Pattern Recognition 46 (2013) 3519–3532

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Regularized vector field learning with sparse approximation for mismatch removal

Jiayi Ma [a], Ji Zhao [a], Jinwen Tian [a], Xiang Bai [b,d,*], Zhuowen Tu [c,d]

[a] Institute for Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology, Wuhan 430074, China
[b] Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[c] Lab of Neuro Imaging, University of California, Los Angeles, CA 90095, USA
[d] Microsoft Research Asia, Beijing 100080, China

ARTICLE INFO

Article history:
Received 16 May 2012
Received in revised form
28 February 2013
Accepted 21 May 2013
Available online 6 June 2013

Keywords:
Vector field learning
Sparse approximation
Regularization
Reproducing kernel Hilbert space
Outlier
Mismatch removal

ABSTRACT

In vector field learning, regularized kernel methods such as regularized least-squares require the number of basis functions to be equivalent to the training sample size, N. The learning process thus has $O(N^3)$ and $O(N^2)$ in the time and space complexity, respectively. This poses significant burden on the vector learning problem for large datasets. In this paper, we propose a sparse approximation to a robust vector field learning method, sparse vector field consensus (*SparseVFC*), and derive a statistical learning bound on the speed of the convergence. We apply SparseVFC to the mismatch removal problem. The quantitative results on benchmark datasets demonstrate the significant speed advantage of SparseVFC over the original VFC algorithm (two orders of magnitude faster) without much performance degradation; we also demonstrate the large improvement by SparseVFC over traditional methods like RANSAC. Moreover, the proposed method is general and it can be applied to other applications in vector field learning.

1. Introduction

For a given set of data $S = \{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}\}_{n=1}^N$, one can learn a function $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$ so that it approximates a mapping to output $\mathbf{y}_n$ for an input $\mathbf{x}_n$. In this process one can fit a function to the given training data with a smoothness constraint, which typically achieves some form of capacity control of the function space. The learning process aims to balance the data fitting and model complexity, and thus, produces a robust algorithm that generalizes well to the unseen data [1]. This can one way be formulated into an optimization problem with a certain choice of regularization [2,3], which typically operates in the Reproducing Kernel Hilbert Space (RKHS) [4] (associated with a particular kernel). Two well-known methods along the line of learning with regularization are regularized least-squares (RLS) and support vector machines (SVM) [2]. These regularized kernel methods have drawn much attention due to their computational simplicity and strong generalization power in traditional machine learning problems such as regression and classification.

Regularized kernel methods over RKHS often lead to solving a convex optimization problem. For example, RLS directly solves a linear system. However, the computational complexity associated with applying kernel matrices is relatively high. Given a set of N training samples, the kernel matrix is of size $N \times N$. This suggests a space complexity $O(N^2)$ and a time complexity at least $O(N^2)$; in fact most regularized kernel methods have core operations such as matrix inversion in RLS which is of $O(N^3)$. It is therefore computationally demanding if N is large. This situation is more troublesome in vector-valued cases where we focus on the vector-valued regularized least-squares (vector-valued RLS). There are several different names for the RLS model [5] such as regularization networks (RN) [3], kernel ridge regression (KRR) [6], least squares support vector machines (LS-SVM) [7], etc. We use a widely adopted one in the literature, i.e. RLS in this paper.

A vector field is a map that assigns each position $\mathbf{x} \in \mathbb{R}^P$ with a vector $\mathbf{y} \in \mathbb{R}^D$, defined by a vector-valued function. Examples of vector field range from the velocity fields of fluid particles to the optical flow fields associated with visual motion. The problem of vector field learning is tied with functional estimation and supervised learning. We may learn a vector field by using regularized kernel methods and, in particular, vector-valued RLS [8,9]. According to the representer theorem in RKHS, the optimal solution is a linear combination of a number (the size of training data points N) of basis functions [10]. The corresponding kernel matrix is of size $DN \times DN$. Compared to the scalar case, the

* Corresponding author at: Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, China. Tel.: +86 13297073017.
E-mail addresses: jyma2010@gmail.com (J. Ma), zhaoji84@gmail.com (J. Zhao), jwtian@mail.hust.edu.cn (J. Tian), xbai@hust.edu.cn, xiang.bai@gmail.com (X. Bai), ztu@loni.ucla.edu (Z. Tu).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.05.017

vectored values pose more challenges on large datasets. One possible solution is to learn sparse basis functions; i.e. we could pick a subset of size M basis functions, making the kernel matrix of size $DM \times DM$. Thus, the computational complexity could be greatly reduced when $M \ll N$. Moreover, the solution based on sparse bases also enables faster prediction. Therefore, when we learn a vector field via vector-valued regularized kernel methods, using a sparse representation may be of particular advantage.

