Image representation and processing through multiscale local jet features

Antoine Manzanera

Image representation and processing through multiscale local jet features

Antoine Manzanera

ENSTA-ParisTech UEI
32 Bd Victor, 75739 Paris CEDEX 15, France
antoine.manzanera@ensta-paristech.fr
http://www.ensta.fr/~manzaner

Abstract. We propose a unified framework for representing and processing images using a feature space related to local similarity. We choose the multiscale and versatile local jet feature space to represent the visual data. This feature space may be reduced by vector quantisation and/or be represented by data structures enabling efficient nearest neighbours search (e.g. kd-trees). We show the interest of the local jet feature space processing through three fundamental low level tasks: noise reduction, motion estimation and background modelling/subtraction. We also show the potential of our system in terms of visual representation for higher level (e.g. object modelling and recognition) tasks.

Key words: feature space, image representation, local jet, scale space, nearest neighbour search, non local means, optical flow, background subtraction, visual salience, multidimensional mode

1 Introduction

Many problems in image processing and vision relate to visual similarity. Since the earliest processes of denoising or perceptual grouping, to the higher level tasks of object recognition, measuring the resemblance, or matching two objects according to the visual appearance are fundamental functions. In the traditional space $\times$ time representations of video sequences, the canonical distance is not related to visual similarity, which induces a major computational drawback. Indeed, similar objects from the video data are expected to interact in the processing, and then should be contiguous in the representation. These general remarks do not only apply to the current data, image or recent frames history, but also to the global visual knowledge that the vision system is constructing during its operating lifetime.

The purpose of this work is to design a global, generic and computationally tractable framework for the representation and the processing of the visual data, based on: (1) the projection of the space $\times$ time image data within a transformed space whose metrics correspond to visual similarity, (2) a set of functions operating in the transformed and/or the image domain, for extracting relevant information from the video, updating the transformed domain structure

accordingly, and/or modifying the video in the image domain according to some specific task (filtering, detecting, predicting), and (3) dedicated data structures for making such framework computationally feasible, in terms of memory and processing time. In our philosophy, such unified framework should be usable for the whole vision process, from the lowest level of regularisation and enhancement to the levels of higher semantics related to recognition and understanding. The framework should also be compliant with real-time video processing, which implies both dynamical and efficient construction of the visual representation.

The inspiring and related works are presented in Section 2. In our work the preferred similarity space is made of the collection of spatial derivatives estimated at different scales (the local jet). We present this space and justify its use in Section 3. Section 4 presents the data structure, based on binary space partition trees, used to represent the similarity space and discuss some visual models that can be directly extracted from this data structure. The following sections present the applications of the framework for different low level visual processing tasks: non-local means image denoising (Sec. 5), optical flow estimation (Sec. 6) and background subtraction based motion detection (Sec. 7).

2 Related works

Our work is closely related to Peyré's manifold models [1]. In this theoretical framework, the image data is embedded within a higher dimensional feature space and forms a manifold. It is shown that many inverse problems in low level computer vision can be expressed by regularising the manifold in the feature space and then back-projecting the transformed manifold within the image space. In this sense, the different low level algorithms proposed in this paper can be seen as instances of the manifold model. Conversely, our work can also be seen as an extension of the manifold models to higher level representations.

Our framework naturally exploits many ideas from previous works on textured objects modelling, segmentation and recognition. Filter banks have been used for a long time as a way to extract meaningful local information on direction, scale, and frequency [2]. Quantising such information is also a commonplace in textons [3] or bag of features [4] approaches. Compared with those methods, one fundamental property of our framework is that the feature is intrinsically dense in the image space and thus can be computed at any location. Another particularity of our approach is that reducing the information support to salient points is done by finding the isolated or clustered points in the feature space, thus avoiding the classical distinction which is made between detection and description of the salient points [5, 6].

The importance of the local jet in image representation has been identified a few decades ago. Koenderink and Van Doorn [7] pointed out the fundamental role of the first three orders of derivatives in the human visual system. They also noticed that some Euclidean distance on the local jet vectors could be used to perform local comparisons. Curiously, this has not been really used in the

literature. One of our contributions is then to investigate the metrical aspects of local jet feature space.

Anyway, the local jet has been used much for the construction of invariants, particularly in image retrieval [8]. It has also been used more recently for the classification of pixels according to their local geometry, see for example [9]. As shown later, we can exploit such classification approach to reduce the dimension of the local jet descriptor.

3 Multiscale Local Jet

3.1 Similarity space

Using the partial derivatives to represent the local similarity within a multidimensional signal is a natural choice since the local behaviour of any differentiable function f can be predicted from its derivatives. For a function from $\mathbb{R}^2$ to $\mathbb{R}$, at order r, the Taylor expansion provides:

$$f(\mathbf{x}+\mathbf{c}) = \sum_{k=0}^{r}\sum_{i=0}^{k}\binom{k}{i}c_1^{k-i}c_2^{i}\frac{\partial^k f}{\partial \mathbf{x}_1^{k-i}\partial \mathbf{x}_2^{i}}(\mathbf{x}) + o(||\mathbf{c}||^r) \tag{1}$$

with $\mathbf{x}_1$ and $\mathbf{x}_2$ a basis of $\mathbb{R}^2$, in which the components of the residual $\mathbf{c}$ are $c_1 = \mathbf{c}\cdot\mathbf{x}_1$ and $c_2 = \mathbf{c}\cdot\mathbf{x}_2$. To simplify we shall use from now on the notation $f_{ij} = \frac{\partial^{i+j} f}{\partial \mathbf{x}_1^i \partial \mathbf{x}_2^j}$. In digital images, the concept of derivative only makes sense up to a level of regularity corresponding to the scale of estimation [10]. Practically, this is performed by explicitly smoothing the image before differentiation, or equivalently, convolving the image with the corresponding derivative of the smoothing operator. In the Gaussian framework:

$$f_{ij}^{\sigma} = f \star \frac{\partial^{i+j} G_\sigma}{\partial \mathbf{x}_1^i \partial \mathbf{x}_2^j} \tag{2}$$

where G_σ is the 2d Gaussian function of standard deviation σ. The multiscale local jet similarity space is then given by the collection $\{f_{ij}^{\sigma}; i+j \le r, \sigma \in S\}$, where r is the order of derivation, $S = \{\sigma_1, \ldots \sigma_q\}$ the selected scales. Figure 1 illustrates the induced representation for a few points taken from a natural image, for one scale $\sigma = 1.0$. The image is split into 15×15 patches, and the reconstruction is performed on patches of the same size, using only the local jet computed at the patch centre.

It can be mentioned that the relevance of the local jet as description vector on natural images is confirmed by the fact that the first singular (or eigen) vectors that arise in SVD or PCA based decomposition of natural image patches look much like the first derivatives of a 2d Gaussian function: see for example Figure 2 taken from [11].

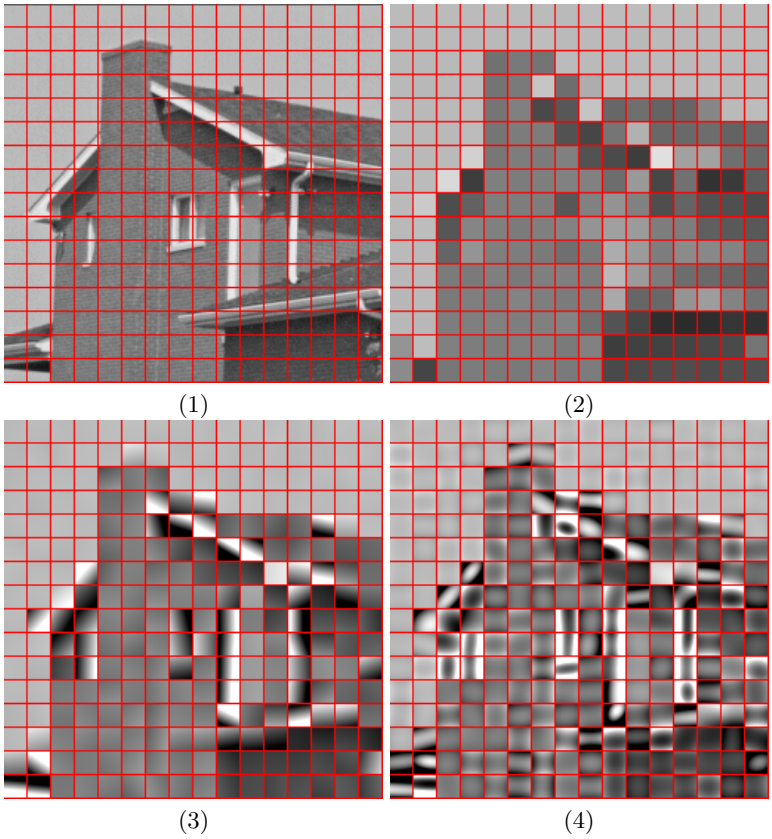

Fig. 1. Local representation by the local jet at fine scale ($\sigma = 1.0$), illustrated by "reconstructing" patches by Taylor expansion at: (2) Order 0 (1d feature) (3) Order 1 (3d feature), (4) Order 2 (6d feature)

3.2 Metrics and invariance

Now the local similarity between 2 points $\mathbf{x}$ and $\mathbf{y}$ with respect to image f has to be actually measured using a distance in the similarity space, which naturally will have an influence on the data structure (see next section).

