan

2614 Cette thèse a été effectuée dans le cadre d'un contrat cifre¹ à l'Ina, et l'accent a donc été porté sur
2615 l'optimisation de nos algorithmes afin qu'ils trouvent leurs applications dans les très grands volumes
2616 d'images et de vidéos détenus par l'Ina. Nous avons effectué des expérimentations sur une collection de
2617 petite taille (FlickrBelgaLogos : 10 000 images) afin d'évaluer quantitativement nos performances, et de
2618 permettre la comparaison avec de futurs travaux sur le sujet. Mais nous avons également montré que
2619 notre système pouvait s'adapter à des collections de plus de 2 000 heures de télévision (sur une seule
2620 machine).

2621

2622 En ce qui concerne l'algorithme de découverte RANSAS, nous avons montré l'intérêt de la mise à jour
2623 de la fonction de probabilité de masse à chaque itération. Nous avons aussi évalué l'accélération (en
2624 termes d'objets et d'instances découverts en fonction du nombre d'itérations) obtenue avec nos scores
2625 de vraisemblance par rapport à l'échantillonnage uniforme. Nous avons ainsi montré que l'utilisation de
2626 scores de vraisemblance adaptés permettait de réduire le nombre d'itérations de RANSAS d'un facteur
2627 pouvant aller jusqu'à 32. Au total, il faut environ une quinzaine d'heures pour découvrir 90% des objets de
2628 la vérité terrain, et moins de neuf heures pour découvrir 50% des instances dans la base FlickrBelgaLogos.

2629

2630 Pour ce qui est du calcul du score de vraisemblance, nous avons observé que chacun des quatre
2631 attributs de géométrie faible apportait une information utile, et que l'utilisation de ces quatre attributs
2632 complétait efficacement les phases de hachage et de filtrage des descripteurs visuels. Nous avons
2633 constaté que le temps de calcul de ce score restait assez faible (moins d'un demi-heure pour 10 000
2634 images et 50 millions de descripteurs, avec 16 tables de hachage visuel). Les évaluations ont également
2635 mis en évidence que l'utilisation de plus de 16 tables de hachage visuel était peu pertinente (faible
2636 amélioration de la précision et du rappel pour un coût calculatoire important). En comparant notre

1. http://www.anrt.asso.fr/fr/espace_cifre/accueil.jsp

2637 approche à celle basée sur la méthode Geometric min-Hashing, considérée comme l'actuel état de l'art,
2638 nous avons montré que notre méthode était en mesure d'atteindre de bien meilleures performances.

2639

2640 10.3 Perspectives

2641 Il serait intéressant d'étudier la taille et la forme des requêtes issues de la phase d'échantillonnage
2642 de RANSAS. Plus précisément, il serait utile de mesurer l'incidence que ce choix de taille et de forme
2643 peut engendrer sur les performances de RANSAS, et en particulier pour différentes tailles d'instances. En
2644 effet, des requêtes de grande taille sont adaptées aux objets ayant une forte « couverture moyenne »,
2645 et inversement. Un choix inadapté a pour conséquence une sur/sous-segmentation des instances
2646 découvertes. Ce problème peut-être plus ou moins résolu par l'étape de *clustering* effectuée a posteriori,
2647 mais l'introduction de nouvelles méthodes pour sélectionner automatiquement la couverture de chaque
2648 requête pourrait améliorer nos résultats.

2649

2650 Dans cette thèse, nous avons adressé le problème de la découverte et de la fouille d'objets
2651 $\{c, f\}$ -fréquents. Nous avons montré que notre système permettait de découvrir et de fouiller des objets
2652 visuels fréquents efficacement dans de grandes collections multimédias. L'objectif serait maintenant de
2653 pouvoir paramétrer précisément notre système en fonction de deux seuls paramètres : la couverture c_0
2654 et la fréquence f_0 minimales. La réalisation de cet objectif supposerait que nous soyons capables de
2655 mesurer la couverture et la fréquence réelles d'un objet visuel dans une collection, ce qui est loin d'être
2656 trivial, surtout si l'on considère le fait que la plupart des méthodes que nous utilisons (ainsi que celles de
2657 l'état de l'art) sont des méthodes approximatives et basées sur de l'aléatoire.

2658

2659 L'une des évolutions envisageables de notre algorithme de calcul de score de vraisemblance basé sur
2660 la géométrie faible, serait d'ajouter de nouveaux attributs aux vecteurs de géométrie faible. On pourrait
2661 ainsi étudier l'intérêt de la translation, de la couleur, ou de toutes autres informations contextuelles,
2662 comme le type de média par exemple, afin de créer un score de vraisemblance filtrant directement les
2663 objets transmédiés.

2664

2665 Afin de diminuer l'éclatement des objets visuels découverts en plusieurs sous-groupes, nous avons
2666 proposé d'effectuer un *clustering* des instances. Cet éclatement aurait pu être atténué en utilisant une
2667 extension de requêtes lors de la phase de recherche d'objets visuels de RANSAS. On a cependant vu
2668 que l'extension de requêtes était en règle générale un processus très coûteux, et donc inadapté. Une
2669 alternative à ces deux approches serait de créer des modèles d'objets visuels au fur et à mesure de la
2670 découverte. Cette solution aurait l'avantage de permettre d'effectuer des requêtes multiples à partir des
2671 modèles.

2672

2673 Comme toutes les méthodes de recherche ou de découverte d'objets visuels, le volume des collections

2674 que nous traitons est principalement limité par les capacités des ordinateurs, et en particulier par l'espace
2675 mémoire. Il serait donc intéressant d'étudier la distribution de nos algorithmes sur de multiples nœuds de
2676 calcul. Une grande partie de nos algorithmes étant basés sur du hachage, cet objectif paraît réaliste. On
2677 pourrait alors s'attaquer à l'échelle du web et exploiter les liens hypervisuels dans les moteurs de recherche.

2678

2679 Les applications proposées dans cette thèse sont actuellement à l'état de prototypes, mais sont d'ores
2680 et déjà en phase de test par des journalistes et des sociologues dans le cadre du projet OTMedia. Nous
2681 projetons ainsi de réaliser grâce à eux des évaluations qualitatives de nos applications sur des cas
2682 d'usages réels et concrets.