Motivated by the recent sparse representation literature [11], here we investigate a suboptimal solution to the vector-valued RLS problem. We derive an upper bound for the approximation accuracy based on the representer theorem. The upper bound has a rate of convergence in $O(\sqrt{1/M})$, indicating a fast convergence rate. We develop a new algorithm, SparseVFC, by using the sparse approximation in a robust vector field learning problem, vector field consensus (VFC) [9]. As in [9], the SparseVFC algorithm is applied to the mismatch removal problem. The experimental results on various 2D and 3D datasets show the clear advantages of SparseVFC over the competing methods in efficiency and robustness.

1.1. Related work

Sparse representations [11] have recently gained a considerable amount of interest for learning the kernel machines [12–15]. To scale up kernel methods to deal with large data, a wealthy body of efficient sparse approximations have been suggested in the scalar case, including the Nyström approximation [16], sparse greedy approximations [17], reduced support vector machines [18], and incomplete Cholesky decomposition [19]. Common to all these approximation schemes is that only a subset of the latent variables are treated exactly, with the remaining variables being approximated. These methods differ in the mechanism for choosing the subset, and in the matrix form used to represent the hypothesis space. In this paper, we generalize this idea to the vector-valued setting.

Another approach to achieve the sparse representation is to add regularization term penalizing the ℓ_1 norm of the expansion coefficients [20,21]. However, this does not alleviate the problem, since it has to compute (and invert) the kernel matrix of size $N \times N$, and then it is not suitable for large datasets. To overcome this problem, Kim et al. [22] described a specialized interior-point method for solving large scale ℓ_1-regularized that uses the preconditioned conjugate gradients algorithm to compute the search direction.

The large scale kernel learning problem can also be addressed by iterative optimization such as conjugate gradient [23] and more efficient decomposition-based methods [24,25]. Specially, for the RLS model with the ℓ_2 loss, fast optimization algorithms such as accelerated gradient descent in conjunction with stochastic methods [26–28] can perform very fast as well. However, a main drawback of these methods is that they have to store a kernel matrix of size $N \times N$ which is not realistic for large datasets.

The main contributions of our work include (i) we present a sparse approximation algorithm for vector-valued RLS which has linear time and space complexity w.r.t. the training data si ze;
(ii) we derive an upper bound for the approximation accuracy of the proposed sparse approximation algorithm in terms of regularized risk functional; (iii) based on the sparse approximation and our the vector field learning method VFC, we give a new algorithm SparseVFC which significantly speeds up the original VFC algorithm without scarifies in accuracy; (iv) we apply SparseVFC to mismatch removal which is a fundamental problem in computer vision and the results demonstrate its superiority over the original VFC algorithm and many other state-of-the-art methods.

The rest of the paper is organized as follows. In Section 2, we briefly review the vector-valued Tikhonov regularization and lay out a sparse approximation algorithm for solving vector-valued RLS. In Section 3, we apply the sparse approximation to robust vector field learning and mismatch removal problem. The datasets and evaluation criteria used in the experiments are presented in Section 4. In Section 5, we evaluate the performance of our algorithm on both synthetic data and real-world images. Finally, we conclude in Section 6.

2. Sparse approximation algorithm

We start by defining the vector-valued RKHS and recalling the vector-valued Tikhonov regularization, and then lay out a sparse approximation algorithm for solving the vector-valued RLS problem. At last, we derive a statistical learning bound for the sparse approximation.

2.1. Vector-valued reproducing kernel Hilbert spaces

We are interested in a class of vector-valued kernel methods, where the hypotheses space is chosen to be a reproducing kernel Hilbert space (RKHS). This motivates reviewing the basic theory of vector-valued RKHS. The development of the theory in the vector case is essentially the same as in the scalar case. We refer to [10,29] for further details and references.

Let $\mathcal{Y}$ be a real Hilbert space with inner product (norm) $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$, ($\| \cdot \|_{\mathcal{Y}}$), for example, $\mathcal{Y} \subseteq \mathbb{R}^D$, $\mathcal{X}$ a set, for example, $\mathcal{X} \subseteq \mathbb{R}^P$, and $\mathcal{H}$ a Hilbert space with inner product (norm) $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, ($\| \cdot \|_{\mathcal{H}}$). A norm can be defined via an inner product, for example, $\|\mathbf{f}\|_{\mathcal{H}} = \langle \mathbf{f}, \mathbf{f} \rangle_{\mathcal{H}}^{1/2}$. Next, we recall the definition of RKHS as well as some properties of it which we will use in this paper.