Considering the Taylor expansion, it can be noticed that the sum of squared differences (SSD) between image patches, classically used to compute local similarity, can be approximated [7, 12] by the squared Euclidean distance between the local jet vectors. Naturally, for large and complicated patches, the approximation is very coarse, and the weights attached to every component depends on the size of the patch. However, a distance between local jet vectors is actually significant to distinguish similar pixels. So we have:

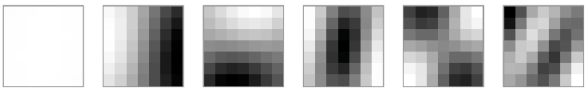

Fig. 2. 6 first singular vectors obtained from Lena image in the singular value decomposition of the 7×7 patches. (Taken from [11])

$$d_f(\mathbf{x}, \mathbf{y}) = \left(\sum_{i+j \leq r} a_{(i,j)} (f_{ij}(\mathbf{x}) - f_{ij}(\mathbf{y}))^2 \right)^{\frac{1}{2}} \tag{3}$$

Where $a_{(i,j)}$ is the coefficients attached to every derivative. Practically, those coefficients will depend on the normalisation which is done on the component of the local jet. We use the following normalised local jet components, combining the scale normalisation factor from scale space theory [10], and the number of $(i+j)$-order derivatives:

$$F_{ij}^{\sigma} = \frac{\sigma^{i+j}}{i+j+1} f_{ij}^{\sigma} \tag{4}$$

where F (resp. f) is the normalised (resp. non normalised) local jet component of order (i, j) and scale σ. Figure 3 shows some components of the normalised local jet.

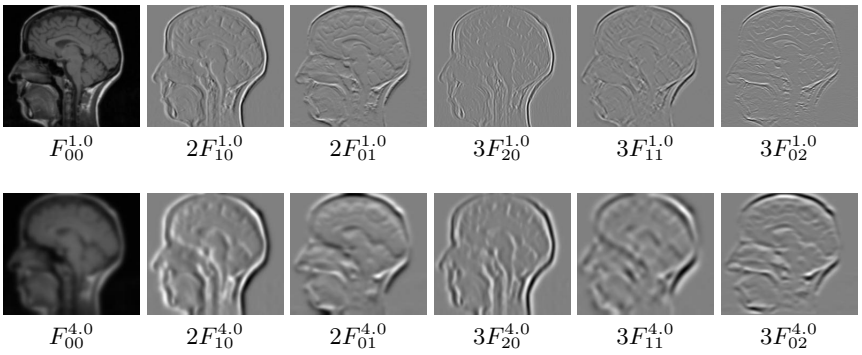

Fig. 3. Some components of the canonical local jet, for orders of derivation from 0 to 2, and two scales.

If required, rotation invariant derivatives can be obtained by expressing the derivatives within the local basis of coordinates $(\mathbf{g}, \mathbf{t})$, where $\mathbf{g} = \frac{\nabla_f}{||\nabla_f||}$ is the gradient component and $\mathbf{t}$ is the isophote component orthogonal to $\mathbf{g}$. (See Figure 4).

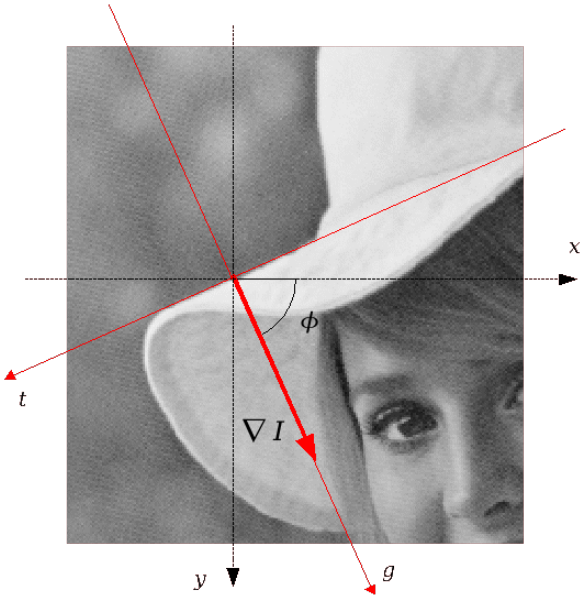

Fig. 4. Rotation invariance is achieved by computing the partial derivative with respect to the local coordinates $(\mathbf{g}, \mathbf{t})$, where $\mathbf{g}$ is the normalised gradient and $\mathbf{t}$ the isophote component.

So if $\nabla f = \binom{f_{10}}{f_{01}}$ and $H_f = \left(\begin{smallmatrix} f_{20} & f_{11} \\ f_{11} & f_{02} \end{smallmatrix}\right)$ the gradient and Hessian matrix in the canonical basis, the first (resp. second) derivative with respect to any vector $\mathbf{u}$ (resp. any couple of vectors $(\mathbf{u}, \mathbf{v})$) is $f_{\mathbf{u}} = \mathbf{u} \cdot \nabla_f$ (resp. $f_{\mathbf{uv}} = \mathbf{u}^t H_f \mathbf{v}$). Hence the rotation invariant local jet, at order two:

$$f = f_{00} \tag{5}$$
$$f_{\mathbf{g}} = (f_{10}^2 + f_{01}^2)^{1/2} \tag{6}$$
$$f_{\mathbf{t}} = 0 \tag{7}$$
$$f_{\mathbf{gg}} = (f_{20} f_{10}^2 + 2 f_{11} f_{10} f_{01} + f_{02} f_{01}^2) f_{\mathbf{g}}^{-2} \tag{8}$$
$$f_{\mathbf{tt}} = (f_{20} f_{01}^2 - 2 f_{11} f_{10} f_{01} + f_{02} f_{10}^2) f_{\mathbf{g}}^{-2} \tag{9}$$
$$f_{\mathbf{gt}} = (f_{10} f_{01} (f_{20} - f_{02}) + f_{11} (f_{01}^2 - f_{10}^2)) f_{\mathbf{g}}^{-2} \tag{10}$$

Figure 5 shows those components at two different scales.

This representation includes classical invariants: $f_{\mathbf{g}}$ is the magnitude of the gradient, $f_{\mathbf{tt}}$ is the curvature of the isophote, $f_{\mathbf{gg}} + f_{\mathbf{tt}} = f_{20} + f_{02}$ is the Laplacian, and the principal curvatures given by the eigen values of the Hessian matrix:

$$\Lambda_f = \frac{f_{20} + f_{02} + \alpha_f}{2} \; ; \; \lambda_f = \frac{f_{20} + f_{02} - \alpha_f}{2} \tag{11}$$
$$\tag{12}$$

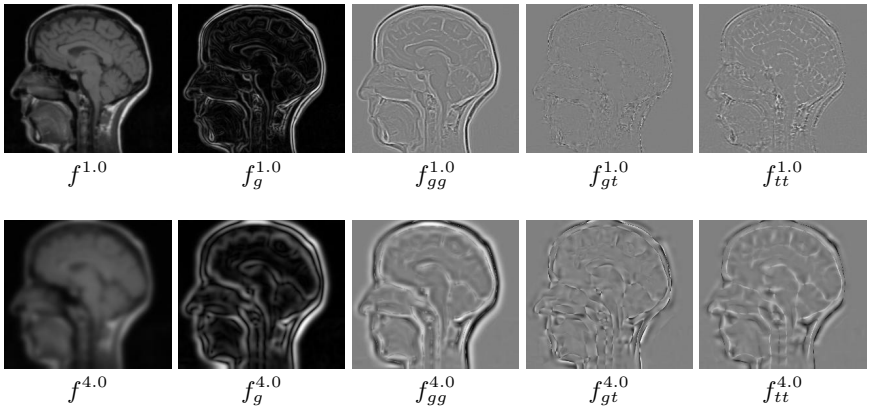

Fig. 5. The components of the rotation invariant local jet, for orders of derivation from 0 to 2, and two scales.