Bibliographie

- 2684 [1] J. LAW-TO et al. « Video copy detection : a comparative study ». Dans : *Proceedings of the 6th ACM*
2685 *international conference on Image and video retrieval*. ACM. 2007, p. 371–378.
- 2686 [2] M.S. LEW et al. « Content-based multimedia information retrieval : State of the art
2687 and challenges ». Dans : *ACM Transactions on Multimedia Computing, Communications, and*
2688 *Applications (TOMCCAP) 2.1 (2006)*, p. 1–19.
- 2689 [3] J. SIVIC et A. ZISSERMAN. « Video Google : A text retrieval approach to object matching in videos ».
2690 Dans : *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. Ieee. 2003,
2691 p. 1470–1477.
- 2692 [4] Alexis JOLY et Olivier BUISSON. « Logo retrieval with a contrario visual query
2693 expansion ». Dans : *Proceedings of the seventeen ACM international conference on Multimedia*.
2694 MM '09. Beijing, China : ACM, 2009, p. 581–584. ISBN : 978-1-60558-608-3. DOI :
2695 <http://doi.acm.org/10.1145/1631272.1631361>.
- 2696 [5] F. PERRONNIN, J. SÁNCHEZ et T. MENSINK. « Improving the fisher kernel for large-scale image
2697 classification ». Dans : *Computer Vision--ECCV 2010 (2010)*, p. 143–156.
- 2698 [6] M.J. SWAIN et D.H. BALLARD. « Color indexing ». Dans : *International journal of computer vision 7.1*
2699 (1991), p. 11–32.
- 2700 [7] M. FLICKNER et al. « Query by image and video content : The QBIC system ». Dans : *Computer 28.9*
2701 (1995), p. 23–32.
- 2702 [8] T. GEVERS et A.W.M. SMEULDERS. « Pictoseek : Combining color and shape invariant features for
2703 image retrieval ». Dans : *Image Processing, IEEE Transactions on 9.1 (2000)*, p. 102–119.
- 2704 [9] J.R. SMITH et S.F. CHANG. « VisualSEEk : a fully automated content-based image query
2705 system ». Dans : *Proceedings of the fourth ACM international conference on Multimedia*. ACM.
2706 1997, p. 87–98.
- 2707 [10] G.D. FINLAYSON. « Color in perspective ». Dans : *Pattern Analysis and Machine Intelligence, IEEE*
2708 *Transactions on 18.10 (1996)*, p. 1034–1038.
- 2709 [11] J. HUANG et al. « Spatial color indexing and applications ». Dans : *International Journal of*
2710 *Computer Vision 35.3 (1999)*, p. 245–268.
- 2711 [12] J.Z. WANG et al. « Content-based image indexing and searching using Daubechies' wavelets ».
2712 Dans : *International Journal on Digital Libraries 1.4 (1998)*, p. 311–328.
- 2713 [13] P. SALEMBIER, T. SIKORA et BS MANJUNATH. *Introduction to MPEG-7 : multimedia content description*
2714 *interface*. John Wiley & Sons, Inc., 2002.
- 2715 [14] B.S. MANJUNATH et al. « Color and texture descriptors ». Dans : *Circuits and Systems for Video*
2716 *Technology, IEEE Transactions on 11.6 (2001)*, p. 703–715.
- 2717 [15] A. OLIVA et A. TORRALBA. « Modeling the shape of the scene : A holistic representation of the
2718 spatial envelope ». Dans : *International Journal of Computer Vision 42.3 (2001)*, p. 145–175.

BIBLIOGRAPHIE

- 2719 [16] S. ULLMAN, M. VIDAL-NAQUET, E. SALI et al. « Visual features of intermediate complexity and their
2720 use in classification ». Dans : *Nature neuroscience* 5.7 (2002), p. 682–687.
- 2721 [17] L. FEI-FEI et P. PERONA. « A bayesian hierarchical model for learning natural scene
2722 categories ». Dans : *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer
2723 Society Conference on*. T. 2. IEEE. 2005, p. 524–531.
- 2724 [18] C. HARRIS et M. STEPHENS. « A combined corner and edge detector ». Dans : *Alvey vision
2725 conference*. T. 15. Manchester, UK. 1988, p. 50.
- 2726 [19] J. SHI et C. TOMASI. « Good features to track ». Dans : *Computer Vision and Pattern Recognition,
2727 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*. IEEE. 1994, p. 593–600.
- 2728 [20] S. M. SMITH et J. M. BRADY. « SUSAN - A New Approach to Low Level Image Processing ». Dans :
2729 *International Journal of Computer Vision* 23 (1995), p. 45–78.
- 2730 [21] E. ROSTEN et T. DRUMMOND. « Machine learning for high-speed corner detection ». Dans :
2731 *Computer Vision--ECCV 2006* (2006), p. 430–443.
- 2732 [22] T. LINDBERG. « Feature detection with automatic scale selection ». Dans : *International journal of
2733 computer vision* 30.2 (1998), p. 79–116.
- 2734 [23] D.G. LOWE. « Distinctive image features from scale-invariant keypoints ». Dans : *International
2735 journal of computer vision* 60.2 (2004), p. 91–110.
- 2736 [24] H. BAY et al. « Speeded-up robust features (SURF) ». Dans : *Computer Vision and Image
2737 Understanding* 110.3 (2008), p. 346–359.
- 2738 [25] J. MATAS et al. « Robust wide-baseline stereo from maximally stable extremal regions ». Dans :
2739 *Image and Vision Computing* 22.10 (2004), p. 761–767.
- 2740 [26] K. MIKOLAJCZYK et C. SCHMID. « An affine invariant interest point detector ». Dans : *Computer
2741 Vision—ECCV 2002* (2002), p. 128–142.
- 2742 [27] K. MIKOLAJCZYK et C. SCHMID. « Scale & affine invariant interest point detectors ». Dans :
2743 *International journal of computer vision* 60.1 (2004), p. 63–86.
- 2744 [28] M. PERD'OCH, O. CHUM et J. MATAS. « Efficient representation of local geometry for large
2745 scale object retrieval ». Dans : *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE
2746 Conference on*. IEEE. 2009, p. 9–16.
- 2747 [29] A.E. JOHNSON et M. HEBERT. « Recognizing objects by matching oriented points ».
2748 Dans : *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society
2749 Conference on*. IEEE. 1997, p. 684–689.
- 2750 [30] S. LAZEBNIK, C. SCHMID et J. PONCE. « A sparse texture representation using affine-invariant
2751 regions ». Dans : *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer
2752 Society Conference on*. T. 2. IEEE. 2003, p. II–319.
- 2753 [31] T. TUYTELAARS et L. VAN GOOL. « Wide baseline stereo matching based on local, affinely invariant
2754 regions ». Dans : *british Machine vision conference*. 2000, p. 412–425.
- 2755 [32] R. ZABIH et J. WOODFILL. « Non-parametric local transforms for computing visual correspondence ».
2756 Dans : *Computer Vision ?ECCV'94* (1994), p. 151–158.
- 2757 [33] S. BELONGIE, J. MALIK et J. PUZICHA. « Shape matching and object recognition using shape
2758 contexts ». Dans : *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.4 (2002),
2759 p. 509–522.
- 2760 [34] Y. KE et R. SUKTHANKAR. « PCA-SIFT : A more distinctive representation for local image
2761 descriptors ». Dans : *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of
2762 the 2004 IEEE Computer Society Conference on*. T. 2. Ieee. 2004, p. II–506.
- 2763 [35] K. MIKOLAJCZYK et al. « A comparison of affine region detectors ». Dans : *International journal of
2764 computer vision* 65.1 (2005), p. 43–72.