Definition 1. A Hilbert space $\mathcal{H}$ is an RKHS if the evaluation maps $ev_{\mathbf{x}} : \mathcal{H} \to \mathcal{Y}$ are bounded, i.e. if $\forall \mathbf{x} \in \mathcal{X}$ there exists a positive constant $C_{\mathbf{x}}$ such that

$$\|ev_{\mathbf{x}}(\mathbf{f})\|_{\mathcal{Y}} = \|\mathbf{f}(\mathbf{x})\|_{\mathcal{Y}} \le C_{\mathbf{x}} \|\mathbf{f}\|_{\mathcal{H}}, \quad \forall \mathbf{f} \in \mathcal{H}. \tag{1}$$

A reproducing kernel $\mathbf{\Gamma} : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{Y})$ is then defined as

$$\mathbf{\Gamma}(\mathbf{x}, \mathbf{x}') := ev_{\mathbf{x}} ev_{\mathbf{x}'}^{*}, \tag{2}$$

where $\mathcal{B}(\mathcal{Y})$ is the space of bounded operators on $\mathcal{Y}$, for example, $\mathcal{B}(\mathcal{Y}) \subseteq \mathbb{R}^{D \times D}$, and $ev_{\mathbf{x}}^{*}$ is the adjoint of $ev_{\mathbf{x}}$.

From the definition of the RKHS, we can derive the following properties:

(i) For each $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$, the kernel $\mathbf{\Gamma}$ has the following reproducing property

$$\langle \mathbf{f}(\mathbf{x}), \mathbf{y} \rangle_{\mathcal{Y}} = \langle \mathbf{f}, \mathbf{\Gamma}(\cdot, \mathbf{x}) \mathbf{y} \rangle_{\mathcal{H}}, \quad \forall \mathbf{f} \in \mathcal{H}. \tag{3}$$

(ii) For every $\mathbf{x} \in \mathcal{X}$ and $\mathbf{f} \in \mathcal{H}$, we have that

$$\|\mathbf{f}(\mathbf{x})\|_{\mathcal{Y}} \le \|\mathbf{\Gamma}(\mathbf{x}, \mathbf{x})\|_{\mathcal{Y},\mathcal{Y}}^{1/2} \|\mathbf{f}\|_{\mathcal{H}}, \tag{4}$$

where $\| \cdot \|_{\mathcal{Y},\mathcal{Y}}$ is the operator norm.

The property (i) can be easily derived from the equation $ev_{\mathbf{x}}^{*}\mathbf{y} = \mathbf{\Gamma}(\cdot, \mathbf{x})\mathbf{y}$. In the property (ii), we assume that $\sup_{\mathbf{x} \in \mathcal{X}} \| \mathbf{\Gamma}(\mathbf{x}, \mathbf{x})\|_{\mathcal{Y},\mathcal{Y}}^{1/2} = s_{\mathbf{\Gamma}} < \infty$.

Similar to the scalar case, for any $N \in \mathbb{N}$, $\{\mathbf{x}_n : n \in \mathbb{N}_N\} \subseteq \mathcal{X}$, and a reproducing kernel $\mathbf{\Gamma}$, a unique RKHS can be defined by considering the completion of the space

$$\mathcal{H}_N = \left\{ \sum_{n=1}^{N} \mathbf{\Gamma}(\cdot, \mathbf{x}_n) \mathbf{c}_n : \mathbf{c}_n \in \mathcal{Y} \right\}, \tag{5}$$

with respect to the norm induced by the inner product

$$\langle \mathbf{f}, \mathbf{g} \rangle_{\mathcal{H}} = \sum_{i,j=1}^{N} \langle \boldsymbol{\Gamma}(\mathbf{x}_j, \mathbf{x}_i)\mathbf{c}_i, \mathbf{d}_j \rangle_{\mathcal{Y}}, \tag{6}$$

for any $\mathbf{f}, \mathbf{g} \in \mathcal{H}_N$ with $\mathbf{f} = \sum_{i=1}^{N} \boldsymbol{\Gamma}(\cdot, \mathbf{x}_i)\mathbf{c}_i$ and $\mathbf{g} = \sum_{j=1}^{N} \boldsymbol{\Gamma}(\cdot, \mathbf{x}_j)\mathbf{d}_j$.

2.2. Vector-valued Tikhonov regularization

Regularization aims to stabilize the solution of an ill-conditioned problem. Given an N-sample $S = \{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y} : n \in \mathbb{N}_N\}$ of patterns, where $\mathcal{X} \subseteq \mathbb{R}^P$ and $\mathcal{Y} \subseteq \mathbb{R}^D$ are input space and output space, respectively, in order to learn a mapping $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$, the vector-valued Tikhonov regularization in an RKHS $\mathcal{H}$ with kernel $\boldsymbol{\Gamma}$ minimizes a regularized risk functional [10]

$$\Phi(\mathbf{f}) = \sum_{n=1}^{N} V(\mathbf{f}(\mathbf{x}_n), \mathbf{y}_n) + \lambda \|\mathbf{f}\|_{\mathcal{H}}^2, \tag{7}$$

where the first term is called the empirical error with the loss function $V : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ satisfying $V(\mathbf{y}, \mathbf{y}) = 0$, the second term is a stabilizer with a regularization parameter λ controlling the trade-off between these two terms.