such that $\Lambda_f < \lambda_f$, with $\alpha_f = \sqrt{(f_{20} - f_{02})^2 + 4f_{11}^2}$, and the associated (non normalised) eigen vectors:

$$V_{\Lambda_f} = \begin{pmatrix} 2f_{11} \\ f_{02} - f_{20} + \alpha_f \end{pmatrix} ; V_{\lambda_f} = \begin{pmatrix} 2f_{11} \\ f_{02} - f_{20} - \alpha_f \end{pmatrix} \tag{13}$$

$$\tag{14}$$

The local jet also provides contrast invariant measures: at order 1, the direction of the gradient (orthogonal to the direction of the isophote), is given by $\arg \nabla f = \arctan \frac{f_{01}}{f_{10}}$. At order 2, the direction of the eigen vectors $\arg V_{\Lambda_f}$ and $\arg V_{\lambda_f}$ represent the main directions of curvature. Figure 6 shows two examples of such contrast invariant measures.

Finally, following [9], we can categorise the local behaviour of every pixel at every scale according to the dominant order of derivation. The dominant order 0 corresponds to flat or homogeneous zones, the dominant order 1 to straight contours, the dominant order 2 to elliptic or tubular curvatures depending on the signs of the eigen values of H_f. Once categorised, the dimensions of the local jet descriptor can be reduced to significant derivatives (see Figure 7). This categorisation is clearly visible in the figure 1, showing the representation of image patches by local jet of different orders.

In terms of metrics, we will typically consider three types of distances in the applications:

- the single scale distance $d_f^\sigma(\mathbf{x}, \mathbf{y})$, checking whether $\mathbf{x}$ and $\mathbf{y}$ are similar at a given scale σ.
- the pan-scalic distance $D_f^S(\mathbf{x}, \mathbf{y})$, checking whether $\mathbf{x}$ and $\mathbf{y}$ are similar for all the scales $\sigma \in S$.
- the trans-scalic distance $\delta_f^S(\mathbf{x}, \mathbf{y})$, checking whether there exists a couple of scales $(\sigma_1, \sigma_2) \in S^2$ for which $\mathbf{x}$ and $\mathbf{y}$ are similar.

(1) $\arg \nabla f$ (2) $\arg V_{\arg\max_{l\in\{\lambda_f,\lambda_f\}}|l|}$

Fig. 6. Contrast invariant local jet based components ($\sigma = 2.0$): (1) Gradient direction. (2) Main absolute curvature direction.

The single scale distance is simply defined as follows:

$$d_f^\sigma(\mathbf{x},\mathbf{y}) = \left(\sum_{i+j\leq r} (F_{ij}^\sigma(\mathbf{x}) - F_{ij}^\sigma(\mathbf{y}))^2 \right)^{\frac{1}{2}} \tag{15}$$

The pan-scalic distance can be naturally defined as:

$$D_f^S(\mathbf{x},\mathbf{y}) = \max_{\sigma\in S} \left(\sum_{i+j\leq r} (F_{ij}^\sigma(\mathbf{x}) - F_{ij}^\sigma(\mathbf{y}))^2 \right)^{\frac{1}{2}} \tag{16}$$

However for representation purposes (one single kd-tree) it can be better to define it as a Euclidean distance:

$$D_f^S(\mathbf{x},\mathbf{y}) = \left(\sum_{i+j\leq r,\sigma\in S} (F_{ij}^\sigma(\mathbf{x}) - F_{ij}^\sigma(\mathbf{y}))^2 \right)^{\frac{1}{2}} \tag{17}$$

The purpose of the trans-scalic distance is to provide scale invariance. This is done by minimising over the scales, which turns the measure to a pseudo-distance:

$$\delta_f^S(\mathbf{x},\mathbf{y}) = \min_{(\sigma_n,\sigma_m)\in S^2} \max_{p;(\sigma_{n+p},\sigma_{m+p})\in S^2} \left(\sum_{i+j\leq r} (F_{ij}^{\sigma_{n+p}}(\mathbf{x}) - F_{ij}^{\sigma_{m+p}}(\mathbf{y}))^2 \right)^{\frac{1}{2}} \tag{18}$$

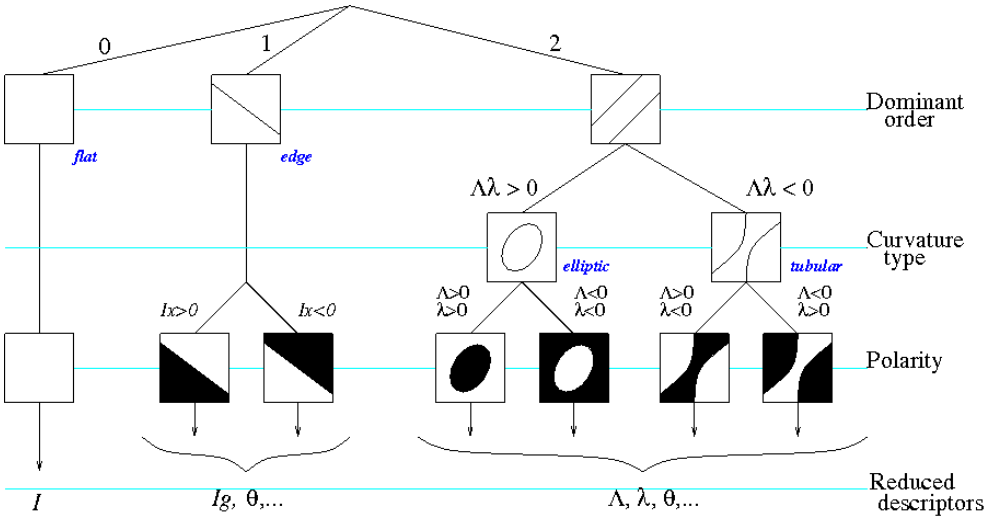

Fig. 7. Pixel categorisation for the reduction of the local jet descriptors: at order 2, pixels can be categorised into 4 categories (7 if the polarity is considered).

Figure 8 shows examples of similarity maps computed using those different distances. For a given pixel $\mathbf{x_0}$ the similarity map is defined as $M^d_{\mathbf{x_0}}(\mathbf{x}) = \Phi(d(\mathbf{x_0}, \mathbf{x}))$, with d the local jet distance, and Φ some real increasing function. In this figure $\Phi(z) = 1 - e^{-\frac{z^2}{C^2}}$, with $C = 10$. Those similarity maps show the properties of the different distances and local jet components in terms of rotation and scale invariance.

4 Visual representations

4.1 Data structures

The first step of the representation then consists in projecting the image data into the chosen similarity space. For every pixel, a feature vector is computed, and the collection of features is kept in adequate data structure for further processing. If the feature space dimensionality is low, the data structure may be a simple array, whose coordinates are indexed by each component of the feature space, which must then be quantised properly. For the purpose of further processing in the image domain, every feature vector must be attached a pointer to the original pixel. Depending on the quantisation, several pixels may point on the same feature vector, the feature array must then be able to code a list rather than a single element in every bucket. Such data structure finally coincides with an hash table whose hash function is the quotient of the quantisation.

But whatever the quantisation, the size of the array structure obviously grows exponentially with the dimension, so the memory cost rapidly becomes red-

hibitory. For higher dimension, then, one much better solution is to use a binary space partition tree, which is optimal in terms of memory occupation (its size is proportional to the number of feature vectors). We use the classical kd-tree data structure [13], whose general construction principle is recalled thereunder:

Every node of the kd-tree represents a subset of the feature vectors, and every node (except the leaves) has two successor nodes representing two sets which form a partition of the set represented by the ancestor node, with respect to an hyperplane which is orthogonal to one coordinate. More specifically, let $\mathcal{F}$ denote the whole set of features. Let n be a node of the tree, $\mathcal{S}(n)$ denote the subset of $\mathcal{F}$ represented by n, $\mathrm{Succ}_L(n)$ (resp. $\mathrm{Succ}_R(n)$) denote the left (resp. right) successor node of n. If U denotes a feature vector, let U_j denote the j-th component of U, let d be the dimension of the feature vectors.