- 2765 [36] J.M. MOREL et G. YU. « ASIFT : A new framework for fully affine invariant image comparison ». Dans : *SIAM Journal on Imaging Sciences* 2.2 (2009), p. 438–469.
2766
- 2767 [37] G. YU et J.M. MOREL. « ASIFT : An Algorithm for Fully Affine Invariant Comparison ». Dans : *Image Processing On Line* (2011).
2768
- 2769 [38] J.J. KOENDERINK et AJ VAN DOORN. « Representation of local geometry in the visual system ». Dans : *Biological cybernetics* 55.6 (1987), p. 367–375.
2770
- 2771 [39] LMJ FLORACK et al. « General intensity transformations and differential invariants ». Dans : *Journal of Mathematical Imaging and Vision* 4.2 (1994), p. 171–187.
2772
- 2773 [40] L. VAN GOOL, T. MOONS et D. UNGUREANU. « Affine/photometric invariants for planar intensity patterns ». Dans : *Computer Vision ?ECCV'96* (1996), p. 642–651.
2774
- 2775 [41] W.T. FREEMAN et E.H. ADELSON. « The design and use of steerable filters ». Dans : *IEEE Transactions on Pattern analysis and machine intelligence* 13.9 (1991), p. 891–906.
2776
- 2777 [42] A. JOLY. « New local descriptors based on dissociated dipoles ». Dans : *Proceedings of the 6th ACM international conference on Image and video retrieval*. Sous la dir. de Nicu SEBE et Marcel WORRING. Amsterdam, The Netherlands : ACM, juil. 2007, p. 573–580. ISBN : 978-1-59593-733-9.
2778
2779
2780
- 2781 [43] M. CALONDER et al. « Brief : Binary robust independent elementary features ». Dans : Springer, 2010, p. 778–792.
2782
- 2783 [44] S.E. GRIGORESCU, N. PETKOV et P. KRUIZINGA. « Comparison of texture features based on Gabor filters ». Dans : *Image Processing, IEEE Transactions on* 11.10 (2002), p. 1160–1167.
2784
- 2785 [45] E. KHVEDCHENYA. *Feature descriptor comparison report*. Août 2011. URL : <http://computer-vision-talks.com/2011/08/feature-descriptor-comparison-report>.
2786
- 2787 [46] T. DESELAERS, D. KEYSERS et H. NEY. « Features for image retrieval : A quantitative comparison ». Dans : *Pattern Recognition* (2004), p. 228–236.
2788
- 2789 [47] K. GRAUMAN et T. DARRELL. « The pyramid match kernel : Discriminative classification with sets of image features ». Dans : *Proceedings of the 10th International Conference on Computer Vision, ICCV*. T. 2. Beijing, China : IEEE Computer Society, oct. 2005, p. 1458–1465. ISBN : 0-7695-2334-X.
2790
2791
- 2792 [48] S. LAZEBNIK, C. SCHMID et J. PONCE. « Beyond bags of features : Spatial pyramid matching for recognizing natural scene categories ». Dans : *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY. T. 2. New York, NY, USA : IEEE Computer Society, juin 2006, p. 2169–2178. ISBN : 0-7695-2597-0.
2793
2794
2795
- 2796 [49] T.S. JAAKKOLA et D. HAUSSLER. « Exploiting generative models in discriminative classifiers ». Dans : *Advances in neural information processing systems* (1999), p. 487–493.
2797
- 2798 [50] F. PERRONNIN et C. DANCE. « Fisher kernels on visual vocabularies for image categorization ». Dans : *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY. Minneapolis, Minnesota, USA : IEEE Computer Society, juin 2007, p. 1–8.
2799
2800
- 2801 [51] M. BRESSAN, C. CIFARELLI et F. PERRONNIN. « An analysis of the relationship between painters based on their work ». Dans : *Proceedings of the International Conference on Image Processing, ICIP*. San Diego, California, USA : IEEE, oct. 2008, p. 113–116.
2802
2803
- 2804 [52] L. MARCHESOTTI, C. CIFARELLI et G. CSURKA. « A framework for visual saliency detection with applications to image thumbnailing ». Dans : *Proceedings of the 12th International Conference on Computer Vision, ICCV*. Kyoto, Japan : IEEE, oct. 2009, p. 2232–2239.
2805
2806
- 2807 [53] F. PERRONNIN et al. « Large-scale image retrieval with compressed fisher vectors ». Dans : *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY. San Francisco, CA, USA : IEEE, juin 2010, p. 3384–3391.
2808
2809