Theorem 1 (*Vector-valued Representer Theorem Micchelli and Pontil [10]*). *The optimal solution of the regularized risk functional (7) has the form*:

$$\mathbf{f}^o(\mathbf{x}) = \sum_{n=1}^{N} \boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x}_n)\mathbf{c}_n, \quad \mathbf{c}_n \in \mathcal{Y}. \tag{8}$$

Hence, minimizing over the (possibly) infinite dimensional Hilbert space, boils down to find a finite set of coefficients $\{\mathbf{c}_n : n \in \mathbb{N}_N\}$.

A number of loss functions have been discussed in the literature [1]. In this paper, we focus on vector-valued RLS which is a vector-valued Tikhonov regularization with an ℓ_2 loss function, i.e.,

$$V(\mathbf{f}(\mathbf{x}_n), \mathbf{y}_n) = p_n \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2, \tag{9}$$

where $p_n \geq 0$ is the weight, and $\|\cdot\|$ denotes the ℓ_2 norm. Other common loss functions are the absolute value loss $V(\mathbf{f}(\mathbf{x}), \mathbf{y}) = |\mathbf{f}(\mathbf{x}) - \mathbf{y}|$ and Vapnik's ϵ–insensitive loss $V(\mathbf{f}(\mathbf{x}), \mathbf{y}) = \max(|\mathbf{f}(\mathbf{x}) - \mathbf{y}| - \epsilon, 0)$.

Using the ℓ_2 loss, the coefficients $\mathbf{c}_n$ of the optimal solution, i.e. Eq. (8), is then determined by a linear system [10,9]

$$(\tilde{\boldsymbol{\Gamma}} + \lambda \tilde{\mathbf{P}}^{-1})\mathbf{C} = \mathbf{Y}, \tag{10}$$

where the kernel matrix $\tilde{\boldsymbol{\Gamma}}$ is called the Gram matrix which is an $N \times N$ block matrix with the (i,j)–th block $\boldsymbol{\Gamma}(\mathbf{x}_i, \mathbf{x}_j)$. $\tilde{\mathbf{P}} = \mathbf{P} \otimes \mathbf{I}_{D \times D}$ is a $DN \times DN$ diagonal matrix, here $\mathbf{P} = \mathrm{diag}(p_1, \ldots, p_N)$, and $\otimes$ denotes Kronecker product. $\mathbf{C} = (\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_N^{\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{Y} = (\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_N^{\mathrm{T}})^{\mathrm{T}}$ are $DN \times 1$ dimensional vectors.

Note that when the loss function V is not quadratic anymore, the solution of the regularized risk functional (7) still has the form (8), but the coefficients $\mathbf{c}_n$ cannot be found anymore by solving a linear system.

2.3. Sparse approximation in vector-valued regularized least-squares

Under the Representer Theorem, the optimal solution $\mathbf{f}^o$ comes from an RKHS $\mathcal{H}_N$ defined as in Eq. (5). Finding the coefficients $\mathbf{c}_n$ of the optimal solution $\mathbf{f}^o$ in vector-valued RLS merely requires to solve the linear system (10). However, for large values of N, it may pose a serious problem due to heavy computational (i.e. scales as $O(N^3)$) or memory (i.e. scales as $O(N^2)$) requirements, and, even when it is implementable, one may prefer a suboptimal but simpler method. In this section, we propose an algorithm that is based on a similar kind of idea as the subset of regressors method [30,31] for the standard vector-valued RLS problem.

Rather than searching for the optimal solution $\mathbf{f}^o$ in $\mathcal{H}_N$, we use a sparse approximation and search a suboptimal solution $\mathbf{f}^s$ in a space $\mathcal{H}_M$ ($M \ll N$) with much less basis functions defined as

$$\mathcal{H}_M = \left\{ \sum_{m=1}^{M} \boldsymbol{\Gamma}(\cdot, \tilde{\mathbf{x}}_m)\mathbf{c}_m : \mathbf{c}_m \in \mathcal{Y} \right\}, \tag{11}$$

with $\{\tilde{\mathbf{x}}_m : m \in \mathbb{N}_M\} \subseteq \mathcal{X}$,[1] and then minimize the loss over all the training data. Yet the problem that remains is how to choose the point set $\{\tilde{\mathbf{x}}_m : m \in \mathbb{N}_M\}$, and accordingly find a set of coefficients $\{\mathbf{c}_m : m \in \mathbb{N}_M\}$. In the scalar case, different approaches for selecting this point set are discussed, for example, in [5]. There, it was found that simply selecting an arbitrary subset of the training inputs performs no worse than more sophisticated methods. Recent progress in compressed sensing [11] also demonstrated the power of sparse random basis representation. Therefore, in the interest of computational efficiency, we use the simply random sampling method to choose sparse basis functions in the vector case. And we also compare the influence of different choices of sparse basis functions in the experiment section.