1. There exists one node r such that $\mathcal{S}(r) = \mathcal{F}$. This corresponds to the root of the tree.

2. For every node n, if the cardinality of $\mathcal{S}(n)$ is greater than one, then there exist two nodes p and q such that $\mathrm{Succ}_L(n) = p$ and $\mathrm{Succ}_R(n) = q$, and then: $\mathcal{S}(p) \cup \mathcal{S}(q) = \mathcal{S}(n)$, $\mathcal{S}(p) \cap \mathcal{S}(q) = \emptyset$, and there exists some $z \in \mathbb{R}$ and $j \in \{1, \ldots, d\}$ such that: $\forall U \in \mathcal{S}(p), \forall V \in \mathcal{S}(q)$, $U_j \leq z$ and $V_j > z$. The cutting hyperplane is then orthogonal to the j-th component of the basis, and corresponds to the points whose j-th component has value z (called the cutting value in the j-th dimension).

The actual implementations of kd-tree vary according to different criteria, mainly: (1) the choice of the cutting dimension (simply picking one dimension after the other repeatedly, or cutting first through dimension with maximal variance), and (2) the choice of the cutting value (can be the middle of the range that remains to be cut, which leads to a generalisation of octrees to d-dimension, or can be the median value of the j-th components of the vector in the subset, which allows to get balanced trees). The figure 9 illustrates the construction of a kd-tree in dimension 2, by changing the cutting direction at each level and choosing the median as the cutting value.

The kd-tree is a useful tool for performing nearest neighbours (NN) search in the feature space. It will be extensively used in the following to perform efficiently operations based on visual similarity, that are intrinsically non local in the image space. However, there are many operations where NN search will be needed very intensively (e.g. for every pixel / feature vector). In that case, the computational cost will remain too important for real-time video. Two important optimisations are employed:

– **Approximate Nearest Neighbour** (ANN) search: If $\mathbf{u}$ and $\mathbf{v}$ are two feature vectors, $\mathbf{v}$ is said to be an ε-approximate k-th nearest neighbour of $\mathbf{u}$ if the distance between $\mathbf{u}$ and $\mathbf{v}$ is less than $1 + \varepsilon$ larger than the distance between $\mathbf{u}$ and its actual k-th nearest neighbour. Unlike the exact NN search, the ANN search provides worst-case lower bounds complexity,

and significantly diminishes the average complexity [14]. We have used in our implementations the ANN library developed by Arya and Mount [15].
- **Quantisation of the Feature Space**: Generally the main term in the complexity of the (A)NN search comes from the number of feature vectors ($O(\log(n))$ operations per query in average). So whenever possible we apply vector quantisation in order to limit the size (number of leaves) of the kd-tree.

Depending on the available computation resources, different types of vector quantisation algorithms can be used. For real-time video purposes, the codebook must be constructed dynamically, however it usually does not need to be updated every frame. In the following applications, we use an approximation of the K-means clustering method with linear complexity with respect to the number of pixels used to update the codebook: only one assignment step is performed using a constant radius ε as classification criterion (a new feature vector is assigned to the first word of the codebook which is at distance less than ε if such word exists, a new word is created otherwise), and the centroid of every cluster is computed recursively. Figure 10 illustrates the creation and representation of the codebook.

4.2 Image and object characteristics

The words of the codebook are related to the classical concept of textons, whose statistics provide relevant information on the visual appearance of objects. The first useful visual object descriptor that emerges from this representation is the histogram, or weight vector of the codebook, which is a global descriptor that can be associated to any set of pixels representing an image, a room, or different instances of an object.

Such histogram / weight vector is naturally computed during the quantisation or updating of the codebook, as it takes part of the recursive computation of the centroid of every cluster. At the same time, the set of indexes corresponding to the list of pixels assigned to every word of the codebook is kept in memory. Let $\mathbf{x}$ be a pixel from the image space (typically $\mathbf{x} = (x_1, x_2)$), $\hat{\mathbf{x}}_f$ be the projection of $\mathbf{x}$ in the feature space of f (i.e. the feature vector associated to $\mathbf{x}$, e.g. $\hat{\mathbf{x}}_f = (f_{ij}^{\sigma}(\mathbf{x}))$). Let $\mathcal{F}_f$ be the set of features of image f. We shall denote d^F the distance in the feature space (see Sec. 3), and d^I the distance in the image space. If $\mathbf{u}$ is a feature vector, let $\nu_k^{\mathcal{F}_f}(\mathbf{u})$ be its k-th nearest neighbour in the feature space of f. We denote $\mathcal{F}_f^{-1}(\mathbf{u})$ the set of pixels which are assigned to the codebook $\mathbf{u}$ in the feature space of f. If there is no quantisation, the notation remains valid as $\mathcal{F}_f^{-1}(\mathbf{u}) = \{\mathbf{x}\}$ such that $\hat{\mathbf{x}}_f = \mathbf{u}$.

Figure 11 shows an example of feature quantisation back-projected in the image space. The detail image (right) illustrates the possible utilisation that can be made in terms of higher order statistics (i.e. cooccurrence) of visual words from the codebook.

The nearest neighbour framework also provides an interesting conception of the notion of salient point. Whereas the classical characterisation of interest

points is purely geometrical and relatively independent of the image content, the NN feature based salience is entirely statistical and content-dependent: The salient points correspond to the isolated points in the feature space. Such characterisation is not new and has been done before in the space of patches by Kervrann and Boulanger [16].

More formally, the rarest pixels are defined as:

$$R_1^m = \mathcal{F}_f^{-1}(\arg\max_{\mathbf{u}\in\mathcal{F}_f} \frac{1}{m}\sum_{k=1}^{m} d^F(\mathbf{u}, \nu_k^{\mathcal{F}_f}(\mathbf{u})) \tag{19}$$

The rarest pixels are those assigned to the word with maximal average distance to its m nearest neighbours. Without quantisation, there is only one such pixel. The second rarest pixels R_2^m are defined similarly by excluding the word with maximal distance and so on. The only parameter m merely acts as a filtering value and is of moderate practical importance.

Figure 12 shows three examples of NN based salient points in a single scale local jet feature space. The difference with a purely geometric approach can be seen on images with large regular textures, e.g. the centre image.

Finally we propose another descriptor whose purpose is to provide an alternate and intermediate representation between the fully global codebook histogram and fully local salient point. It is based on the selection and representation of the statistical mode in the feature space.

The selection of the mode in multidimensional data is a difficult problem which has received relatively few attention. Our technique is an adaptation of the methods proposed by Burman and Polonik in [17], and implemented through the framework of geodesic reconstruction in the feature space, which implies the definition of a proper topology.

Suppose that a topology is defined in the feature space, we define a cluster as follows: Let the centre of the main cluster $\kappa_1^{\mathcal{F}_f}$ be defined as the feature vector with minimal average distance to its m NN. The main cluster $K_1^{\mathcal{F}_f}$ is then defined as the connected component of $\mathcal{F}_f$ that contains $\kappa_1^{\mathcal{F}_f}$, or equivalently the geodesic reconstruction of $\kappa_1^{\mathcal{F}_f}$ within $\mathcal{F}_f$. The second main cluster $K_2^{\mathcal{F}_f}$ is defined the same way on $\mathcal{F}_f \setminus K_1^{\mathcal{F}_f}$, and so on.

Now we need a topology, i.e. a criterion to decide whether two vectors of the feature space are connected or not. This is done by distance threshold (2 vectors are connected if their distance is small enough). However such minimal distance has to be adapted to the data. It must be significantly smaller than $\mu_m^{\mathcal{F}_f} = \frac{1}{|\mathcal{F}_f|}\sum_{\mathbf{u}\in\mathcal{F}_f}\frac{1}{m}\sum_{k=1}^{m} d^F(\mathbf{u}, \nu_k^{\mathcal{F}_f}(\mathbf{u}))$, i.e. the average value over $\mathcal{F}_f$ ($|\mathcal{F}_f|$ denotes the cardinality of $\mathcal{F}_f$) of the mean distance to the m nearest neighbours. It must also be larger than $\tau_m^{\mathcal{F}_f} = \min_{\mathbf{u}\in\mathcal{F}_f}\frac{1}{m}\sum_{k=1}^{m} d^F(\mathbf{u}, \nu_k^{\mathcal{F}_f}(\mathbf{u}))$, the minimal average distance to the m nearest neighbours, which identifies the cluster centre $\kappa_1^{\mathcal{F}_f}$. We then dynamically define the topology by using a variable distance threshold defined

as the geometric mean between $\mu_m^{\mathcal{F}_f}$ and $\tau_m^{\mathcal{F}_f}$, i.e. two feature vector $\mathbf{u}$ and $\mathbf{v}$ are connected if and only if: $d^F(\mathbf{u},\mathbf{v}) < \sqrt{\mu_m^{\mathcal{F}_f}\tau_m^{\mathcal{F}_f}}$.

