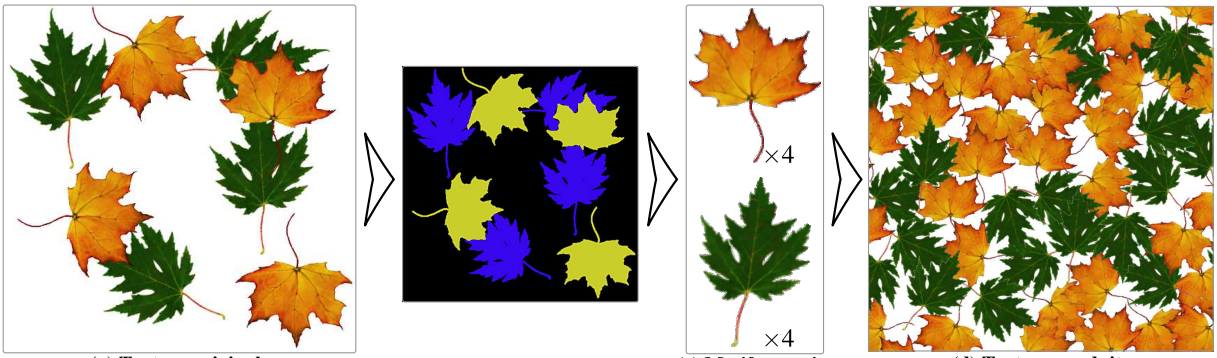
Revue Électronique Francophone d'Informatique Graphique, Volume 3, Numéro 2, pp. 35–45, 2009

Analyse et synthèse de textures composées de motifs répétitifs

Pierre-Edouard Landes & Cyril Soler

Universités de Grenoble, INRIA, LJK



(a) Texture originale (b) Carte des instances (c) Motifs extraits (d) Texture produite

Figure 1: *Cet article propose une nouvelle méthode d'analyse de textures composées de motifs : en exploitant la répétition des motifs dans l'image source* (a)*, la méthode présentée identifie chaque motif* (c) *ainsi que ses instances dans l'image* (b)*. Les instances peuvent être partielles par effet de recouvrement. À partir de cette information il est alors possible de générer une image visuellement proche* (d)*. Bien que n'apparaissant jamais entière dans l'échantillon, la feuille verte a néanmoins été reconstruite avant re-synthèse.*

Résumé

Les recherches présentées dans cet article visent à l'analyse et à la re-synthèse de textures arborant une stochasticité qualifiée ici de "haut-niveau". De telles textures sont constituées d'une distribution quelconque de différents motifs *susceptibles de se recouvrir partiellement. Leur distribution elle-même peut être aléatoire ou bien obéir à des règles de disposition géométrique. Entrent ainsi dans cette catégorie, arrangements de primitives 2d et textures quasi-régulières.*

Le principe fondamental de l'approche proposée est d'extraire les motifs en exploitant la répétitivité de leurs sous-parties au sein de l'image. Il devient alors possible de s'affranchir des contraintes et limitations qu'impose une analyse markovienne à l'échelle des pixels et obtenir des composants pertinents permettant une analyse adaptée de l'image d'entrée.

Mots clé : analyse de textures, synthèse de textures à base de patches, descripteurs locaux

1. Introduction

L'omniprésence des textures dans le domaine de la synthèse d'images est indiscutable. Intensivement utilisées dans les jeux vidéos, elles permettent d'enrichir considérablement l'aspect final des rendus tout en épargnant un très fastidieux travail de modélisation. Nombreuses sont les autres applications, depuis la visualisation de données 3d jusqu'au *design* de sites internet les utilisant tant le gain visuel et esthétique qu'elles confèrent est immédiat. Dès lors, il est compréhensible que l'automatisation de leur création fasse l'objet d'intenses recherches en infographie.

La synthèse automatique de textures par l'exemple constitue notamment un défi important. Elle consiste à générer, à partir d'un échantillon fourni en entrée, une nouvelle texture donnant l'illusion d'être issue du même processus stochastique que l'exemple. La difficulté est alors de parvenir à ce que la texture produite paraisse visuellement similaire à cet exemple et ce sans répétition apparente. La technique de synthèse doit alors parvenir à identifier et à exploiter la nature aléatoire de la texture, tout en assurant la préservation d'éventuelles structures caractéristiques. L'impression-

nante variété des textures rend le respect de l'ensemble de ces conditions extrêmement délicat et il n'est guère étonnant de constater qu'aucun algorithme de synthèse ne peut les remplir pour toutes les différentes classes de textures.

La difficulté est d'autant plus importante si le caractère aléatoire de la texture d'entrée n'intervient pas à l'échelle du pixel, mais à un niveau plus élevé. Un exemple typique est celui d'une distribution aléatoire d'objets ou de formes clairement discernables à l'image de l'arrangement de feuilles présenté en figure 1. La texture ne doit plus être traitée uniquement comme une distribution de couleurs, mais plutôt comme une distribution de primitives, la stochasticité de la texture se manifestant via leur emplacement et leur apparence.

C'est justement ce genre de textures, qualifiées ici de *textures à stochasticité de haut niveau*, que nous tâchons d'analyser et décrire en tant que distributions de motifs. La figure 1 en présente une illustration, avec sur la gauche l'échantillon à analyser. Il consiste clairement en une accumulation de feuilles que le système visuel humain – et le cerveau – identifie sans aucune difficulté. À défaut de disposer de connaissances *a priori* qui permettraient une reconnaissance de ces feuilles en tant qu'entités sémantiques, nous choisissons d'exploiter leur répétition au sein de la texture afin de les extraire et capturer leur distribution.

Jusqu'alors, aucune technique de synthèse ne visait à l'extraction explicite de formes en vue de leur préservation ; de même, aucune méthode d'analyse n'assurait l'utilisabilité des textons identifiés en vue d'une re-synthèse. Nous prenons également le parti de procéder à l'analyse automatique de l'échantillon et ce sans aucune connaissance *a priori*, qu'il s'agisse de modèles des formes à identifier ou d'hypothèses de régularité.

Nous prouvons que lier la répétitivité des motifs à leur détectabilité permet une analyse pertinente et utilisons à cette fin des descripteurs locaux invariants par similarités 2d. Notre approche ne peut donc associer que les formes séparées par ce type de transformation. Cette limitation pourra néanmoins être levée par l'usage de descripteurs plus élaborés et l'utilisation en amont de notre méthode d'algorithmes de correction des déformations perspectives.

2. Travaux précédents

2.1. Synthèse de textures

La recherche en synthèse de textures est l'une des plus conséquentes en infographie. Nous présentons donc ici les principales familles de techniques, accompagnées des articles jugés les plus représentatifs. Nous soulignons également les limitations de celles-ci dans notre cadre : la re-synthèse d'arrangements de primitives identifiables, et disposées selon une distribution quelconque.