According to the sparse approximation, the unique solution of the vector-valued RLS in $\mathcal{H}_M$ has this form:

$$\mathbf{f}^s(\mathbf{x}) = \sum_{m=1}^{M} \boldsymbol{\Gamma}(\mathbf{x}, \tilde{\mathbf{x}}_m)\mathbf{c}_m. \tag{12}$$

To solve the coefficients $\mathbf{c}_m$, we now consider the Hilbertian norm and the corresponding inner product (6)

$$\|\mathbf{f}\|_{\mathcal{H}}^2 = \sum_{i=1}^{M} \sum_{j=1}^{M} \langle \boldsymbol{\Gamma}(\tilde{\mathbf{x}}_j, \tilde{\mathbf{x}}_i)\mathbf{c}_i, \mathbf{c}_j \rangle_{\mathcal{Y}} = \mathbf{C}^{\mathrm{T}} \tilde{\boldsymbol{\Gamma}} \mathbf{C}, \tag{13}$$

where $\mathbf{C} = (\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_M^{\mathrm{T}})^{\mathrm{T}}$ is a $DM \times 1$ dimensional vector, the kernel matrix $\tilde{\boldsymbol{\Gamma}}$ is an $M \times M$ block matrix with the (i,j)–th block $\boldsymbol{\Gamma}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$. Thus, the minimization of the regularized risk functional (7) becomes

$$\begin{aligned} \min_{\mathbf{f} \in \mathcal{H}} \Phi(\mathbf{f}) &= \min_{\mathbf{f} \in \mathcal{H}} \left\{ \sum_{n=1}^{N} p_n \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2 + \lambda \|\mathbf{f}\|_{\mathcal{H}}^2 \right\} \\ &= \min_{\mathbf{C}} \{ \|\tilde{\mathbf{P}}^{1/2}(\mathbf{Y} - \tilde{\mathbf{U}}\mathbf{C})\|^2 + \lambda \mathbf{C}^{\mathrm{T}} \tilde{\boldsymbol{\Gamma}} \mathbf{C} \}, \end{aligned} \tag{14}$$

where $\tilde{\mathbf{U}}$ is an $N \times M$ block matrix with the (i,j)–th block $\boldsymbol{\Gamma}(\mathbf{x}_i, \tilde{\mathbf{x}}_j)$:

$$\tilde{\mathbf{U}} = \begin{bmatrix} \boldsymbol{\Gamma}(\mathbf{x}_1, \tilde{\mathbf{x}}_1) & \cdots & \boldsymbol{\Gamma}(\mathbf{x}_1, \tilde{\mathbf{x}}_M) \\ \vdots & \ddots & \vdots \\ \boldsymbol{\Gamma}(\mathbf{x}_N, \tilde{\mathbf{x}}_1) & \cdots & \boldsymbol{\Gamma}(\mathbf{x}_N, \tilde{\mathbf{x}}_M) \end{bmatrix}. \tag{15}$$

Taking the derivative of the right hand of Eq. (14) with respect to the coefficient matrix $\mathbf{C}$ and setting it to zero, we can then compute the coefficient matrix $\mathbf{C}$ from the following linear system:

$$(\tilde{\mathbf{U}}^{\mathrm{T}} \tilde{\mathbf{P}} \tilde{\mathbf{U}} + \lambda \tilde{\boldsymbol{\Gamma}})\mathbf{C} = \tilde{\mathbf{U}}^{\mathrm{T}} \tilde{\mathbf{P}} \mathbf{Y}. \tag{16}$$

In contrast to the optimal solution $\mathbf{f}^o$ given by the Representer Theorem, which is a linear combination of the basis functions $\boldsymbol{\Gamma}(\cdot, \mathbf{x}_1), \ldots, \boldsymbol{\Gamma}(\cdot, \mathbf{x}_N)$ determined by the inputs $\mathbf{x}_1, \ldots, \mathbf{x}_N$ of training samples, the suboptimal solution $\mathbf{f}^s$ is formed by a linear combination of arbitrary M-tuples of the basis functions. Generally, this sparse approximation will yield a vast increase in speed and decrease in memory requirements with negligible decrease in accuracy.

Note that the sparse approximation is somewhat related to SVM since SVM's predictive function also depends on a few

[1] Note that the point set $\{\tilde{\mathbf{x}}_m : m \in \mathbb{N}_M\}$ may not be a subset of the input training data $\{\mathbf{x}_n : n \in \mathbb{N}_N\}$.

samples (i.e. support vectors). In fact, under certain conditions, sparsity leads to SVM, which is related to the Structural Risk Minimization principle [1]. As derived in [20], when the data is noiseless, and the coefficient matrix $\mathbf{C}$ is chosen to minimize the following cost function, the sparse approximation gives the same solution of SVM:

$$\mathcal{Q}(\mathbf{C}) = \left\| \mathbf{f}(\mathbf{x}) - \sum_{n=1}^{N} \mathbf{c}_n \boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x}_n) \right\|_{\mathcal{H}}^2 + \lambda \|\mathbf{C}\|_{\ell_1}, \tag{17}$$

where $\|\cdot\|_{\ell_1}$ is the usual ℓ_1 norm. If N is very large and $\boldsymbol{\Gamma}(\cdot, \mathbf{x}_n)$ is not an orthonormal basis, it is possible that many different sets of coefficients will achieve the same error on a given data set. Among all the approximating functions that achieve the same error, using the ℓ_1 norm in the second term of Eq. (17) favors the one with the smallest number of non-zero coefficients. Our approach follows this basic idea. The difference is that in the cost function (17) the sparse basis functions are chosen automatically during the optimization process, while in our approach the basis functions are chosen randomly for the purpose of reducing both time and space complexities.