Figure 13 shows the 12 main modes for 3 images. The modes appear as a complementary information of singularities shown on Figure 12. They represent homogeneous zone, simple regular textures, or long straight contours as can be seen in the centre and right images.

5 Non-local means video filtering

The non-local (NL) means filter, originally proposed by Buades *et al* in [18] is a powerful image denoising technique, which can be seen as an extension of the classical convolution scheme, i.e. every pixel value is replaced by a weighted average of the other pixels, but here the weights do not depend on the distance (in the image space) between pixels, but on the local similarity. In our framework, the NL-means is simply expressed by computing the weights according to the feature space.

Let $\mathbf{u}$ and $\mathbf{v}$ be two vectors of the feature space. $\omega(\mathbf{u},\mathbf{v})$ the relative (symmetric) weight of $\mathbf{u}$ with respect to $\mathbf{v}$, is defined as follows:

$$\omega(\mathbf{u},\mathbf{v}) = e^{-\frac{d^F(\mathbf{u},\mathbf{v})}{h^2}} \tag{20}$$

where d^F is the distance in the feature space, and h is a decay parameter, related to the amount of noise to be removed.

Now two variants of the NL means can be considered:

1. Limited range method

$$f_{LR}^{NL}(\mathbf{x}) = \frac{1}{\zeta(\mathbf{x})} \sum_{\mathbf{y}\in\mathcal{N}(\mathbf{x})} f(\mathbf{y})\omega(\hat{\mathbf{x}}_f,\hat{\mathbf{y}}_f) \tag{21}$$

2. Unlimited range method

$$f_{UR}^{NL}(\mathbf{x}) = \frac{1}{\xi(\mathbf{x})} \sum_{\mathbf{u}\in\mathcal{W}(\hat{\mathbf{x}}_f)} \check{f}(\mathbf{u})\omega(\hat{\mathbf{x}}_f,\mathbf{u}) \tag{22}$$

where $\mathcal{N}(\mathbf{x})$ (resp. $\mathcal{W}(\mathbf{v})$) is a neighbourhood of $\mathbf{x}$ (resp. $\mathbf{v}$), usually corresponding to the k nearest neighbours of $\mathbf{x}$ (resp. $\mathbf{v}$) in the image (resp. feature) space. Figure 14 illustrates the difference between the limited and unlimited range methods.

$\zeta(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{N}(\mathbf{x})} \omega(\hat{\mathbf{x}}_f,\hat{\mathbf{y}}_f)$ and $\xi(\mathbf{x}) = \sum_{\mathbf{u}\in\mathcal{W}(\hat{\mathbf{x}}_f)} \omega(\hat{\mathbf{x}}_f,\mathbf{u})$ are the respective normalisation constants. $\check{f}(\mathbf{u})$ is the value of f corresponding to feature $\mathbf{u}$. It is defined as:

$$\check{f}(\mathbf{u}) = \frac{1}{|\mathcal{F}_f^{-1}(\mathbf{u})|} \sum_{\mathbf{x}\in\mathcal{F}_f^{-1}(\mathbf{u})} f(\mathbf{x}) \tag{23}$$

i.e. the average value of f on the pixels corresponding to feature $\mathbf{u}$, this quantity is recursively calculated - or estimated - during the quantisation.

Figure 15 shows some results on the same noisy image (only the bottom half diagonal is processed). In all images, the decay parameter was set automatically by using a fast estimation of the noise variance (see [12] for more details). It is somewhat surprising that the denoising quality looks better for the limited range (LR, Fig. 15(1)) than for the unlimited range (UR, Fig. 15(2)). However this can be explained by the fact that on the one hand, the edge and corner pixels will be more affected by the UR methods, since the relative weights of their neighbours will be much higher in the feature space than in the image space. One the other hand, for large noisy homogeneous regions, the local jet UR method will be able to find patterns that will tend to exaggerate the texturation of these regions. Anyway, using kd-trees, the unlimited range method is much faster than the limited range one when the cardinalities of $\mathcal{N}(\mathbf{x})$ and $\mathcal{W}(\hat{\mathbf{x}}_f)$ are of the same order. Furthermore, using approximate search significantly lowers the computation time without affecting the results too much (Fig. 15(3)). Finally, the most important acceleration is obtained by reducing the size of the search structure thanks to quantisation. We also expect that quantising the feature space should compensate the drawbacks of the UR method evoked above, and then also improve the quality of denoising (Fig. 15(4)), but more quantitative evaluation is needed.

6 Optical flow estimation

The optical flow, or apparent motion estimation, is one of the most classical inverse problem in video computing. It consists in estimating, for every pixel $\mathbf{x}$ at frame t the apparent velocity, corresponding to the projection in the focal plane of the relative velocity of a visible physical point of the scene with respect to the camera. It is actually estimated using the correspondence of pixels from one frame to the other.

The optical flow estimation turns out to be - from a conceptual point of view at least - one of the most straightforward applications of the feature space based similarity. At frame t, for image f_t, and for every pixel $\mathbf{x}$, we compute $\mathbf{u}(f_{t-1}, f_t, \mathbf{x})$, the nearest neighbour of the feature vector associated to $\mathbf{x}$, in the feature space of f_{t-1}:

$$\mathbf{u}(f_{t-1}, f_t, \mathbf{x}) = \arg \min_{\mathbf{v} \in \mathcal{F}_{f_{t-1}}} d^F(\hat{\mathbf{x}}_{f_t}, \mathbf{v}) = \nu_1^{\mathcal{F}_{f_{t-1}}}(\hat{\mathbf{x}}_{f_t}) \tag{24}$$

Then we can compute $\mathbf{y}(f_{t-1}, f_t, \mathbf{x})$, the pixel from f_{t-1} which is the most similar to $\mathbf{x}$ from f_t:

$$\mathbf{y}(f_{t-1}, f_t, \mathbf{x}) = \arg \min_{\mathbf{z} \in \mathcal{F}^{-1}_{f_{t-1}}(\mathbf{u}(f_{t-1}, f_t, \mathbf{x}))} d^I(\mathbf{x}, \mathbf{z}) \tag{25}$$

If no quantisation is performed, this is simply the pixel corresponding to feature $\mathbf{u}$ in f_{t-1}, otherwise it is the pixel from $\mathcal{F}^{-1}_{f_{t-1}}(\mathbf{u}(f_{t-1}, f_t, \mathbf{x}))$ (the set of

pixels associated to feature vector $\mathbf{u}$) which is the closest from $\mathbf{x}$ in the image space. Figure 16 illustrates the principle of the optical flow estimation by nearest neighbour search in the local jet feature space.

An alternate formulation can also be proposed by following the NL-mean framework:

$$\mathbf{y}(f_{t-1}, f_t, \mathbf{x}) = \frac{1}{\xi^F(\mathbf{x})} \sum_{k=1}^{m} \left(\frac{\omega^F(\hat{\mathbf{x}}_{f_t}, \nu_k^{\mathcal{F}_{f_{t-1}}}(\hat{\mathbf{x}}_{f_t}))}{\xi^I(\mathbf{x})} \sum_{\mathbf{z} \in \mathcal{F}_{f_{t-1}}^{-1}(\nu_k^{\mathcal{F}_{f_{t-1}}}(\hat{\mathbf{x}}_{f_t}))} \omega^I(\mathbf{x}, \mathbf{z}) \mathbf{z} \right) \tag{26}$$

with $\omega^I(\mathbf{x}, \mathbf{y}) = e^{-\frac{d^I(\mathbf{x},\mathbf{y})}{h^2}}$ and $\omega^F(\mathbf{u}, \mathbf{v}) = e^{-\frac{d^F(\mathbf{u},\mathbf{v})}{h^2}}$ the weight functions in the image and feature space respectively. $\xi^I(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{F}_{f_{t-1}}^{-1}(\nu_k^{\mathcal{F}_{f_{t-1}}}(\hat{\mathbf{x}}_{f_t}))} \omega^I(\mathbf{x}, \mathbf{y})$ and $\xi^F(\mathbf{x}) = \sum_{k=1}^{m} \omega^F(\hat{\mathbf{x}}_{f_t}, \nu_k^{\mathcal{F}_{f_{t-1}}}(\hat{\mathbf{x}}_{f_t}))$ are the two corresponding normalising functions. This second formulation, more general, allows to perform regularisation, in the feature space, in the image space, or in both. Obviously, when the number of NN m is one, and without quantisation, the two formulations are equivalent.