Synthèse paramétrique : La synthèse de textures via leur modélisation explicite, soit à base de fonctions procédurales [Per85, Wor96] ou par l'inférence des paramètres d'un modèle statistique [PS00], ne sont ici guère adaptées car ardues à généraliser pour la synthèse par l'exemple sans connaissances *a priori*. Elles consituent néanmoins des outils efficaces pour la création de textures de bruit. Notons en outre que des techniques procédurales ont été proposées afin de permettre la synthèse d'arrangements de primitives [LD05]. Cependant, ces travaux ne visent aucunement à la détection de celles-ci, la gestion des recouvrements ou la capture de leur agencement.

Échantillonnage non paramétrique : La plupart des techniques actuelles de synthèse par l'exemple considèrent la distribution des couleurs de l'échantillon comme la réalisation d'un champ markovien. L'état d'un pixel ne dépend alors que de statistiques locales établies dans son voisinage dans l'image. Heeger et Bergen proposent ainsi une méthode transformant une texture de bruit en une texture similaire à l'exemple par transfert multi-échelle de statistiques [HB95]. Supposant l'auto-similarité multi-échelle de l'exemple, leur méthode peine en présence d'inhomogénéités et plus communément de structures. D'autres travaux choisissent d'échantillonner directement la texture d'entrée pour générer leur sortie [Bon97, EL99, WL00]. Cet échantillonnage est contrôlé par l'appariement de voisinages de pixels dont la similarité visuelle est généralement évaluée comme la distance euclidienne en espace RGB. L'extension multi-résolution des voisinages utilisés permet d'aisément capturer les dépendances entre bandes de fréquences. Les pixels de sortie étant synthétisés itérativement, il est également possible en examinant les parties déjà obtenues d'en favoriser la cohérence [Ash01]. En s'affranchissant de la contrainte de dépendance à l'ordre de parcours des pixels [WL03], ces techniques ont pu être parallélisées et bénéficier d'implémentations extrêmement efficaces sur cartes graphiques [LH05]. Ces méthodes, malgré leur apparente simplicité, fournissent des résultats convaincants sur de nombreux types de textures. Elles sont par ailleurs d'une grande contrôlabilité dont il est aisé de tirer profit, soit pour en raffiner la mesure de similarité entre voisinages [LH06], soit pour proposer des outils puissants d'édition [BD02]. Néanmoins, elles ne parviennent pas à restituer en sortie les structures dont la taille excède celle des voisinages utilisés. Notre objectif est donc de parvenir à obtenir automatiquement une représentation plus haut-niveau de l'exemple, permettant de manipuler lesdites structures directement, et non plus comme des amas de pixels évalués indépendamment.

Synthèse par patches : Afin de préserver de plus grandes structures, des techniques procédant à l'ajout, non plus de pixels, mais de patches ont été établies. La principale difficulté est alors de limiter tout artéfact visuel aux zones de recouvrement entre ceux-ci, que ce soit par *blending* [PFH00] ou par calcul de chemins optimaux de raccord

[EF01, KSE*03]. Cependant, l'obtention de patches pertinents n'est pas une question triviale et, sans véritable analyse de l'entrée, les structures seront toujours limitées à leur dimension. Les travaux de Dischler *et al*. sur la décomposition de textures en "particules", éléments visuels élémentaires, et l'analyse de leurs placements relatifs constituent des pistes de réflexion très intéressantes, fortement liées à nos travaux [DMLC02]. Cependant, l'extraction effective desdites particules nécessite l'assistance de l'utilisateur. Enfin, aucune de ces techniques ne visent à l'analyse des éventuels recouvrements mutuels entre les structures visibles de l'échantillon.

Synthèse par optimisation : La plupart des méthodes citées jusqu'à présent effectuent la minimisation gloutonne de mesures locales de similarité. Ainsi elles risquent donc la convergence vers des minima locaux. Afin de surmonter cette difficulté, Kwatra propose une technique non causale de synthèse par minimisation itérative d'une énergie de dissimilarité entre l'échantillon et la sortie en cours de création [KEBK05]. Les résultats proposés sont d'une grande qualité mais l'énergie employée, bien qu'évaluée globalement, consiste en la sommation de distances visuelles entre voisinages de pixels. Ainsi les structures sont certes mieux préservées mais ne sont pas explicitement obtenues, compromettant ainsi l'analyse de leur disposition et recouvrements.

Textures quasi-régulières : Une famille de méthodes, traitant spécifiquement des *near-regular textures*, parvient à s'affranchir des contraintes d'une analyse pixellique de l'échantillon. Usant de l'hypothèse selon laquelle l'exemple est composé de tuiles élementaires disposées selon un des dix-sept possibles pavages du plan, elles cherchent à déterminer la structure du treillis sous-jacent par l'analyse d'auto-corrélations locales [LCT04]. Susceptibles de nécessiter l'intervention de l'utilisateur lors une phase corrective pour gérer les déformations [LLH04], ces techniques génèrent de très bons résultats une fois le pavage détecté, mais ne sont guère transposables aux textures arborant des distributions quelconques. Notre approche vise notamment à la levée de cette limitation tout en demeurant automatique.

2.2. Analyse épitomique

Théorie issue de la vision par ordinateur pour encoder l'apparence d'images pour leur catégorisation ou segmentation, l'analyse épitomique connaît un récent gain d'intérêt en infographie [WWOH08, WHZ*08]. Elle consiste en la factorisation d'éléments répétitifs jugés redondants, tout en contrôlant la perte d'information [JFK03, KWR06]. Seuls quelques patches représentatifs de pans entiers de l'image sont stockés pour constituer l'*épitome* de celle-ci, ainsi que les transformations permettant sa reconstruction. La compression peut également allier efficacité et lisibilité, si contrôlée par une mesure de similarité bi-directionnelle entre l'image et son épitome [NJ07]. La volonté de factoriser les répétitions d'éléments répétitifs s'apparente grandement à notre approche, néanmoins les épitomes obtenus ne disposent d'aucune information de structure et ne se prêtent pas à la génération de nouveaux contenus.

Il convient enfin de citer les travaux d'Ajuha visant à l'extraction automatique de textons [AT07]. Basée sur la détection de similarités entre sous-arbres obtenus par segmentation hiérarchique, la technique proposée acquiert les paramètres d'un modèle générateur de textons par apprentissage bayésien. *A contrario*, nous nous plaçons dans un cadre d'extraction de motifs entièrement non-supervisée.

3. Idée générale

L'approche présentée détecte la répétition de motifs au sein de l'échantillon et l'exploite en vue de leur identification et extraction (cf. figure 2). À cette fin, les voisinages de chacun de ses pixels sont décrits par des signatures (cf. 4.1), qui permettront d'établir des correspondances pertinentes entre différents points de l'image (cf. 4.2). Ce sont elles qui permettent la détection de répétitions par l'analyse d'accumulations dans l'espace dual des transformations reliant les paires de pixels visuellement semblables. Les similarités 2d définies par les correspondances sont réparties en classes de transformations plus ou moins importantes (cf. 4.3). On obtient alors des ensembles de pixels se répétant au sein de la texture selon une même transformation. Les motifs ne sont cependant effectivement extraits qu'au terme d'un accroissement contrôlé de région (cf. 4.4). L'analyse des recouvrements entre les formes alors obtenues est effectuée via la construction d'un graphe et la recherche de ses composantes connexes permet de réunir les motifs jusqu'alors indépendants, en classes (cf. 4.5). Enfin, dernière étape avant l'extraction des motifs finaux, le partitionnement des pixels en instances visibles valide les transformations obtenues et permet la gestion d'éventuelles occlusions (cf. 4.6).