- 2810 [54] D.A. KEIM et A. HINNEBURG. « Clustering techniques for large data sets ». Dans : *Tutorial notes of*
2811 *the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. Sous la
2812 dir. d'ACM. ACM. San Diego, CA, USA, 1999, p. 141–181.
- 2813 [55] R.M. GRAY et D.L. NEUHOF. « Quantization ». Dans : *Information Theory, IEEE Transactions on* 44.6
2814 (1998), p. 2325–2383.
- 2815 [56] JH CONWAY et NJA SLOANE. « On the enumeration of lattices of determinant one ». Dans : *Journal*
2816 *of Number Theory* 15.1 (1982), p. 83–94.
- 2817 [57] T. TUYTELAARS et C. SCHMID. « Vector quantizing feature space with a regular lattice ». Dans :
2818 *Proceedings of the 11th International Conference on Computer Vision, ICCV*. Rio de Janeiro, Brazil :
2819 IEEE, oct. 2007, p. 1–8.
- 2820 [58] H. SAGAN. *Space-filling curves*. T. 2. Springer-Verlag New York, 1994.
- 2821 [59] A. JOLY. « Recherche par similarité statistique dans une grande base de signatures locales pour
2822 l'identification rapide d'extraits vidéo ». Thèse de doct. Université de la Rochelle, 2005.
- 2823 [60] S. POULLOT, O. BUISSON et M. CRUCIANU. « Z-grid-based probabilistic retrieval for scaling up
2824 content-based copy detection ». Dans : *Proceedings of the 6th ACM international conference on*
2825 *Image and video retrieval*. ACM. 2007, p. 348–355.
- 2826 [61] P. INDYK et R. MOTWANI. « Approximate Nearest Neighbors : Towards Removing the Curse
2827 of Dimensionality ». Dans : *Proceedings of the thirtieth annual ACM symposium on Theory of*
2828 *computing*. Sous la dir. d'ACM. Dallas, TX, USA, 1998, p. 604–613.
- 2829 [62] A. ANDONI et P. INDYK. « Near-optimal hashing algorithms for approximate nearest neighbor in
2830 high dimensions ». Dans : *Foundations of Computer Science, FOCS'06. 47th Annual IEEE Symposium*
2831 *on*. Sous la dir. d'IEEE. Ieee. 2006, p. 459–468.
- 2832 [63] M. DATAR et P. INDYK. « Locality-sensitive hashing scheme based on p-stable distributions ». Dans :
2833 *SCG*. ACM Press, 2004, p. 253–262.
- 2834 [64] K. TERASAWA et Y. TANAKA. « Spherical lsh for approximate nearest neighbor search on unit
2835 hypersphere ». Dans : *Algorithms and Data Structures (2007)*, p. 27–38.
- 2836 [65] M. RAGINSKY et S. LAZEBNIK. « Locality-sensitive binary codes from shift-invariant kernels ». Dans :
2837 *The Neural Information Processing Systems 22 (2009)*.
- 2838 [66] A. BRODER. « On the Resemblance and Containment of Documents ». Dans : *SEQUENCES '97 :*
2839 *Proceedings of the Compression and Complexity of Sequences 1997*. Washington, DC, USA : IEEE
2840 Computer Society, 1997, p. 21.
- 2841 [67] A. GUTTMAN. *R-trees : a dynamic index structure for spatial searching*. T. 14. 2. ACM, 1984.
- 2842 [68] Y. MANOPOULOS, A. NANOPOULOS et E. TOUSIDOU. « Advanced signature indexing for multimedia
2843 and web applications (Series on advances in database systems, Vol. 27) ». Dans : *Recherche 67*
2844 (2003), p. 02.
- 2845 [69] N. BECKMANN et al. *The R*-tree : an efficient and robust access method for points and rectangles*.
2846 T. 19. 2. ACM, 1990.
- 2847 [70] D.A. WHITE et R. JAIN. « Similarity indexing with the SS-tree ». Dans : *Data Engineering, 1996.*
2848 *Proceedings of the Twelfth International Conference on*. IEEE. 1996, p. 516–523.
- 2849 [71] N. KATAYAMA et S. SATOH. « The SR-tree : An index structure for high-dimensional nearest neighbor
2850 queries ». Dans : *ACM SIGMOD Record*. T. 26. 2. ACM. 1997, p. 369–380.
- 2851 [72] N. BOUTELDJA, V. GOUET-BRUNET et M. SCHOLL. « Evaluation of strategies for multiple sphere
2852 queries with local image descriptors ». Dans : *Proceedings of SPIE*. T. 6073. 2006, p. 87–98.
- 2853 [73] S. BERCHTOLD, D.A. KEIM et H.P. KRIEGEL. *The X-tree : An index structure for high-dimensional data*.
2854 Bibliothek der Universität Konstanz, 1996.