Finally, the velocity vector is computed as the difference:

$$\mathbf{c}(f_{t-1}, f_t, \mathbf{x}) = \mathbf{x} - \mathbf{y}(f_{t-1}, f_t, \mathbf{x}) \tag{27}$$

Figure 17 shows examples of optical flow estimation on different images taken from classical test sequences. In these examples, the number of NN is 1, there is no quantisation and the local jet is calculated at order 2, for 5 different scales. It is worth mentioning that no post-processing regularisation is performed; the regularity of the result depends on the number of scales of the local jet.

7 Background subtraction

Motion detection refers to the binary labelling of every pixel of a video sequence: 0 if it belongs to the static scene (background), 1 if it belongs to a moving object (foreground). When the camera is stationary, the most popular methods are based on background modelling and subtraction, which correspond to calculating locally (say for every pixel or block), a set of parameters representing temporal statistics of the background, in order to compare every new value with those parameters, to decide whether this value is typical of the background or not.

This problem can be relatively challenging in many cases, because usually the background is not completely static, but is evolving because of illumination changes, and also because parts of the scene are animated by irrelevant motion (e.g. waving trees, snowfall, air fan, etc), those changes must then be taken into account in the background model. The precision of modelling of the background

statistics has strong influence on the computational cost, both in terms of memory and time. One good trade-off in terms of representation is obtained by the sample and consensus methods [19, 20], which consist in keeping in memory a limited set of sampled values, and then comparing the current value to those samples to decide whether the pixel is foreground or not. Vector quantisation has also been used for background modelling [21] in colour/brightness space. The algorithm we propose here is a combination of sample/consensus and vector quantisation in the local jet feature space.

In this application, we use a single codebook $\mathcal{F}_f$ of quantised features for the whole sequence (and not one codebook by frame $\mathcal{F}_{f_t}$). However, $\mathcal{F}_f$ may evolve over time, as we shall see later. The basic principle is the following: In every pixel, the temporal activity is modelled by keeping in memory M prototypes $\{\mathbf{m}_j(f,\mathbf{x})\}_{j\in\{1,M\}}$, such that $\mathbf{m}_j(f,\mathbf{x}) \in \mathcal{F}_f$. The set of pixel prototypes represent a sample of its past values in the feature space, and M is a temporal depth parameter.

Let ρ be a positive number representing the distance threshold in the feature space; τ an integer such that $1 < \tau < N$. The foreground label $e(f,t,\mathbf{x})$, indicating whether $\mathbf{x}$ in f_t belongs to a moving object or not is then calculated as follows:

$$e(f,t,\mathbf{x}) = 1 \text{ if } |\{j \in \{1,M\}; d^F(\hat{\mathbf{x}}_{f_t}, \mathbf{m}_j(f,\mathbf{x})) > \rho\}| > \tau \tag{28}$$

$$= 0 \text{ otherwise} \tag{29}$$

Then a pixel whose value in the feature space is at a distance greater than ρ for more than τ of its M prototypes is considered foreground, elsewhere it is classified as background. The advantage of using a complex feature space instead of the mere colour is that we are able to capture more sophisticated image structure and then make the background modelling more robust. On the other hand, the vector quantisation dramatically reduces the memory cost, because only the index of the word from the codebook is used instead of a high dimensional vector. It is typically observed that a large majority of pixels only have one or two different indexes within their M background prototypes.

Our practical implementation for coding and updating the prototypes is simply an adaptation of the state-of-the-art *ViBe* algorithm [20]:

- **Coding:** For every pixel $\mathbf{x}$, we code the prototypes P using a matrix with two rows and a variable number of columns (maximum M). P(i,1) represents the index of the i-th prototype, and P(i,2) its frequency, such that the sum of all the P(i,2) is always equal to M.
- **Initialisation:** At the first frame ($t = 0$) P(1,1) equals the index of $\hat{\mathbf{x}}_{f_0}$ and P(1,2) = M.
- **Updating:** At time t, the index of $\hat{\mathbf{x}}_{f_t}$ replaces one of the prototypes randomly selected: choosing a random value k between 1 and M, we calculate the smallest index h such that $\sum_{i=1}^{h} \mathrm{P(i,2)} \geq k$, and decrement P(h,2). Then $\hat{\mathbf{x}}_{f_t}$ is compared to the existing prototypes represented by P(i,1), until it

matches one of them, say number h', and in that case the matched index frequency P(h',2) is incremented. If no prototype is matched, a new one is created by searching the first index h" such that P(h",2) = 0, which by construction exists. Then P(h",2) is set to 1, and P(h",1) to the index of $\hat{\mathbf{x}}_{f_t}$.

Figure 18 illustrates the principles of the ViBe method applied in the local jet feature dictionary.

For the creation of the codebook, we use, as in the NL-mean case a very basic incremental version of the K-means algorithm for real-time video purposes. It is worth mentioning that the codebook does not need to be updated for every frame, nor everywhere, for example, it can be updated every 5 frames for the foreground pixels, and every 100 frames for the whole image.

Figure 19 shows an example of foreground labelling results for three frames from an outdoor colour sequence. We use a 2 order, 1 scale, and 3 colour local jet feature space (i.e. 18D vector features), with a codebook limited to 3,000 words, the temporal depth is $M = 20$, the distance threshold in the feature space $\rho = 30.0$, and the consensus threshold is $\tau = M/2$. Note that, unlike [20], no spatial diffusion is performed, and the update is not strictly conservative, i.e. the update is made every 4 frames for background pixels, and every 16 frames for foreground pixels.

8 Conclusion

We have proposed in this report a unified framework based on the representation of the visual data in the local jet feature space. This framework is devoted to a universal use, i.e. from the lowest to the highest tasks of artificial vision. We have shown the relevance of the approach for several low level vision tasks. This representation also naturally provides image reduction and description tools that can be used at a higher processing level. We think particularly to object modelling and recognition, which is part of our ongoing work.

Some of the proposed algorithms, for example local jet based NL-means and background subtraction based on sample and consensus in the local jet space are particularly efficient and can be easily adapted to real-time. However, the computational cost remains an issue for different implementations: the optical flow by nearest neighbour search in the local jet space and the computation of the mode of the local jet distribution are two important examples. We are then investigating new ways to compute the nearest neighbours in the feature space using parallel implementations.