4. Méthode proposée

4.1. Description locale de la texture

L'encodage de l'apparence des pixels de la texture est de première importance car il assure la détectabilité des répétitions et conditionne donc la réussite de l'approche. Nous faisons pour cela appel à des descripteurs locaux car ils présentent une bonne robustesse au bruit et occlusions. De nombreuses alternatives de descripteurs sont possibles pour peu que ceux-ci assurent un bon compromis entre répétitivité et discriminabilité.

Nous utilisons le descripteur SIFT (*scale invariant feature transform*) [Low99] qui garantit l'invariance par dilatation uniforme et rotation. Son calcul en un pixel nécessite la spécification d'une échelle d'intérêt, obtenue par recherche des extrema du *scale space* de l'image, ainsi qu'une orientation

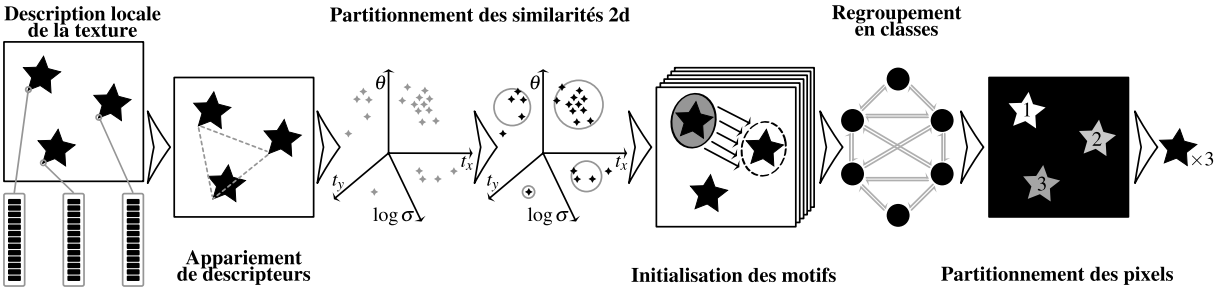


Figure 2: *La méthode proposée utilise des descripteurs locaux en chaque pixel pour déceler la présence de correspondances. Celles-ci sont ensuite agglomérées et permettent la détection d'éventuels motifs dont l'étude des recouvrements est effectuée via une analyse de graphe.*

canonique correspondant à un maximum de l'histogramme local des directions du gradient.

Une signature SIFT consiste en la concaténation de plusieurs histogrammes de directions de gradients disposés autour du point. Ces directions sont évaluées dans le repère local du pixel, contribuant à hauteur de leur magnitude aux histogrammes. Le SIFT permet ainsi de décrire de manière compacte l'information locale de forme et de texture. L'efficacité de cette représentation est d'ailleurs reconnue dans le cadre de la recherche supervisée d'objets et la catégorisation d'images [MS05].

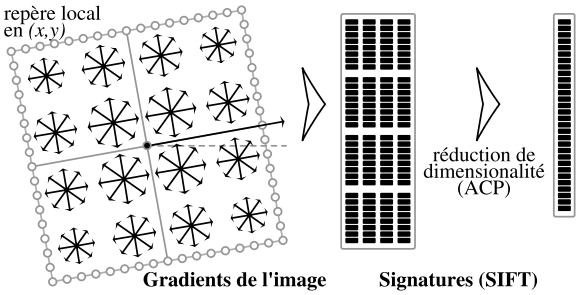


Figure 3: *Une signature SIFT est originellement de dimension 128 et consiste en la concaténation de 16 histogrammes d'orientations de gradient au sein d'un repère local. Pour améliorer le coût de notre algorithme, une réduction de dimension par ACP permet de se ramener dans un espace de moindre dimension.*

Défini par Lowe comme étant l'association de 16 histogrammes de 8 orientations, le SIFT constitue une signature de dimension 128. Nous opérons une réduction de dimensionnalité par une analyse en composantes principales (ACP) sur l'ensemble des signatures. Ceci nous permet de se ramener à des descripteurs de dimension variant entre 10 et 20, ce qui accélère grandement le reste des calculs.

4.2. Appariement de descripteurs

Chaque pixel de l'image s'étant vu attribuer une description de la région qui l'entoure, on peut alors établir des correspondances entre pixels, en assimilant similarité visuelle et distance euclidienne entre leur vecteur de caractéristiques. Afin d'accélérer l'établissement des signatures les plus proches pour chaque pixel, nous utilisons un *kd-tree* et procédons à une recherche de rayon fixe autour de la signature considérée. Dans l'ensemble de nos exemples, le rayon de recherche en termes de SIFT normalisés est de 0.15.

Puisque calculés en chaque pixel, il arrive souvent que les SIFT établissent des correspondances avec des pixels adjacents. Afin de limiter ce phénomène et juguler l'explosion des temps de calcul qui en découle, on procède à l'agglomération des points proches pour ne garder que la correspondance établissant la distance minimale. En outre, afin de supprimer les éventuels appariements ambigus, ne sont gardées que les correspondances associées aux points les plus sélectifs (2-quantile inférieur). L'ensemble des correspondances conservées peut enfin faire l'objet d'un échantillonnage uniforme si nécessaire, celui-ci ne remettant pas fondamentalement en cause la distribution des transformations induites.

4.3. Partitionnement des similarités 2d

Chacune des correspondances entre pixels définit explicitement une similarité 2d au sein de l'image et atteste ainsi de la présence d'un patch existant en deux exemplaires. L'idée fondamentale ici est de considérer ces transformations comme des votes et d'en rechercher les plus plébiscitées. L'analyse de ces votes permettra ainsi de détecter des ensembles épars de pixels suivant approximativement une même transformation.

Les transformations sont réparties en classes par *mean shift clustering* [CM02], méthode itérative de partitionnement non supervisée dont l'avantage est de n'imposer aucun *a priori* sur le nombre ou la forme des classes recherchées. Ce partitionnement, effectué dans l'espace des transformations, s'opère en six dimensions : quatre pour les transfor-

mations (translations, rotation et dilatation), les deux autres pour assurer la localité dans l'image des groupes en cours de formation (positions des points sources des correspondances).

Chaque transformation est considérée comme l'échantillon d'une fonction de densité inconnue dont on cherche les maxima locaux. Chaque point se déplace vers une moyenne pondérée calculée localement jusqu'à convergence à un point stationnaire appelé *mode*. L'ensemble des échantillons ayant convergé vers un même mode forme une classe car appartenant au "bassin d'attraction" de ce dernier.