2.4. Bounds of the sparse approximation

To derive upper bounds on error of approximation of the optimal solution $\mathbf{f}^o$ by the suboptimal one $\mathbf{f}^s$, we employ Maurey–Jones–Barron's theorem [32,33] reformulated in terms of *G-variation* [34]: for a Hilbert space X with norm $\|\cdot\|$ and $\mathbf{f} \in X$, G is a bounded subset of it, the following upper bound holds

$$\|\mathbf{f} - span_n G\| \le \sqrt{\frac{(s_G \|\mathbf{f}\|_G)^2 - \|\mathbf{f}\|^2}{n}}, \tag{18}$$

where $span_n G$ is a linear combination of n arbitrary basis functions in G, $s_G = \sup_{\mathbf{g} \in G} \|\mathbf{g}\|$, and $\|\cdot\|_G$ denotes the G-variation which is defined as

$$\|\mathbf{f}\|_G = \inf\{c > 0 : \mathbf{f}/c \in cl\ conv(G \cup -G)\}, \tag{19}$$

with $cl\, conv(\cdot)$ being the closure of the convex hull of a set and $-G = \{-\mathbf{g} : g \in G\}$. Here the closure of a set A is defined as $clA = \{\mathbf{f} \in X : (\forall \epsilon > 0)(\exists \mathbf{g} \in A) \|\mathbf{f} - \mathbf{g}\| < \epsilon\}$. For properties of G-variation, we refer the reader to [34–36].

Taking advantage of this upper bound, we derive the following proposition which compares the optimal solution $\mathbf{f}^o$ with a suboptimal solution $\mathbf{f}^s$ in terms of regularized risk functional $\Phi(\mathbf{f})$.

Proposition 1. *Let* $\{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}\}_{n=1}^N$ *be a finite set of input-output pairs of data,* $\boldsymbol{\Gamma} : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$ *a matrix-valued kernel,* $s_{\boldsymbol{\Gamma}} = \sup_{\mathbf{x} \in \mathcal{X}} \|\boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x})\|_{\mathcal{Y},\mathcal{Y}}^{1/2}$, $\mathbf{f}^o(\mathbf{x}) = \sum_{n=1}^N \boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x}_n)\mathbf{c}_n$ *the optimal solution of the regularized risk functional* (7), $\mathbf{f}^s(\mathbf{x}) = \sum_{m=1}^M \boldsymbol{\Gamma}(\mathbf{x}, \tilde{\mathbf{x}}_m)\mathbf{c}_m$ *a suboptimal solution, suppose* $\forall n \in \mathbb{N}_N$, $\|\mathbf{f}^s(\mathbf{x}_n) + \mathbf{f}^o(\mathbf{x}_n) - 2\mathbf{y}_n\|_{\mathcal{Y}} \le \sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{f}^s(\mathbf{x}) + \mathbf{f}^o(\mathbf{x})\|_{\mathcal{Y}}$, *then we have the following upper bound*:

$$\Phi(\mathbf{f}^s) - \Phi(\mathbf{f}^o) \le (N s_{\boldsymbol{\Gamma}}^2 + \lambda)\left(\frac{\alpha}{M} + 2\|\mathbf{f}^o\|_{\mathcal{H}} \sqrt{\frac{\alpha}{M}}\right), \tag{20}$$

where $\alpha = (s_{\boldsymbol{\Gamma}} \|\mathbf{f}^o\|_G)^2 - \|\mathbf{f}^o\|_{\mathcal{H}}^2$, $\mathcal{H}$ *is the RKHS corresponding to the reproducing kernel* $\boldsymbol{\Gamma}$, λ *is the regularization parameter.*

Proof. According to the property (ii) of an RKHS in Section 2.1, for every $\mathbf{f} \in \mathcal{H}$ and $\mathbf{x} \in \mathcal{X}$ we have

$$\|\mathbf{f}(\mathbf{x})\|_{\mathcal{Y}} \le \|\boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x})\|_{\mathcal{Y},\mathcal{Y}}^{1/2} \|\mathbf{f}\|_{\mathcal{H}} \le s_{\boldsymbol{\Gamma}} \|\mathbf{f}\|_{\mathcal{H}}, \tag{21}$$

where $s_{\boldsymbol{\Gamma}} = \sup_{\mathbf{x} \in \mathcal{X}} \|\boldsymbol{\Gamma}(\mathbf{x}, \mathbf{x})\|_{\mathcal{Y},\mathcal{Y}}^{1/2}$. Thus we obtain

$$\sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{f}(\mathbf{x})\|_{\mathcal{Y}} \le s_{\boldsymbol{\Gamma}} \|\mathbf{f}\|_{\mathcal{H}}. \tag{22}$$