References

1. Peyré, G.: Manifold models for signals and images. Computer Vision and Image Understanding **113**(2) (2009) 249–260

2. Freeman, W., Adelson, E.: The design and use of steerable filters. IEEE Trans. on Pattern Analysis and Machine Intelligence **13**(9) (1991) 891–906
3. Rubner, Y., Tomasi, C.: Texture-based image retrieval without segmentation. In: IEEE International Conference on Computer Vision, Kerkyra, Greece (1999) 1018–1024
4. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: Workshop on Statistical Learning in Computer Vision (ECCV'04). (2004) 1–22
5. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. International Journal of Computer Vision **60**(1) (2004) 63–86
6. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2) (2004) 91–110
7. Koenderink, J., Van Doorn, A.: Representation of local geometry in the visual system. Biological Cybernetics **55** (1987) 367–375
8. Schmid, C., Mohr, R.: Local grayvalue invariants for image retrieval. IEEE Trans. on Pattern Analysis and Machine Intelligence **19**(5) (1997) 530–534
9. Crosier, M., Griffin, L.: Using basic image features for texture classification. International Journal of Computer Vision **88**(3) (2010) 447–460
10. Lindeberg, T.: Feature detection with automatic scale selection. International Journal of Computer Vision **30**(2) (1998) 77–116
11. Orchard, J., Ebrahimi, M., Wong, A.: Efficient non-local means denoising using the SVD. In: Proc. IEEE ICIP. (2008) 1732–1735
12. Manzanera, A.: Local jet based similarity for NL-means filtering. In: International Conference on Pattern Recognition, Istambul, Turkey (2010) 2668–2671
13. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9) (1975) 509–517
14. Arya, S., Mount, D., Netanyahu, N., Silverman, R., Wu, A.: An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. Journal of the ACM **45**(6) (1998) 891–923
15. Mount, D., Arya, S.: ANN: A library for approximate nearest neighbor searching. In: CGC Workshop on Computational Geometry. (1997) `http://www.cs.umd.edu/~mount/ANN/`.
16. Kervrann, C., Boulanger, J.: Local adaptivity to variable smoothness for exemplar-based image denoising and representation. International Journal of Computer Vision **79**(1) (August 2008) 45–69
17. Burman, P., Polonik, W.: Multivariate mode hunting: Data analytic tools with measures of significance. J. Multivar. Anal. **100**(6) (2009) 1198–1218
18. Buades, A., Coll, B., Morel, J.: A non-local algorithm for image denoising. In: Proc. IEEE CVPR. Volume 2. (2005) 60–65
19. Wang, H., Suter, D.: A consensus-based method for tracking: Modelling background scenario and foreground appearance. Pattern Recognition **40**(3) (2007) 1091–1105
20. Barnich, O., Van Droogenbroeck, M.: ViBe: a powerful random technique to estimate the background in video sequences. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2009), IEEE (April 2009) 945–948
21. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: International Conference on Image Processing (ICIP'04). Volume 5., IEEE (2004) 3061–3064

Fig. 8. Similarity maps for 3 different pixels: (1), (2) and (3), and 6 different distances: Single scale: (a) canonical components (CC), (b) rotation invariant components (RIC), Pan-scalic: (c) CC, (d) RIC, and Trans-scalic: (e) CC, (f) RIC. (Painting by Lowell Herrero)

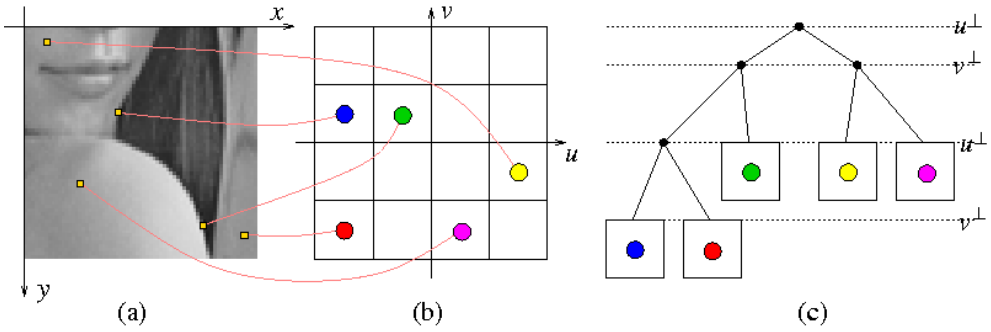

Fig. 9. Pixels from the image space (a) are projected into the feature space (b) and collected into a kd-tree structure (c).

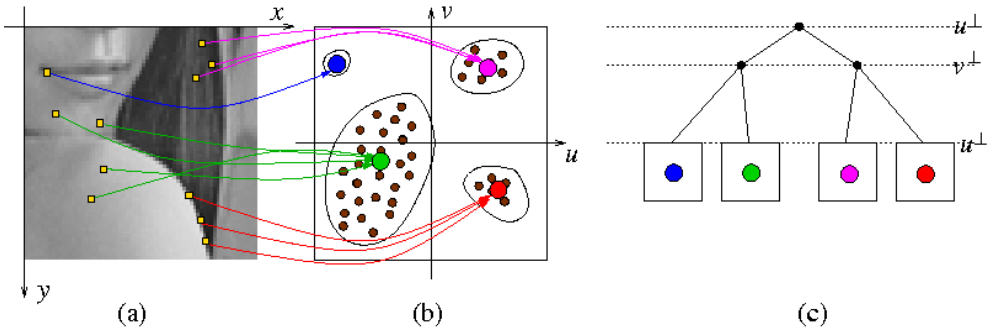

Fig. 10. Quantisation of the feature space: many pixels from the image space (a) are associated to the same word of the codebook (quantised feature space) (b) and only the words of the codebook are coded in the kd-tree (c).

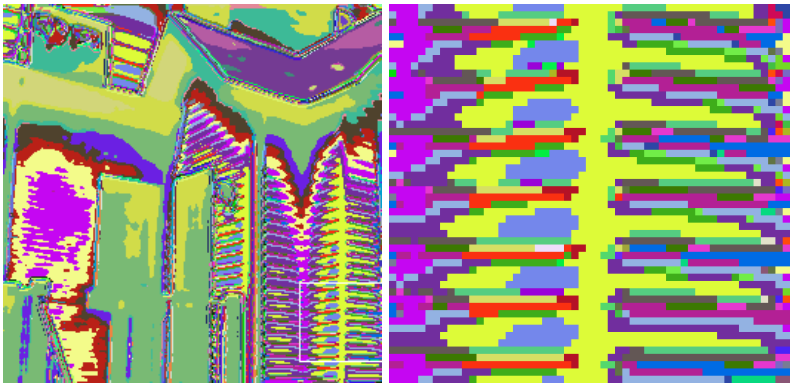

Fig. 11. Quantisation in the local jet space. The feature space is reduced to 506 vectors. The right image is a detail corresponding to the white rectangle in the left image.

Fig. 12. Salient points (isolated points in the feature space back-projected in the image space): The figure displays the pixel corresponding to the 100 feature vectors whose average distance to their 10 NN is the largest. The feature space is: Local jet of order 2, one single scale $\sigma = 1.5$, no quantisation. Here a minimal exclusion distance of 5 in the image space is used to avoid clustering of the salient pixels.

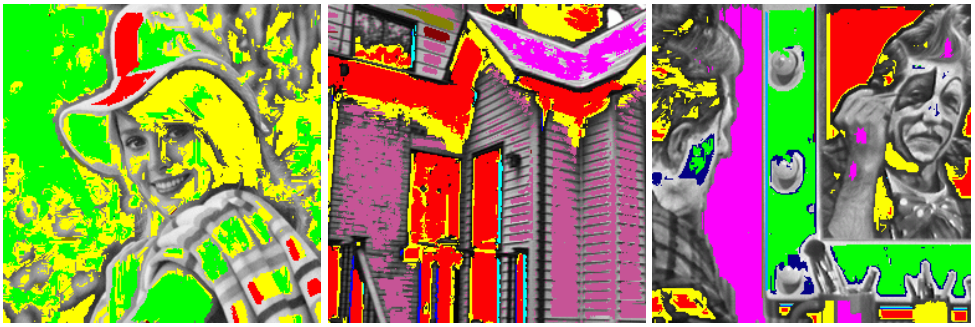

Fig. 13. First modes of the representation (clusters of the feature space back-projected in the image space): The figure displays the pixel corresponding to the 12 main clusters of the feature space (Local Jet of order 2, with 2 scales $\{\sigma_1 = 1.0, \sigma_2 = 2.0\}$, no quantisation). A cluster is the connected component containing the cluster centre, defined as the point of the feature space with smallest average distance to its 20 NN.

(a) (b)

Fig. 14. Limited (a) *vs* Unlimited (b) range approaches in the computation of the NL-means. $\mathcal{N}(\mathbf{x})$ is the set of the nearest neighbours of $\mathbf{x}$ in the image space. $\check{\mathcal{W}}(\hat{\mathbf{x}})$ is the back-projection in the image space of the nearest neighbours of $\hat{\mathbf{x}}$ in the feature space.

Fig. 15. NL-means filtering in the local jet feature space. (1) Limited range ($\mathcal{N}(\mathbf{x})$ is a 17×17 square neighbourhood of $\mathbf{x}$ in the image domain). (2) Unlimited range, exact search ($\mathcal{W}(\hat{\mathbf{x}}_f)$ corresponds to the 30 NN in the local jet domain), (3) Unlimited range, approximate search ($\varepsilon = 10.0$), (4) Unlimited range, exact search, with quantised feature space (1938 words in the dictionary).