Les noyaux gaussiens utilisés pour calculer la moyenne locale à chaque itération permettent une paramétrisation intuitive via le contrôle de leur déviation standard et autorisent le traitement approprié de chaque dimension, selon qu'il s'agisse une moyenne arithmétique (translations et position), géométrique (dilatation) ou éventuellement angulaire (rotation).

Les cardinaux des groupes obtenus au terme du partitionnement permettent d'évaluer la pertinence des transformations établies. Un simple test suffit à retirer le cas trivial de la transformation identité naturellement majoritaire. Les groupes restants décrivent alors des ensembles de pixels spatialement proches dont on peut affirmer qu'ils suivent une même transformation notée $\overline{\tau}$ pour se répéter dans l'image.

4.4. Initialisation des motifs

Ce sont ces associations de pixels qui vont guider le processus d'initialisation des motifs. Nous disposons à présent d'indices permettant de suspecter la présence de motifs se répétant. Il nous faut à présent, à partir de ces groupes de points dispersés, obtenir des formes qui, à défaut d'être parfaitement connexes, permettront une re-synthèse de la texture d'entrée.

Chaque amas de transformations dont le cardinal est estimé suffisant (au moins égal au quart de la taille du groupe le plus important) est étudié et devient dès lors susceptible de générer un ou plusieurs motifs au sein de la texture. L'enjeu est de parvenir à étendre continûment le domaine de leur transformation, à partir des points sources des correspondances du groupe, à une zone de l'image.

4.4.1. Conditions d'acceptation des pixels

Nous utilisons un algorithme d'accroissement de régions à partir des groupes de points partageant une même transformation. Les pixels devenant adjacents à la zone en cours de création $\mathscr{P}$ y sont incorporés à deux conditions :

– que leur voisinage respectif soit compatible avec la transformation du groupe,
– que leur ajout ne provoque pas de recouvrement entre la zone avant et après transformation.

La première condition se résume à vérifier que la distance euclidienne des SIFT , avant et après application de la transformation, reste en dessous d'un seuil. Par défaut, il a été choisi comme une fois et demi le rayon utilisé pour l'appariement au moment du parcours du *kd-tree*.

Le succès de cette opération dépend en outre de la qualité de la transformation dont on essaie d'accroître le champ d'action. Nous avons pris le parti de ne pas directement s'appuyer sur les correspondances contenues au sein des groupes tant leur variabilité et leur caractère approximatif nuiraient aux résultats. *A contrario* les similarités associées à chacune des classes au terme du *mean shift* sont très pertinentes et robustes. Mais décrire la transformation subie par le motif supposé via une unique similarité rigide est bien trop restrictif et peu compatible avec des images réelles.

C'est pourquoi, afin d'obtenir plus de flexibilité tout en limitant l'influence de possibles correspondances erronées, nous utilisons des splines "plaque-mince" (ou *thin plate splines*) [Boo89], volontairement approximantes, pour extrapoler à l'ensemble des pixels de l'image les valeurs de translation, rotation et dilatation que les correspondances les plus proches de la similarité du groupe $\overline{\tau}$ spécifient en leur point source. On obtient alors une transformation non rigide, localement égale à une similarité, dont l'évolution spatiale des degrés de liberté est décrite par des fonctions à base radiale.

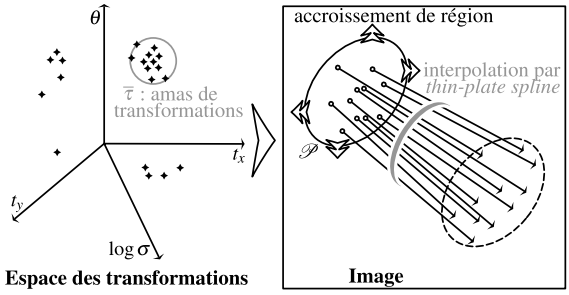


Figure 4: *La création d'un motif $\mathscr{P}$ s'effectue par un accroissement de région à partir des points sources des correspondances contenues dans une classe de transformations $\overline{\tau}$.*

4.4.2. Contrôle de l'ordre de parcours

Afin d'obtenir des motifs dont la forme est la plus simple possible, il convient d'établir des heuristiques afin de contrôler l'ordre de parcours des pixels lors de l'accroissement glouton de leur région. Pour cela, nous utilisons une file d'attente avec priorité pour prendre en considération l'aspect du motif en cours d'initialisation. Pour tous les pixels susceptibles d'être ajoutés (entrant en contact avec le motif), nous mesurons leur impact sur celui-ci en évaluant l'évolution potentielle du rapport du carré de son périmètre sur son aire. Cette mesure permet en effet de quantifier la simplicité du contour de la région. Pour privilégier les formes régulières,

une priorité accrue est donnée aux candidats d'impact minimal et, en cas d'équipriorité, les pixels les plus proches des points sources à l'origine du motif seront traités les premiers.

4.4.3. Traitement aux frontières des motifs

L'inconvénient d'utiliser le SIFT comme mesure de validité de la transformation en un point est qu'au moment où les bords effectifs d'un motif entrent dans le voisinage utilisé pour son calcul, la signature risque d'encoder une partie de l'arrière-plan et donc être invalide. Afin de capturer les pixels aux bords des motifs, nous effectuons, en cas de dépassement du seuil d'acceptation, la comparaison entre des versions simplifiées du SIFT, en diminuant sa dimensionnalité ainsi que le rayon du voisinage sur lequel il est évalué.

Ainsi d'une signature 16×8 initialement évaluée sur un voisinage 17×17, nous utilisons des descripteurs 4×6, puis 1×4 encodant les gradients d'une zone 9×9 et 5×5 respectivement. Cependant l'utilisation de ces versions plus permissives du SIFT doivent uniquement permettre l'acceptation de pixels aux bords des motifs, la progression de l'accroissement de région est alors limitée à un nombre restreint de pas autorisés. Ceux-ci doivent suffire à combler l'espace ne pouvant être validé par un SIFT entier en raison de la capture d'éléments avoisinants.

4.5. Regroupement en classes

À présent nous disposons non plus de groupes de points seulement, mais de couples $(\{\mathscr{P}_i, \overline{\tau}_i\})_i$ composés d'une zone continue de l'image $\mathscr{P}_i$ se répétant via une similarité 2d $\overline{\tau}_i$. Ces couples constituent autant de motifs candidats dont nous allons évaluer l'utilisabilité et que nous tâcherons de factoriser en différentes classes en vue de décrire la texture d'entrée le plus concisément possible.

4.5.1. Création du graphe de recouvrements

En raison de l'impossibilité d'utiliser directement les correspondances au moment de leur initialisation, nous savons que les zones obtenues certes se répètent *quelque part* dans l'image, mais pas directement sur quelle autre forme. Ainsi, deux instances d'un même motif peuvent avoir engendré deux couples non encore associés. De même, plusieurs formes, encore indépendantes, sont susceptibles de délimiter une même zone de l'image pour se projeter selon des transformations distinctes.