- 2855 [74] J.L. BENTLEY. « Multidimensional binary search trees used for associative searching ». Dans :
2856 *Communications of the ACM* 18.9 (1975), p. 509–517.
- 2857 [75] J.L. BENTLEY et J.H. FRIEDMAN. « Data structures for range searching ». Dans : *ACM Computing*
2858 *Surveys (CSUR)* 11.4 (1979), p. 397–409.
- 2859 [76] J.T. ROBINSON. « The KDB-tree : a search structure for large multidimensional dynamic indexes ».
2860 Dans : *Proceedings of the 1981 ACM SIGMOD international conference on Management of data*.
2861 ACM. 1981, p. 10–18.
- 2862 [77] A. HENRICH. « The LSDh-tree : An access structure for feature vectors ». Dans : *Data Engineering,*
2863 *1998. Proceedings., 14th International Conference on*. IEEE. 1998, p. 362–369.
- 2864 [78] Y. AMIT et D. GEMAN. « Shape quantization and recognition with randomized trees ». Dans : *Neural*
2865 *computation* 9.7 (1997), p. 1545–1588.
- 2866 [79] V. LEPETIT, P. LAGGER et P. FUA. « Randomized Trees for Real-Time Keypoint Recognition ». Dans :
2867 *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY.
2868 CVPR '05. San Diego, CA, USA : IEEE Computer Society, juin 2005, p. 775–781. ISBN :
2869 0-7695-2372-2. DOI : [10.1109/CVPR.2005.288](https://doi.org/10.1109/CVPR.2005.288).
- 2870 [80] C. SILPA-ANAN et R. HARTLEY. « Localisation using an image-map ». Dans : *Proceedings of the*
2871 *Australian Conference on Robotics and Automation*. Citeseer. 2004.
- 2872 [81] D. NISTER et H. STEWENIUS. « Scalable recognition with a vocabulary tree ». Dans : *Computer*
2873 *Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. T. 2. Ieee. 2006,
2874 p. 2161–2168.
- 2875 [82] J. PHILBIN. « Scalable Object Retrieval in Very Large Image Collections ». Thèse
2876 de doct. University of Oxford, 2010. URL : [http://marcade.robots.ox.ac.uk:](http://marcade.robots.ox.ac.uk:8080/~vgg/publications/2010/Philbin10c)
2877 [8080/~vgg/publications/2010/Philbin10c](http://marcade.robots.ox.ac.uk:8080/~vgg/publications/2010/Philbin10c).
- 2878 [83] T. ZHANG, R. RAMAKRISHNAN et M. LIVNY. « BIRCH : an efficient data clustering method for very
2879 large databases ». Dans : *ACM SIGMOD Record*. T. 25. 2. ACM. 1996, p. 103–114.
- 2880 [84] S.A. BERRANI. « Recherche approximative de plus proches voisins avec contrôle probabiliste de la
2881 précision ; application à la recherche d'images par le contenu ». Dans : (2004).
- 2882 [85] R. SALAKHUTDINOV, A. MNH et G. HINTON. « Restricted Boltzmann machines for collaborative
2883 filtering ». Dans : *Proceedings of the 24th international conference on Machine learning*. ACM.
2884 2007, p. 791–798.
- 2885 [86] Y. WEISS, A. TORRALBA et R. FERGUS. « Spectral hashing ». Dans : *Neural Information Processing*
2886 *Systems*. Vancouver, B.C., Canada, 2008.
- 2887 [87] J. HEO et al. « Spherical Hashing ». Dans : *Conference on Computer Vision and Pattern Recognition*.
2888 Sous la dir. d'IEEE Computer SOCIETY. Providence, RI, USA : IEEE, juin 2012, p. 2957–2964.
- 2889 [88] B. KULIS et K. GRAUMAN. « Kernelized locality-sensitive hashing for scalable image search ». Dans :
2890 *Computer Vision, 2009 IEEE 12th International Conference on*. Ieee. 2009, p. 2130–2137.
- 2891 [89] B. KULIS et K. GRAUMAN. « Kernelized locality-sensitive hashing ». Dans : *Pattern Analysis and*
2892 *Machine Intelligence, IEEE Transactions on* 34.6 (2012), p. 1092–1104.
- 2893 [90] A. JOLY et O. BUISSON. « Random Maximum Margin Hashing ». Dans : *CVPR*. Colorado springs,
2894 United States : IEEE, juin 2011.
- 2895 [91] A. BOURRIER et al. *Nearest neighbor search for arbitrary kernels with explicit embeddings*. Rap.
2896 tech. INRIA Project-Teams Metiss et Texmex, 2012.
- 2897 [92] L. PAULEVÉ, H. JÉGOU et L. AMSALEG. « Locality sensitive hashing : A comparison of hash function
2898 types and querying mechanisms ». Dans : *Pattern Recognition Letters* 31.11 (2010), p. 1348–1358.
- 2899 [93] H. JÉGOU, M. DOUZE et C. SCHMID. « Product quantization for nearest neighbor search ». Dans :
2900 *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 33.1 (2011), p. 117–128.

BIBLIOGRAPHIE

- 2901 [94] J. PHILBIN et al. « Object Retrieval with Large Vocabularies and Fast Spatial Matching ». Dans :
2902 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2007.
- 2903 [95] M. VARMA et A. ZISSERMAN. « Texture classification : Are filter banks necessary? »
2904 Dans : *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society*
2905 *Conference on*. T. 2. IEEE. 2003, p. II-691.
- 2906 [96] K. MIKOLAJCZYK, B. LEIBE et B. SCHIELE. « Multiple object class detection with a generative model ».
2907 Dans : *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. T. 1.
2908 IEEE. 2006, p. 26-36.
- 2909 [97] F. JURIE et B. TRIGGS. « Creating efficient codebooks for visual recognition ». Dans : *Proceedings of*
2910 *the 10th International Conference on Computer Vision, ICCV*. Beijing, China : IEEE Computer Society,
2911 oct. 2005. ISBN : 0-7695-2334-X.
- 2912 [98] K. FUKUNAGA et L. HOSTETLER. « The estimation of the gradient of a density function, with
2913 applications in pattern recognition ». Dans : *Information Theory, IEEE Transactions on* 21.1 (1975),
2914 p. 32-40.
- 2915 [99] R. WEBER, H.J. SCHEK et S. BLOTT. « A quantitative analysis and performance study for
2916 similarity-search methods in high-dimensional spaces ». Dans : *Proceedings of the International*
2917 *Conference on Very Large Data Bases*. INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS.
2918 1998, p. 194-205.
- 2919 [100] K. LING et G. WU. « Multimedia Technology (ICMT), 2011 International Conference on ». Dans :
2920 *ICMT*. Sous la dir. d'IEEE. Hangzhou, China, juil. 2011, p. 4929-4932.
- 2921 [101] C. SILPA-ANAN et R. HARTLEY. « Optimised KD-trees for fast image descriptor matching ». Dans :
2922 *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY.
2923 Anchorage, Alaska, USA : IEEE Computer Society, juin 2008.
- 2924 [102] M. ALY, M. MUNICH et P. PERONA. « Multiple Dictionaries for Bag of Words Large Scale Image
2925 Search ». Dans : *International Conference on Image Processing (ICIP)*. Sous la dir. d'IEEE. Brussels,
2926 Belgium, sept. 2011.
- 2927 [103] A. JOLY et O. BUISSON. « A Posteriori Multi-Probe Locality Sensitive Hashing ». Dans : *ACM*
2928 *International Conference on Multimedia (MM'08)*. Sous la dir. d'Abdulmoteleb EL-SADDIK et al.
2929 Vancouver, British Columbia, Canada : ACM, oct. 2008, p. 209-218. ISBN : 978-1-60558-303-7.
- 2930 [104] J. PHILBIN et al. « Lost in Quantization : Improving Particular Object Retrieval in Large Scale
2931 Image Databases ». Dans : *Proceedings of the IEEE Conference on Computer Vision and Pattern*
2932 *Recognition*. 2008.
- 2933 [105] A. JOLY, O. BUISSON et C. FRELICOT. « Statistical similarity search applied to content-based video
2934 copy detection ». Dans : *Data Engineering Workshops, 2005. 21st International Conference on*.
2935 IEEE. 2005, p. 1285-1285.
- 2936 [106] Q. Lv et al. « Multi-probe LSH : efficient indexing for high-dimensional similarity search ».
2937 Dans : *Proceedings of the 33rd international conference on Very large data bases*. VLDB '07.
2938 Vienna, Austria : VLDB Endowment, 2007, p. 950-961. ISBN : 978-1-59593-649-3. URL :
2939 <http://portal.acm.org/citation.cfm?id=1325851.1325958>.
- 2940 [107] S. ARYA et al. *An optimal algorithm for approximate nearest neighbor searching in fixed dimensions*.
2941 Rap. tech. College Park, MD, USA, 1995.
- 2942 [108] W. DONG, M. CHARIKAR et K. LI. « Asymmetric distance estimation with sketches for similarity
2943 search in high-dimensional spaces ». Dans : *Proceedings of the 31st annual international ACM*
2944 *SIGIR conference on Research and development in information retrieval*. ACM. 2008, p. 123-130.
- 2945 [109] H. JEGOU, M. DOUZE et C. SCHMID. « Hamming embedding and weak geometric consistency for
2946 large scale image search ». Dans : *Computer Vision--ECCV 2008 (2008)*, p. 304-317.