By the last inequality and Eq. (18), we obtain

$$\begin{aligned}
\Phi(\mathbf{f}^s) - \Phi(\mathbf{f}^o) &= \sum_{n=1}^{N} (\|\mathbf{f}^s(\mathbf{x}_n) - \mathbf{y}_n\|_{\mathcal{Y}}^2 - \|\mathbf{f}^o(\mathbf{x}_n) - \mathbf{y}_n\|_{\mathcal{Y}}^2) + \lambda(\|\mathbf{f}^s\|_{\mathcal{H}}^2 - \|\mathbf{f}^o\|_{\mathcal{H}}^2) \\
&\le \sum_{n=1}^{N} \langle \mathbf{f}^s(\mathbf{x}_n) - \mathbf{f}^o(\mathbf{x}_n), \mathbf{f}^s(\mathbf{x}_n) + \mathbf{f}^o(\mathbf{x}_n) - 2\mathbf{y}_n \rangle_{\mathcal{Y}} \\
&\quad + \lambda \big| \|\mathbf{f}^s\|_{\mathcal{H}} - \|\mathbf{f}^o\|_{\mathcal{H}} \big| \cdot (\|\mathbf{f}^s\|_{\mathcal{H}} + \|\mathbf{f}^o\|_{\mathcal{H}}) \\
&\le \sum_{n=1}^{N} \|\mathbf{f}^s(\mathbf{x}_n) - \mathbf{f}^o(\mathbf{x}_n)\|_{\mathcal{Y}} \|\mathbf{f}^s(\mathbf{x}_n) + \mathbf{f}^o(\mathbf{x}_n) - 2\mathbf{y}_n\|_{\mathcal{Y}} \\
&\quad + \lambda \|\mathbf{f}^s - \mathbf{f}^o\|_{\mathcal{H}} (\|\mathbf{f}^s\|_{\mathcal{H}} + \|\mathbf{f}^o\|_{\mathcal{H}}) \\
&\le N \sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{f}^s(\mathbf{x}) - \mathbf{f}^o(\mathbf{x})\|_{\mathcal{Y}} \sup_{\mathbf{x} \in \mathcal{X}} \|\mathbf{f}^s(\mathbf{x}) + \mathbf{f}^o(\mathbf{x})\|_{\mathcal{Y}} \\
&\quad + \lambda \|\mathbf{f}^s - \mathbf{f}^o\|_{\mathcal{H}} (\|\mathbf{f}^s\|_{\mathcal{H}} + \|\mathbf{f}^o\|_{\mathcal{H}}) \\
&\le N s_{\boldsymbol{\Gamma}} \|\mathbf{f}^o - \mathbf{f}^s\|_{\mathcal{H}} | s_{\boldsymbol{\Gamma}} \|\mathbf{f}^o + \mathbf{f}^s\|_{\mathcal{H}} - 2\mathbf{y}_{\min} | \\
&\quad + \lambda \|\mathbf{f}^o - \mathbf{f}^s\|_{\mathcal{H}} (\|\mathbf{f}^o\|_{\mathcal{H}} + \|\mathbf{f}^s\|_{\mathcal{H}}) \\
&\le N s_{\boldsymbol{\Gamma}} \sqrt{\frac{\alpha}{M}} \left(2 s_{\boldsymbol{\Gamma}} \|\mathbf{f}^o\|_{\mathcal{H}} + s_{\boldsymbol{\Gamma}} \sqrt{\frac{\alpha}{M}}\right) + \lambda \sqrt{\frac{\alpha}{M}} \left(2\|\mathbf{f}^o\|_{\mathcal{H}} + \sqrt{\frac{\alpha}{M}}\right) \\
&= (N s_{\boldsymbol{\Gamma}}^2 + \lambda)\left(\frac{\alpha}{M} + 2\|\mathbf{f}^o\|_{\mathcal{H}} \sqrt{\frac{\alpha}{M}}\right),
\end{aligned} \tag{23}$$

where $\alpha = (s_{\boldsymbol{\Gamma}} \|\mathbf{f}^o\|_G)^2 - \|\mathbf{f}^o\|_{\mathcal{H}}^2$, $\|\cdot\|_G$ corresponds to the G-variation, and G corresponds to $\mathcal{H}_N$ in our problem. □