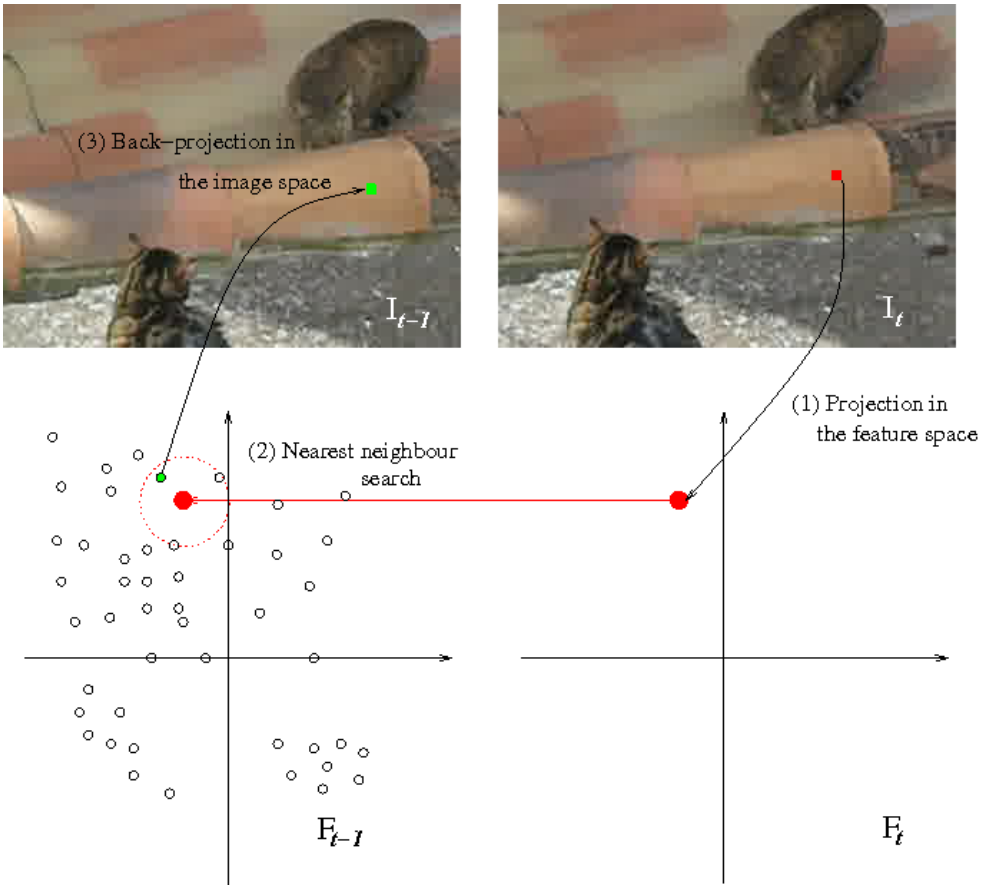

Fig. 16. Optical Flow estimation by Nearest Neighbour Search in the Local Jet Feature Space.

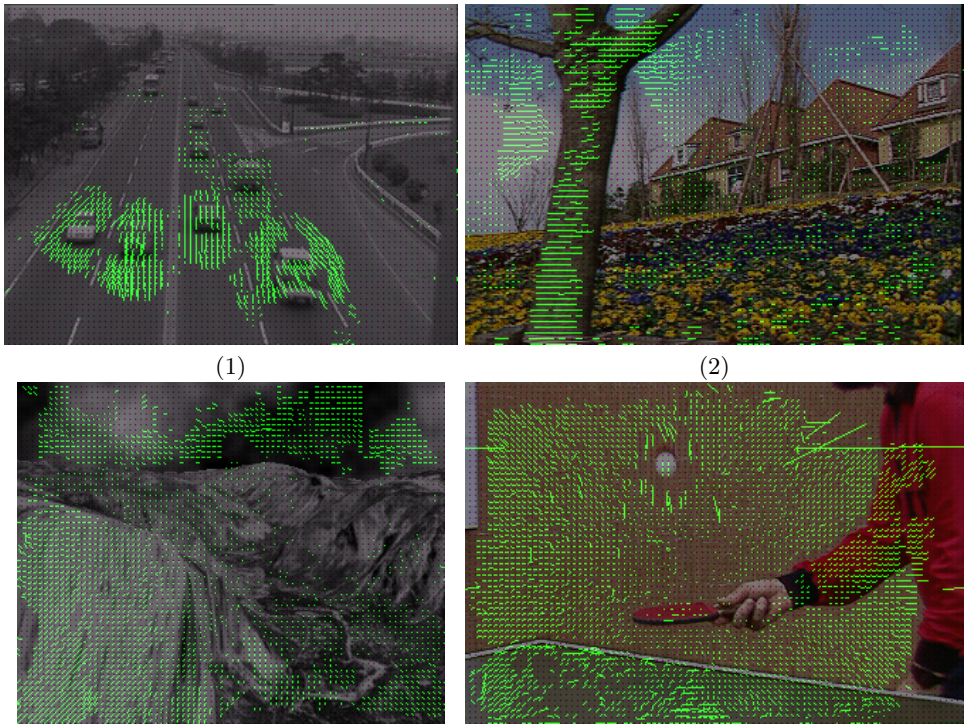

Fig. 17. Optical flow fields estimated by nearest neighbour search in the feature space. (1) Stationary camera, (2) Horizontal travelling, (3) Forward zooming, (4) Backward zooming and moving objects.

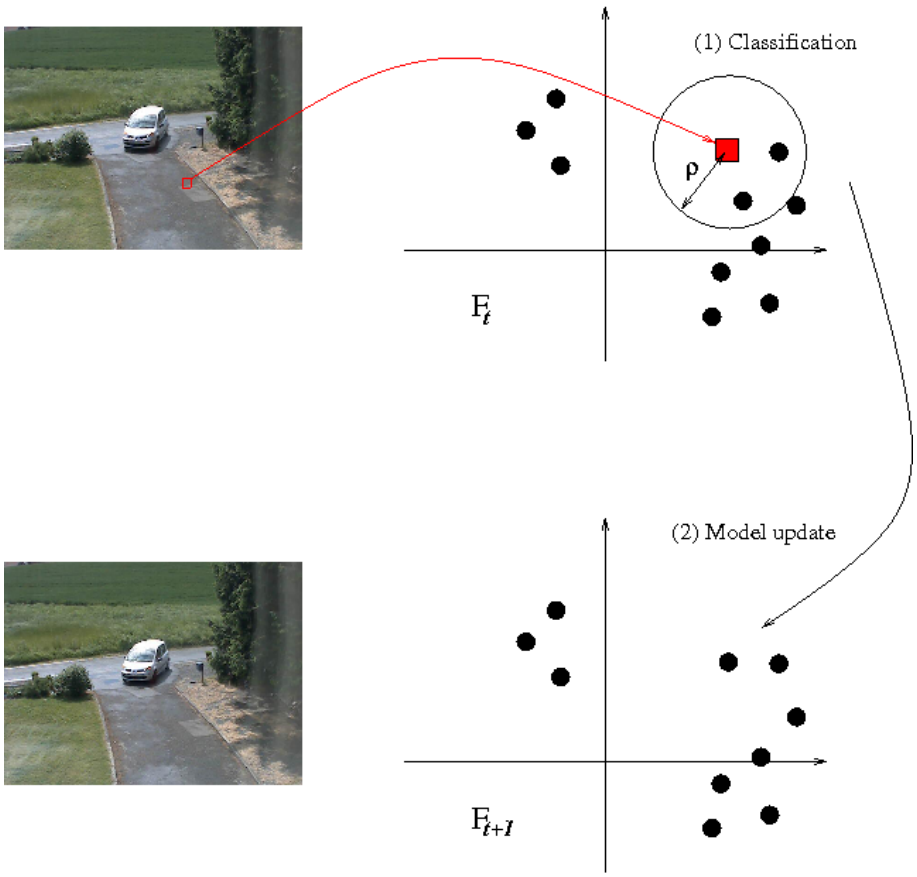

Fig. 18. Adaptation of the ViBe algorithm to the local jet feature dictionary. Top, classification step: The pixel $\mathbf{x}$ is classified foreground if the number of prototypes at distance less than ρ from $\hat{\mathbf{x}}_{f_t}$ is inferior to a certain threshold. Bottom, update step: $\hat{\mathbf{x}}_{f_t}$ replaces one of the prototypes, randomly selected.

Fig. 19. Background subtraction based on sample and consensus using a codebook of colour local jet features.