Pour rétablir les correspondances perdues et traiter la redondance d'information, nous utilisons le *recouvrement* entre les différents couples disponibles. Nous encodons cette information au sein d'un graphe, dont les sommets sont les différentes zones extraites de l'image et les arêtes les transformations susceptibles de les relier.

Les arêtes, évaluées entre chaque forme avant et après transformation, sont orientées mais existent toujours dans les deux sens. Une demi-arête orientée d'extrémités initiale $\mathscr{P}_i$ et extrémale $\mathscr{P}_j$ spécifie, outre la transformation reliant ses deux nœuds τ_{ij}, sa pertinence quantifiée par le recouvrement relatif qu'elle entraîne entre ses sommets, noté $r_{ij} \in [0,1]$. Il est ainsi aisé de savoir, pour chacune des formes obtenues, quelles sont celles qui, transformées ou non, l'intersectent et en quelle proportion.

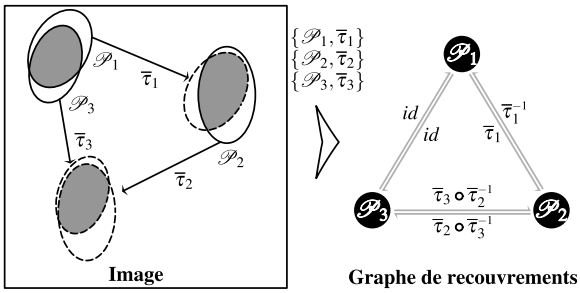


Figure 5: *Exemple d'initialisation d'un graphe de recouvrement entre trois couples $\{\mathscr{P}_i, \overline{\tau}_i\}_{i=\{1,2,3\}}$. Pour chacun d'eux, la région de départ est en trait plein, et la région d'arrivée est en pointillés. Différentes configurations peuvent alors apparaître et nécessiter un traitement adapté à cause de l'asymétrie des transformations.*

Remarquons par ailleurs, qu'étant donnés deux couples $\{\mathscr{P}_i, \overline{\tau}_i\}$ et $\{\mathscr{P}_j, \overline{\tau}_j\}$, quatre alternatives existent quant aux deux demi-arêtes créées $\{\tau_{ij}, r_{ij}\}$ et $\{\tau_{ji}, r_{ji}\}$:

τ_{ij}	r_{ij}	τ_{ji}	r_{ji}
id	$\frac{\lvert\mathscr{P}_i \cap \mathscr{P}_j\rvert}{\lvert\mathscr{P}_j\rvert}$	id	$\frac{\lvert\mathscr{P}_i \cap \mathscr{P}_j\rvert}{\lvert\mathscr{P}_i\rvert}$
$\overline{\tau}_i$	$\frac{\lvert\overline{\tau}_i(\mathscr{P}_i) \cap \mathscr{P}_j\rvert}{\lvert\mathscr{P}_j\rvert}$	$\overline{\tau}_i^{-1}$	$\frac{\lvert\overline{\tau}_i(\mathscr{P}_i) \cap \mathscr{P}_j\rvert}{\lvert\mathscr{P}_i\rvert}$
$\overline{\tau}_j^{-1}$	$\frac{\lvert\mathscr{P}_i \cap \overline{\tau}_j(\mathscr{P}_j)\rvert}{\lvert\mathscr{P}_j\rvert}$	$\overline{\tau}_j$	$\frac{\lvert\mathscr{P}_i \cap \overline{\tau}_j(\mathscr{P}_j)\rvert}{\lvert\mathscr{P}_i\rvert}$
$\overline{\tau}_i \circ \overline{\tau}_j^{-1}$	$\frac{\lvert\overline{\tau}_i(\mathscr{P}_i) \cap \overline{\tau}_j(\mathscr{P}_j)\rvert}{\lvert\mathscr{P}_j\rvert}$	$\overline{\tau}_j \circ \overline{\tau}_i^{-1}$	$\frac{\lvert\overline{\tau}_i(\mathscr{P}_i) \cap \overline{\tau}_j(\mathscr{P}_j)\rvert}{\lvert\mathscr{P}_i\rvert}$

Plus qu'un simple dénombrement de pixels, les recouvrements calculés consistent en la sommation (notée ici $|.|$) des valeurs d'une fonction de pondération évaluée en chaque pixel, dont le rôle est de rendre compte de l'activité locale de l'image. La fonction utilisée ici est la variance estimée sur un voisinage 5×5 autour du point d'évaluation dans l'espace de couleurs *CIELa*b**.

Dans un premier temps, toutes les possibilités sont étudiées puis seules les arêtes correspondant à une intersection maximale sont retenues entre chaque couple de motifs.

4.5.2. Calcul des composantes connexes du graphe

Reste maintenant à parcourir le graphe de recouvrements afin de grouper ses sommets en classes. Pour ce faire, iden-

tifier les paires d'arêtes correspondant à des recouvrements mutuels pertinents est fondamental. Ainsi une paire de demi-arêtes $\{\tau_{ij}, r_{ij}\}$ et $\{\tau_{ji}, r_{ji}\}$ sera considérée comme décrivant un recouvrement mutuel fort si $\min\{r_{ij}, r_{ji}\} > r_{\text{low}}$ et $\max\{r_{ij}, r_{ji}\} > r_{\text{high}}$ (avec dans la totalité de nos exemples r_{low} et r_{high} deux seuils respectivement égaux à 0.5 et 0.75).

La présence de deux seuils s'explique par la nécessité de prendre en compte au moment du regroupement en classes, certes les recouvrements entre formes à une transformation près, mais également les tailles relatives de celles-ci. Leur choix est directement lié à la capture des éventuelles occlusions entre les motifs de l'échantillon. En effet, si abaissé, r_{low} rend plus facile l'accession à une classe à des formes de moindre taille se projetant –partiellement– sur un de ses membres.

On procède alors à une analyse en composantes connexes sur les sommets du graphe. Deux sommets sont considérés compatibles et regroupés au sein d'un même sous-graphe s'ils sont tous deux reliés par deux demi-arêtes de fort recouvrement et si les transformations associées n'entraînent pas de contradiction avec le reste de la composante. En effet, ce sont ces mêmes composantes qui, à terme, constitueront nos classes de motifs. Il est donc fondamental de s'assurer au moment de leur calcul de l'absence de cycles incohérents et de propager les contraintes de transformation que chaque nouveau couple de demi-arêtes impose.

Une fois l'ensemble des sommets du graphe répartis en différentes composantes disjointes, chacune d'elles regroupe les différentes formes d'une classe de motifs ainsi que les transformations (potentiellement l'identité) entre elles. Parmi les différents motifs d'une classe, celui de taille maximale en nombre de pixels est choisi comme référence. On déduit les transformations maximisant les recouvrements à partir de cette dernière via un algorithme de calcul de chemin de coût maximal. Enfin, le sous-graphe correspondant à la classe peut être rendu complet en inférant les éventuelles transformations manquantes et être utilisé afin de partitionner les pixels de l'image et de reconstruire le motif représentant la classe.