- 2947 [110] W. ZHAO, X. WU et C.W. NGO. « On the Annotation of Web Videos by Efficient Near-Duplicate
2948 Search ». Dans : *IEEE Transactions on Multimedia* 12.5 (2010), p. 448–461.
- 2949 [111] H. XIE et al. « Pairwise weak geometric consistency for large scale image search ». Dans :
2950 *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*. ICMR '11. Trento,
2951 Italy : ACM, 2011, 42 :1–42 :8. ISBN : 978-1-4503-0336-1. DOI : [10.1145/1991996.1992038](https://doi.org/10.1145/1991996.1992038).
- 2952 [112] J. RABIN. « Approches robustes pour la comparaison d'images et la reconnaissance d'objets ».
2953 Thèse de doct. Telecom ParisTech, 2009.
- 2954 [113] P.J. ROUSSEEUW. « Least median of squares regression ». Dans : *Journal of the American statistical*
2955 *association* (1984), p. 871–880.
- 2956 [114] P.J. ROUSSEEUW. « Multivariate estimation with high breakdown point ». Dans : *Mathematical*
2957 *statistics and applications* 8 (1985), p. 283–297.
- 2958 [115] P.V.C. HOUGH. « Machine analysis of bubble chamber pictures ». Dans : *International Conference*
2959 *on High Energy Accelerators and Instrumentation*. T. 73. 1959.
- 2960 [116] D. H. BALLARD. « Generalizing the Hough Transform to Detect Arbitrary Shapes ». Dans : *Readings*
2961 *in Computer Vision : Issues, Problems, Principles, and Paradigms*. Sous la dir. de M. A. FISCHLER et
2962 O. FIRSCHEIN. Los Altos, CA. : Kaufmann, 1987, p. 714–725.
- 2963 [117] M. A. FISCHLER et R. C. BOLLES. « Random Sample Consensus : A Paradigm for Model Fitting with
2964 Applications to Image Analysis and Automated Cartography ». Dans : *Commun. ACM* 24.6 (1981),
2965 p. 381–395.
- 2966 [118] P.H.S. TORR et C. DAVIDSON. « IMPSAC : synthesis of importance sampling and random sample
2967 consensus ». Dans : *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 25.3 (2003),
2968 p. 354–364.
- 2969 [119] C.M. CHENG et S.H. LAI. « A consensus sampling technique for fast and robust model fitting ».
2970 Dans : *Pattern Recognition* 42.7 (2009), p. 1318–1329.
- 2971 [120] O. CHUM et J. MATAS. « Optimal randomized RANSAC ». Dans : *Pattern Analysis and Machine*
2972 *Intelligence, IEEE Transactions on* 30.8 (2008), p. 1472–1482.
- 2973 [121] L. MOISAN et B. STIVAL. « A probabilistic criterion to detect rigid point matches between two
2974 images and estimate the fundamental matrix ». Dans : *International Journal of Computer Vision*
2975 57.3 (2004), p. 201–218.
- 2976 [122] P.H.S. TORR. « Motion segmentation and outlier detection ». Thèse de doct. University of Oxford,
2977 1995.
- 2978 [123] M. ZULIANI, CS KENNEY et BS MANJUNATH. « The multiransac algorithm and its application to detect
2979 planar homographies ». Dans : *Image Processing, 2005. ICIP 2005. IEEE International Conference*
2980 *on*. T. 3. IEEE. 2005, p. III–153.
- 2981 [124] K. SCHINDLER et D. SUTER. « Two-view multibody structure-and-motion with outliers through
2982 model selection ». Dans : *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 28.6
2983 (2006), p. 983–995.
- 2984 [125] T. SATTLER, B. LEIBE et L. KOBBELT. « SCRAMSAC : Improving RANSAC's efficiency with a spatial
2985 consistency filter ». Dans : *Computer Vision, 2009 IEEE 12th International Conference on*. Ieee.
2986 2009, p. 2090–2097.
- 2987 [126] O. CHUM et J. MATAS. « Matching with PROSAC-progressive sample consensus ». Dans : *Computer*
2988 *Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. T. 1. Ieee.
2989 2005, p. 220–226.
- 2990 [127] P.H.S. TORR et A. ZISSERMAN. « MLESAC : A new robust estimator with application to estimating
2991 image geometry ». Dans : *Computer Vision and Image Understanding* 78.1 (2000), p. 138–156.

BIBLIOGRAPHIE

- 2992 [128] A.P. DEMPSTER, N.M. LAIRD et D.B. RUBIN. « Maximum likelihood from incomplete data via the
2993 EM algorithm ». Dans : *Journal of the Royal Statistical Society. Series B (Methodological)* (1977),
2994 p. 1–38.
- 2995 [129] C.V. STEWART. « MINPRAN : A new robust estimator for computer vision ». Dans : *Pattern Analysis*
2996 *and Machine Intelligence, IEEE Transactions on* 17.10 (1995), p. 925–938.
- 2997 [130] A. DESOLNEUX, L. MOISAN et J.M. MOREL. « Meaningful Alignments ». Dans : *Int. J. Comput. Vision*
2998 40 (1 2000), p. 7–23. ISSN : 0920-5691. DOI : [10.1023/A:1026593302236](https://doi.org/10.1023/A:1026593302236).
- 2999 [131] J. RABIN et al. « MAC-RANSAC : a robust algorithm for the recognition of multiple objects ». Dans :
3000 *Proceedings of 3DPTV 2010*. 2009.
- 3001 [132] J. CECH et R. SÁRA. « Efficient Sampling of Disparity Space for Fast And Accurate Matching ». Dans :
3002 *Conference on Computer Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY.
3003 Minneapolis, Minnesota, USA : IEEE Computer Society, juin 2007.
- 3004 [133] J. CECH, J. MATAS et M. PERDOCH. « Efficient Sequential Correspondence Selection
3005 by Cosegmentation ». Dans : *IEEE Trans. Pattern Anal. Mach. Intell.* 32.9 (sept. 2010),
3006 p. 1568–1581. ISSN : 0162-8828. DOI : [10.1109/TPAMI.2009.176](https://doi.org/10.1109/TPAMI.2009.176). URL : <http://dx.doi.org/10.1109/TPAMI.2009.176>.
3007
- 3008 [134] O. CHUM et al. « Total Recall : Automatic Query Expansion with a Generative Feature Model for
3009 Object Retrieval ». Dans : *Proceedings of the 11th International Conference on Computer Vision,*
3010 *Rio de Janeiro, Brazil*. 2007.
- 3011 [135] C. BUCKLEY et al. « Automatic Query Expansion Using SMART : TREC 3 ». Dans : *TREC*. 1994, p. 0–.
- 3012 [136] M. MITRA, A. SINGHAL et C. BUCKLEY. « Improving automatic query expansion ». Dans :
3013 *Proceedings of the 21st annual international ACM SIGIR conference on Research and development*
3014 *in information retrieval*. ACM. 1998, p. 206–214.
- 3015 [137] O. CHUM et al. « Total Recall II : Query Expansion Revisited ». Dans : *Conference on Computer*
3016 *Vision and Pattern Recognition*. Sous la dir. d'IEEE Computer SOCIETY. Colorado Springs, CO, USA :
3017 IEEE, juin 2011, p. 889–896.
- 3018 [138] R. ARANDJELOVIC et A. ZISSERMAN. « Three things everyone should know to improve
3019 object retrieval ». Dans : *Conference on Computer Vision and Pattern Recognition*. Sous la dir.
3020 d'IEEE Computer SOCIETY. Providence, RI, USA : IEEE, juin 2012, p. 2911–2918.
- 3021 [139] J. REVAUD, M. DOUZE, C. SCHMID et al. « Correlation-Based Burstiness for Logo Retrieval ». Dans :
3022 *Proceedings of the 20th ACM international conference on Multimedia*. Sous la dir. d'ACM. Nara,
3023 Japan, 2012.
- 3024 [140] J. SIVIC et A. ZISSERMAN. « Video data mining using configurations of viewpoint invariant regions ».
3025 Dans : *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE*
3026 *Computer Society Conference on*. T. 1. IEEE. 2004, p. 1–488.
- 3027 [141] T. QUACK, V. FERRARI et L. VAN GOOL. « Video mining with frequent itemset configurations ». Dans :
3028 *Image and Video Retrieval* (2006), p. 360–369.
- 3029 [142] R. AGRAWAL, R. SRIKANT et al. « Fast algorithms for mining association rules ». Dans : *Proc. 20th Int.*
3030 *Conf. Very Large Data Bases, VLDB*. T. 1215. Santiago de Chile, Chile, 1994, p. 487–499.
- 3031 [143] J. YUAN, Y. WU et M. YANG. « From frequent itemsets to semantically meaningful visual patterns ».
3032 Dans : *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and*
3033 *data mining*. ACM. 2007, p. 864–873.
- 3034 [144] J. YUAN, Y. WU et M. YANG. « Discovery of collocation patterns : from visual words to visual
3035 phrases ». Dans : *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*
3036 *IEEE*. 2007, p. 1–8.

- 3037 [145] J. YUAN et al. « Common spatial pattern discovery by efficient candidate pruning ». Dans : *Image*
3038 *Processing, 2007. ICIP 2007. IEEE International Conference on*. T. 1. IEEE. 2007, p. 1–165.
- 3039 [146] J. YUAN et Y. WU. « Spatial random partition for common visual pattern discovery ». Dans :
3040 *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE. 2007, p. 1–8.
- 3041 [147] A. ANJULAN et N. CANAGARAJAH. « A novel video mining system ». Dans : *Image Processing, 2007.*
3042 *ICIP 2007. IEEE International Conference on*. Sous la dir. d'IEEE. T. 1. IEEE. Brussels, Belgium, 2007,
3043 p. 1–185.
- 3044 [148] A. ANJULAN et N. CANAGARAJAH. « A Unified Framework for Object Retrieval and Mining ». Dans :
3045 *Circuits and Systems for Video Technology, IEEE Transactions on* 19.1 (2009), p. 63–76.
- 3046 [149] T.H. GORMEN et al. « Introduction to algorithms ». Dans : *MIT Press* 44 (1990), p. 97–138.
- 3047 [150] S. BRIN et L. PAGE. « The anatomy of a large-scale hypertextual Web search engine ». Dans :
3048 *Computer networks and ISDN systems* 30.1 (1998), p. 107–117.
- 3049 [151] U. BRANDES. « A faster algorithm for betweenness centrality ». Dans : *Journal of Mathematical*
3050 *Sociology* 25.2 (2001), p. 163–177.
- 3051 [152] T. HOFMANN. « Unsupervised learning by probabilistic latent semantic analysis ». Dans : *Machine*
3052 *Learning* 42.1 (2001), p. 177–196.
- 3053 [153] D.M. BLEI, A.Y. NG et M.I. JORDAN. « Latent dirichlet allocation ». Dans : *the Journal of machine*
3054 *Learning research* 3 (2003), p. 993–1022.
- 3055 [154] P. QUELHAS et al. « Modeling scenes with local descriptors and latent aspects ». Dans : *Computer*
3056 *Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*. T. 1. IEEE. 2005, p. 883–890.
- 3057 [155] J. SIVIC et al. « Discovering object categories in image collections ». Dans : (2005).
- 3058 [156] J. PHILBIN, J. SIVIC et A. ZISSERMAN. « Geometric LDA : A Generative Model for Particular Object
3059 Discovery ». Dans : *Proceedings of the British Machine Vision Conference*. 2008.
- 3060 [157] O. CHUM et al. « Scalable near identical image and shot detection ». Dans :
3061 *Proceedings of the 6th ACM international conference on Image and video retrieval*. CIVR '07.
3062 Amsterdam, The Netherlands : ACM, 2007, p. 549–556. ISBN : 978-1-59593-733-9. DOI :
3063 [10.1145/1282280.1282359](https://doi.org/10.1145/1282280.1282359).
- 3064 [158] O. CHUM, J. PHILBIN et A. ZISSERMAN. « Near duplicate image detection : min-hash and tf-idf
3065 weighting ». Dans : *Proceedings of the British Machine Vision Conference*. T. 3. 2008, p. 4.
- 3066 [159] R.A. BAEZA-YATES et B. RIBEIRO-NETO. *Modern Information Retrieval*. Boston, MA, USA :
3067 Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN : 020139829X.
- 3068 [160] O. CHUM, M. PERDOCH et J. MATAS. « Geometric min-Hashing : Finding a (thick) needle
3069 in a haystack ». Dans : *Conference on Computer Vision and Pattern Recognition*. Sous la dir.
3070 d'IEEE Computer SOCIETY. Miami, Florida, USA : IEEE Computer Society, juin 2009, p. 17–24. ISBN :
3071 978-1-4244-3992-8.
- 3072 [161] C. BORGELT. « Frequent item set mining ». Dans : *Wiley Interdisciplinary Reviews : Data Mining and*
3073 *Knowledge Discovery* 2.6 (2012), p. 437–456. ISSN : 1942-4795. DOI : [10.1002/widm.1074](https://doi.org/10.1002/widm.1074).
- 3074 [162] P. LETESSIER, O. BUISSON et A. JOLY. « Consistent visual words mining with adaptive sampling ».
3075 Dans : *ICMR*. Trento, Italy : ACM, avr. 2011, 49 :1–49 :8.
- 3076 [163] L. DEVROYE et L. DEVROYE. « Non-uniform random variate generation ». Dans : (1986).
- 3077 [164] F. OLKEN. « Random Sampling from Databases ». Thèse de doct. University of California, 1993.
- 3078 [165] S.K. THOMPSON. « Adaptive sampling ». Dans : *The Survey Statistician*. 1995.
- 3079 [166] H. JÉGOU, M. DOUZE et C. SCHMID. « On the burstiness of visual elements ». Dans : *Conference on*
3080 *Computer Vision & Pattern Recognition*. Juin 2009.