4.6. Partitionnement des pixels

Seules les classes d'au moins deux membres sont retenues pour la suite des traitements. Ceux-ci consistent à attribuer à chacun des pixels de l'image, une classe unique, puis un de ses membres si un choix s'avère nécessaire. Une fois ce nouveau partitionnement achevé, il sera alors possible d'extraire un motif référent pour chacune des classes, au terme d'une éventuelle reconstruction dans le cas d'occlusions entre les différentes instances.

4.6.1. Classes en tant qu'ensembles de membres

Sous-graphes complets extraits du graphe de recouvrement, les classes de motifs contiennent à la fois, un ensemble de motifs via leurs nœuds, et l'intégralité des transformations les reliant via leurs arêtes. Par construction, ces transformations sont toutes garanties cohérentes entre elles et l'ensemble des chemins reliant deux nœuds distincts décrivent, après composition de leurs arêtes, la même similarité 2d. Néanmoins, il peut exister au sein de ces sous-graphes des arêtes dont la transformation correspond à l'identité (cf. figure 5). Pour la suite des calculs, nous fusionnons les nœuds séparés par ces arêtes et procédons à l'union booléenne de leur forme associée afin d'obtenir les différents *membres* de la classe considérée. Chaque classe est ainsi dorénavant composée de membres, chacun lié aux autres par une transformation non triviale.

4.6.2. Répartition des pixels en configurations

Il est possible qu'un ou plusieurs membres, d'une même classe ou non, soient susceptibles d'apparaître à l'emplacement d'un même pixel. C'est cette ambiguïté qu'il nous reste à lever afin de fixer l'emplacement des membres de classe et de procéder à l'extraction des motifs référents. Pour faciliter les calculs, les pixels sont tout d'abord regroupés en *configurations*. Ceci consiste à réunir les pixels de l'image partageant le même panel de choix quant aux membres de classes pouvant leur être attribués. Il est alors possible non plus de traiter chacun des pixels séparément, mais plutôt de raisonner en termes ensemblistes, au niveau des configurations.

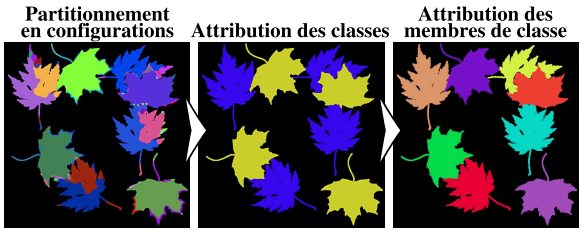


Figure 6: *Les pixels sont d'abord regroupés en configurations correspondant à des ensembles de choix de membres possibles. Chacune se voit ensuite assigner une classe, puis un de ses membres. Cette décomposition du partitionnement en deux étapes permet la gestion des cas où deux membres d'une même classe sont directement en contact.*

4.6.3. Assignation d'une classe par configuration

Dans un premier temps, nous assignons à chacune des configurations une classe unique. Il est possible que certaines d'entre elles voient leurs choix de membres confinés aux membres d'une seule et même classe, ne laissant plus de doute quant à la classe à leur attribuer. Ces éventuelles configurations permettent ensuite l'initialisation d'un processus itératif de propagation de l'information de classe.

Cette étape vise à utiliser les configurations sûres de leur classe pour inférer celle d'autres configurations restées en suspens. Une configuration peut voir sa classe ainsi déduite

si, en appliquant les transformations partant de ses membres candidats, les pixels de la configuration considérée se projettent sur des configurations de même classe que le membre en question, et ce sans ambiguïté ni conflit avec d'autres classes. Le processus est répété tant que le nombre de configurations indécises diminue.

Pour les configurations restées libres au terme de la propagation, on évalue pour chacune de leurs classes candidates un score, estimé pour chaque pixel puis moyenné, et la classe candidate de score maximal remporte la configuration. Les scores de classe quantifient le nombre moyen de fois où les pixels de la configuration, considérés comme appartenant aux différents membres possibles, une fois projetés via leurs transformations associées, sont susceptibles de demeurer dans la même classe candidate.

Les scores sont arrondis au plus proche entier et si plusieurs classes candidates partagent un même score maximal, la configuration choisit parmi les classes arrivées en tête en consultant d'éventuelles configurations adjacentes dont la classe a été déterminée. Si l'hésitation persiste, la classe disposant de la taille maximale déjà allouée hérite de la configuration.

4.6.4. Assignation d'un membre par configuration

L'assignation d'un membre pour chacune des configurations suit exactement le même cheminement : les configurations n'offrant aucune hésitation quant au choix de leur membre sont utilisées pour contraindre la propagation de l'information aux autres configurations. Les modalités de propagation sont néanmoins légèrement différentes : elles doivent ici assurer que deux configurations de même classe, séparées par une transformation, doivent appartenir à deux membres distincts.

Une fois la propagation achevée, le choix du membre des configurations demeurées libres se base à nouveau sur une maximisation de scores. Les scores de membre quantifient, selon à quel membre les pixels sont supposés appartenir, leur "comportement" au sein de la classe. Les transformations vers les autres membres de la classe lui sont appliquées et on comptabilise le nombre de membres de classe distincts qu'il parvient à atteindre après projection. L'évaluation des différents scores de membre, ainsi que la gestion des possibles indécisions, s'effectuent de la même manière que pour les classes.

Nous obtenons ainsi la carte des membres de classe effectivement visibles et pouvons nous baser sur ce partitionnement pour extraire un motif référent par classe. Ceux-ci sont obtenus en parcourant les pixels des différents membres, un nouveau pixel étant ajouté à l'instance référente s'il apparaît dans au moins deux membres de la classe à une transformation près.

5. Résultats et discussion

Une fois l'analyse de l'échantillon effectuée et les formes qui le composent identifiées, il est très aisé de synthétiser de nouvelles images.

Les figures 1 et 7 présentent le résultat de l'analyse de trois textures où des objets sont reconnaissables. Grâce aux classes de motifs détectés, nous regénérons de nouvelles textures à partir d'une distribution de Poisson obtenue par relaxation de Llyod et plaçons en chacun des points ainsi disposés une des images extraites selon une orientation aléatoire.

Bien évidemment, il s'agit là d'un exemple très simpliste d'analyse et d'autres mesures plus pertinentes telles l'établissement de statistiques sur les voisinages de formes obtenus par triangulation de Delaunay pourraient être utilisées. La recherche naissante en rendu expressif sur la création d'arrangements de primitives vectorisées constituent en cela des pistes intéressantes [BBT*06, IMIM08].

5.1. Performances et implémentation

La méthode présentée est de complexité en $O(N^4)$, N désignant le nombre de signatures établies dans l'échantillon. Nous ne procédons à aucune détection de points d'intérêt avant description des pixels. Nous ne conservons cependant seulement les descripteurs dont les magnitudes avant normalisation sont jugées significatives. Cette condition de discriminabilité permet d'accélérer grandement les temps de calcul sans perte notable de qualité.