BIBLIOGRAPHIE

- 3081 [167] S. van DONGEN. *MCL-edge : Analysis of networks with millions of nodes*. 2012. URL :
3082 <http://www.micans.org/mcl/index.html>.
- 3083 [168] S. VAN DONGEN. « A cluster algorithm for graphs ». Dans : *Report-Information systems 10* (2000),
3084 p. 1–40.
- 3085 [169] S. PEYRONNET. *Méthodes de clustering*. 2010. URL : <http://www.spoonylife.org/algorithms-and-computation/methodes-de-clustering>.
3086
- 3087 [170] Y. CHEN, J.Z. WANG et R. KROVETZ. « Content-based image retrieval by clustering ». Dans :
3088 *Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*.
3089 ACM. 2003, p. 193–200.
- 3090 [171] P. LETESSIER, O. BUISSON et A. JOLY. « Scalable mining of small visual objects ». Dans : *Proceedings*
3091 *of the 20th ACM international conference on Multimedia*. MM '12. Nara, Japan : ACM, 2012,
3092 p. 599–608. ISBN : 978-1-4503-1089-5. DOI : [10.1145/2393347.2393431](https://doi.org/10.1145/2393347.2393431).
- 3093 [172] X. WU et S. SATOH. « Temporal recurrence hashing algorithm for mining commercials
3094 from multimedia streams ». Dans : *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE*
3095 *International Conference on*. IEEE. 2011, p. 2324–2327.
- 3096 [173] N. PUTPUEK et al. « Unified Approach to Detection and Identification of Commercial Films
3097 by Temporal Occurrence Pattern ». Dans : *Pattern Recognition (ICPR), 2010 20th International*
3098 *Conference on*. IEEE. 2010, p. 3288–3291.
- 3099 [174] A. JOLY et al. « Visual-Based Transmedia Events Detection ». Dans : *Proceedings of the 20th ACM*
3100 *international conference on Multimedia*. ACM. 2012.
- 3101 [175] Z.J. ZHA et al. « Visual query suggestion ». Dans : *Proceedings of the 17th ACM international*
3102 *conference on Multimedia*. Sous la dir. d'ACM. Beijing, China, 2009, p. 15–24.
- 3103 [176] A. HAMZAoui et al. « Object-based visual query suggestion ». Dans : *Multimedia Tools and*
3104 *Applications* 11042 (2013). Sous la dir. de Springer US. DOI : [10.1007/s11042-012-1340-5](https://doi.org/10.1007/s11042-012-1340-5).
- 3105 [177] A. HAMZAoui. « Shared-Neighbours methods for visual content structuring and mining ». Thèse de
3106 doct. 2012.
- 3107 [178] J. PHILBIN et A. ZISSERMAN. « Object mining using a Matching Graph on Very Large Image
3108 Collections ». Dans : *Computer Vision, Graphics Image Processing, 2008. ICVGIP '08. Sixth Indian*
3109 *Conference on*. 2008, p. 738–745.
- 3110 [179] K. GRAUMAN et T. DARRELL. « Unsupervised Learning of Categories from Sets of Partially Matching
3111 Image Features ». Dans : *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society*
3112 *Conference on*. T. 1. 6. 2006, p. 19–25.
- 3113 [180] O. CHUM et J. MATAS. « Large-Scale Discovery of Spatially Related Images ». Dans : *IEEE*
3114 *Transactions on Pattern Analysis and Machine Intelligence* 32 (2010), p. 371–377.
- 3115 [181] G. XU et al. « Co-clustering analysis of weblogs using bipartite spectral projection approach ».
3116 Dans : *Knowledge-Based and Intelligent Information and Engineering Systems* (2010), p. 398–407.
- 3117 [182] H. ZHA et al. « Bipartite graph partitioning and data clustering ». Dans : *Proceedings of the tenth*
3118 *international conference on Information and knowledge management*. ACM. 2001, p. 25–32.