Il est également utile de préciser que l'ensemble des étapes proposées, hormis le partitionnement final des pixels, se prête aisément à une implémentation en parallèle. Ainsi, les temps de calcul pour les différents résultats obtenus sont de 3 minutes pour la figure 1, et de 6 minutes pour les textures de la figure 7, performances établies sur une machine *quad-core Intel(R)* 64 bits. Enfin, le partitionnement 6d des similarités est effectué dans un espace voxélisé pour permettre la recherche de transformations proches en temps constant.

5.2. Comparaison avec les travaux précédents

La figure 8 présente les résultats obtenus par deux techniques complémentaires : la synthèse de textures par échantillonnage non paramétrique [Ash01], et par *graph cuts* [KSE*03]. La première œuvre uniquement à l'échelle des pixels et s'efforce pour chaque pixel résultat, de trouver dans l'échantillon d'origine un voisinage causal visuellement similaire. La seconde crée une nouvelle texture en plaçant dans un premier temps des patches se recouvrant partiellement et définit ensuite les frontières au sein des parties communes afin de contrôler l'impact visuel des limites entre patches, sources d'artefacts.

Chacune de ces deux techniques s'évertue et parvient

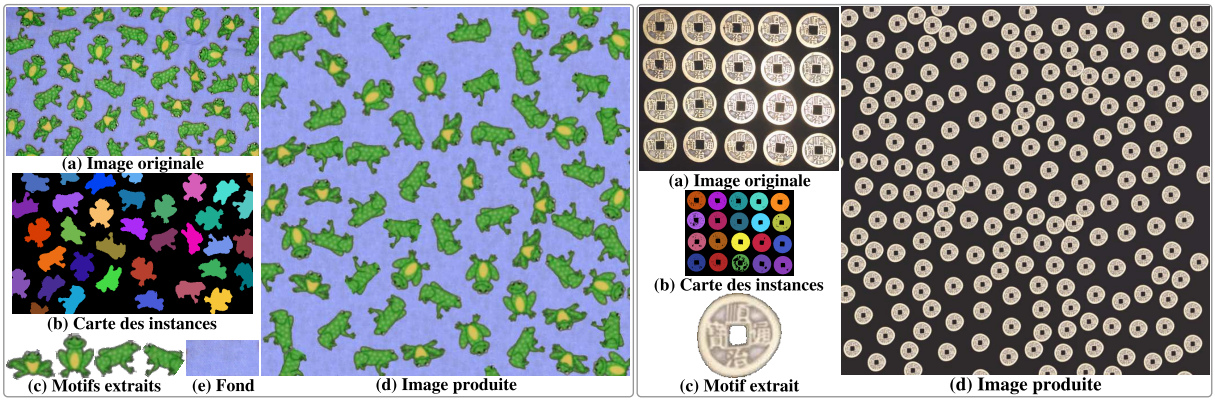


Figure 7: *Deux applications de notre méthode : à chaque fois, les classes d'objets (c) ainsi que la position de leurs instances dans l'image (b) sont calculées automatiquement. Les données obtenues peuvent alors être réutilisées pour produire de nouvelles images (d). Dans le cas des grenouilles, l'utilisateur sélectionne manuellement une partie du fond (e), ensuite utilisé pour remplir l'image produite à l'aide d'une méthode non paramétrique [EL99]. L'utilisation de transformations non rigides est ici essentielle pour correctement capturer les motifs.*

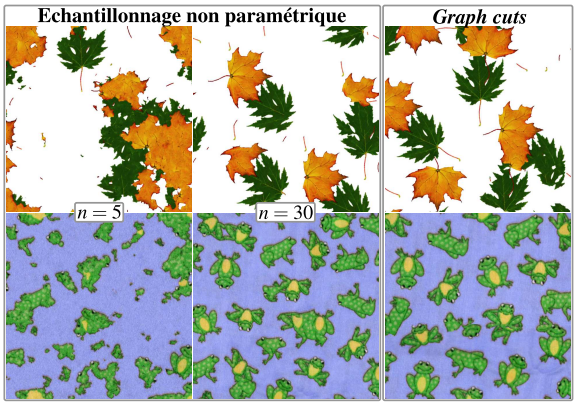


Figure 8: *Ni l'analyse des voisinages des pixels , ni l'apposition sans coutures de patches ne parviennent à fournir des résultats pleinement satisfaisants car toutes deux n'évoluent pas à une échelle appropriée.*

à préserver localement la cohérence des résultats. Néanmoins, ceux-ci ne sont clairement pas satisfaisants : les quelques feuilles encore discernables sont des agglomérats ayant perdu leur forme originelle et des éléments fins partiellement conservés, tels les tiges, donnent lieu à des éléments visuels absents de l'entrée. Une solution possible est d'augmenter arbitrairement la taille du voisinage dans le cadre de l'analyse par pixel et ainsi favoriser la re-synthèse de structures plus importantes. Or un tel recours risque d'introduire des répétitions en raison du nombre réduit de voisinages candidats et ne constitue pas une solution en soit.

5.3. Contexte d'analyse et limitations

Il est évident que notre méthode ne s'applique qu'à une classe spécifique de textures – les arrangements de primitives se répétant à une similarité près – et ne conviendrait à des textures de stochasticité extrême, telles les textures de bruit, où l'identification de répétitions n'a pas de sens. Cette question se pose en outre pour la gestion du *fond* de nos arrangements. Défini par opposition aux instances de motifs détectées, le fond est extrait, voit l'emplacement des motifs comblé et est éventuellement agrandi via une méthode de synthèse par échantillonnage non paramétrique.

Notre approche n'utilise aucune connaissance *a priori* quant aux formes répétitives recherchées, mais se base exclusivement sur les répétitions au sein de l'échantillon d'entrée. Ainsi seules des formes apparaissant au moins deux fois peuvent être validées et donc extraites. Cette condition certes pose une limite intrinsèque à notre méthode mais est le prix à payer pour se passer de l'ajout d'informations d'ordre sémantique. Elle peut donc entraîner l'obtention de résultats peu intuitifs. Ainsi des motifs subissant les mêmes occlusions ou dont l'intégralité n'apparaît pas explicitement dans l'échantillon seront apparemment extraits "incomplets".

Enfin la qualité de nos résultats découle directement de la robustesse des signatures utilisées pour la description des pixels de la texture. Faisant l'objet de recherches en tant que domaine à part entière, la reconnaissance d'images par description de leur contenu connaît de grandes avancées qui pourraient aisément être mises à profit dans notre approche.

6. Conclusion et travaux futurs

Nous avons présenté une nouvelle méthode visant à l'analyse automatique de textures consistant en une distribution

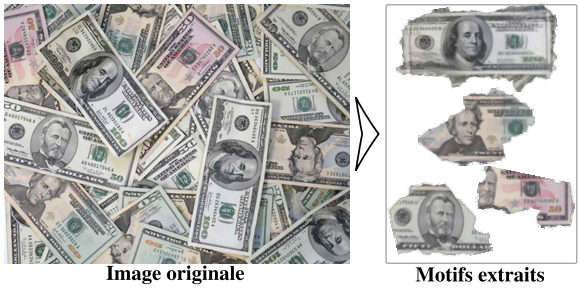


Figure 9: *Dans cet exemple les quatre types de billet ont bien été identifiés et leur extraction a su fusionner toutes les sous-parties apparaissant au moins deux fois dans l'image. Les parties manquantes des billets sont celles qui n'apparaissent qu'une seule fois dans l'image et qui sont donc non identifiables par notre algorithme sans connaissances* a priori*. L'échantillon est d'autant plus difficile à traiter qu'il présente un nombre important d'occlusions, que les instances sont légèrement déformées les unes par rapport aux autres et que l'information de couleur est source d'incertitude.*

aléatoire de motifs. L'originalité de notre approche vis-à-vis des travaux précédents en synthèse de textures est la détection des répétitions pour guider l'extraction desdits motifs. Celles-ci sont établies par appariements de descripteurs locaux, suivis de l'agglomération des transformations induites. Les motifs sont obtenus par accroissement de régions et groupés en classes grâce à l'analyse via un graphe de leurs recouvrements mutuels.

Ainsi les textures étudiées peuvent être explicitement décrites comme des collections de motifs, chacun soumis à un ensemble de transformations. Une telle représentation pour ces textures est particulièrement intéressante pour leur compression, leur re-synthèse ainsi que leur édition haut-niveau. Afin d'élargir le spectre des applications de ce travail, il est possible d'effectuer des statistiques plus poussées sur les distributions d'objets. En particulier, à l'image des travaux de Yanxi Liu [LLH04], il est pertinent d'opérer à présent une ACP sur la distribution des couleurs entre instances d'une même classe pour capturer les variations subtiles d'apparence. De même, étudier les voisinages établis par triangulation de Delaunay, ou mesurer la répartition des objets à la recherche de régularités constituent des pistes de recherche futures intéressantes.

Références

[Ash01] ASHIKHMIN M. : Synthesizing natural textures. In *I3D '01* (2001), pp. 217–226.

[AT07] AHUJA N., TODOROVIC S. : Extracting texels in 2.1d natural textures. In *ICCV '07* (2007), pp. 1–8.

[BBT*06] BARLA P., BRESLAV S., THOLLOT J., SILLION F., MARKOSIAN L. : Stroke pattern analysis and synthesis. In *Eurographics '06* (2006), vol. 25, pp. 663–671.

[BD02] BROOKS S., DODGSON N. : Self-similarity based texture editing. In *SIGGRAPH '02* (2002), pp. 653–656.

[Bon97] BONET J. S. D. : Multiresolution sampling procedure for analysis and synthesis of texture images. In *SIGGRAPH '97* (1997), pp. 361–368.

[Boo89] BOOKSTEIN F. : Principal warps : thin-plate splines and the decomposition of deformations. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1989), vol. 11, pp. 567–585.

[CM02] COMANICIU D., MEER P. : Mean shift : a robust approach toward feature space analysis. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2002), vol. 24, pp. 603–619.

[DMLC02] DICHLER J.-M., MARITAUD K., LÉVY B., CHAZANFARPOUR D. : Texture particles. In *Eurographics '02* (2002), pp. 401–410.

[EF01] EFROS A. A., FREEMAN W. T. : Image quilting for texture synthesis and transfer. In *SIGGRAPH '01* (2001), pp. 341–346.

[EL99] EFROS A. A., LEUNG T. K. : Texture synthesis by non-parametric sampling. In *ICCV '99* (1999), pp. 1033–1038.

[HB95] HEEGER D. J., BERGEN J. R. : Pyramid-based texture analysis/synthesis. In *SIGGRAPH '95* (1995), pp. 229–238.

[IMIM08] IJIRI T., MECH R., IGARASHI T., MILLER G. : An example-based procedural system for element arrangement. In *Eurographics '08* (2008), vol. 27, pp. 429–436.

[JFK03] JOJIC N., FREY B., KANNAN A. : Epitomic analysis of appearance and shape. In *ICCV '03* (2003), pp. 34–41.

[KEBK05] KWATRA V., ESSA I., BOBICK A., KWATRA N. : Texture optimization for example-based synthesis. In *SIGGRAPH '05* (2005), vol. 24, pp. 795–802.

[KSE*03] KWATRA V., SCHÖDL A., ESSA I., TURK G., BOBICK A. : Graphcut textures : image and video synthesis using graph cuts. In *SIGGRAPH '03* (2003), vol. 22, pp. 277–286.

[KWR06] KANNAN A., WINN J., ROTHER C. : Clustering appearance and shape by learning jigsaws. In *NIPS'06* (2006).

[LCT04] LIU Y., COLLINS R. T., TSIN Y. : A computational model for periodic pattern perception based on frieze and wallpaper groups. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2004), vol. 26.

[LD05] LAGAE A., DUTRÉ P. : A procedural object distribution function. *ACM Transactions on Graphics. Vol. 24*, Num. 4 (2005), 1442–1461.

[LH05] LEFEBVRE S., HOPPE H. : Parallel controllable texture synthesis. In *SIGGRAPH '05* (2005), vol. 24, pp. 777–786.

[LH06] LEFEBVRE S., HOPPE H. : Appearance-space texture synthesis. In *SIGGRAPH '06* (2006), vol. 25, pp. 541–548.

[LLH04] LIU Y., LIN W.-C., HAYS J. : Near-regular texture analysis and manipulation. In *SIGGRAPH '04* (2004), vol. 23, pp. 368–376.

[Low99] LOWE D. : Object recognition from local scale-invariant features. In *ICCV '99* (1999), vol. 2, pp. 1150–1157.

[MS05] MIKOLAJCZYK K., SCHMID C. : A performance evaluation of local descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2005), vol. 27, pp. 1615–1630.

[NJ07] NOWAK E., JURIE F. : Learning visual similarity measures for comparing never seen objects. In *CVPR '07* (jun 2007).

[Per85] PERLIN K. : An image synthesizer. In *SIGGRAPH '85* (1985), pp. 287–296.

[PFH00] PRAUN E., FINKELSTEIN A., HOPPE H. : Lapped textures. In *SIGGRAPH '00* (2000), pp. 465–470.

[PS00] PORTILLA J., SIMONCELLI E. P. : A parametric texture model based on joint statistics of complex wavelet coefficients. In *International Journal of Computer Vision* (2000), vol. 40, pp. 49–70.

[WHZ*08] WEI L.-Y., HAN J., ZHOU K., BAO H., GUO B., SHUM H.-Y. : Inverse texture synthesis. In *SIGGRAPH '08* (2008), pp. 1–9.

[WL00] WEI L.-Y., LEVOY M. : Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH '00* (2000), pp. 479–488.

[WL03] WEI L.-Y., LEVOY M. : Order-independent texture synthesis.

[Wor96] WORLEY S. : A cellular texture basis function. In *SIGGRAPH '96* (1996), pp. 291–294.

[WWOH08] WANG H., WEXLER Y., OFEK E., HOPPE H. : Factoring repeated content within and among images. In *SIGGRAPH '08* (2008), pp. 1–10.