From this upper bound, we can easily derive that to achieve an approximation accuracy ϵ, i.e. $\Phi(\mathbf{f}^s) - \Phi(\mathbf{f}^o) \le \epsilon$, the needed minimal number of basis functions satisfies

$$M \le \alpha \left[\frac{\epsilon}{N s_{\boldsymbol{\Gamma}}^2 + \lambda} - 2\|\mathbf{f}^o\|_{\mathcal{H}}^2 \left(\sqrt{1 + \frac{\epsilon}{(N s_{\boldsymbol{\Gamma}}^2 + \lambda)\|\mathbf{f}^o\|_{\mathcal{H}}^2}} - 1 \right) \right]^{-1}. \tag{24}$$

3. Sparse approximation for robust vector field learning

A vector field is also a vector-valued function. The Tikhonov regularization treats all samples as inliers which ignores the issue of robustness, i.e., the real-world data may often contain some unknown outliers. Recently, Zhao et al. [9] present a robust vector-valued RLS method named Vector Field Consensus (VFC) for vector field learning. In which, each sample is associated with a latent variable indicating whether it is an inlier or an outlier, and then the EM algorithm is adopted for optimization. Besides, the technique of robust vector field learning has been also adopted in Gaussian processes, basically by using the so-called t-processes [37,38]. In this section, we present a sparse approximation algorithm for VFC, and apply it to the mismatch removal problem.

3.1. Sparse vector field consensus

Given a set of observed input–output pairs $S = \{(\mathbf{x}_n, \mathbf{y}_n) : n \in \mathbb{N}_N\}$ as samples randomly drawn from a vector field which may contain some unknown outliers, the goal is to learn a mapping $\mathbf{f}$ to fit the inliers well. Due to the existence of outliers, it is desirable to have a robust estimate of the mapping $\mathbf{f}$. There are two choices: (i) to build a more complex model that includes the outliers—which involves modeling the outlier process using extra (hidden) variables which enable us to identify and reject outliers, or (ii) to use an estimator which is less sensitive to outliers, as described in Huber's robust statistics [39]. In this paper, we use the first scenario. In the following we make an assumption that, the vector field samples in the inlier class have Gaussian noise with zero mean and uniform standard deviation σ, while the ones in the outlier class are uniformly distributed $1/a$ with a being the volume of the output domain. Let γ be the percentage of inliers which we do not know in advance, $\mathbf{X} = (\mathbf{x}_1^T, \ldots, \mathbf{x}_N^T)^T$ and $\mathbf{Y} = (\mathbf{y}_1^T, \ldots, \mathbf{y}_N^T)^T$ be $DN \times 1$ dimensional vectors, and $\boldsymbol{\theta} = \{\mathbf{f}, \sigma^2, \gamma\}$ the unknown parameter set. The likelihood is then a mixture model as

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} \left(\frac{\gamma}{(2\pi\sigma^2)^{D/2}} e^{-\|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2/2\sigma^2} + \frac{1-\gamma}{a} \right). \tag{25}$$

We model the mapping $\mathbf{f}$ in a vector-valued RKHS $\mathcal{H}$ with reproducing kernel $\boldsymbol{\Gamma}$, and impose a smoothness constraint on it, i.e. $p(\mathbf{f}) \propto e^{-(\lambda/2)\|\mathbf{f}\|_{\mathcal{H}}^2}$. Therefore, we can estimate a MAP solution of $\boldsymbol{\theta}$ by using the Bayes rule as $\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})p(\mathbf{f})$. An iterative EM algorithm can be used to solve this problem. We associate sample n with a latent variable $z_n \in \{0, 1\}$, where $z_n = 1$ indicates a Gaussian distribution and $z_n = 0$ points to a uniform distribution. We follow the standard notations [40] and omit some terms that are independent of $\boldsymbol{\theta}$. The complete-data log posterior is then given by

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = & -\frac{1}{2\sigma^2} \sum_{n=1}^{N} P(z_n = 1|\mathbf{x}_n, \mathbf{y}_n, \boldsymbol{\theta}^{\text{old}}) \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2 \\ & -\frac{D}{2} \ln \sigma^2 \sum_{n=1}^{N} P(z_n = 1|\mathbf{x}_n, \mathbf{y}_n, \boldsymbol{\theta}^{\text{old}}) \\ & + \ln(1-\gamma) \sum_{n=1}^{N} P(z_n = 0|\mathbf{x}_n, \mathbf{y}_n, \boldsymbol{\theta}^{\text{old}}) \\ & + \ln \gamma \sum_{n=1}^{N} P(z_n = 1|\mathbf{x}_n, \mathbf{y}_n, \boldsymbol{\theta}^{\text{old}}) - \frac{\lambda}{2} \|\mathbf{f}\|_{\mathcal{H}}^2. \end{aligned} \tag{26}$$

E-step: Denote $\mathbf{P} = \text{diag}(p_1, \ldots, p_N)$, where the probability $p_n = P(z_n = 1|\mathbf{x}_n, \mathbf{y}_n, \boldsymbol{\theta}^{\text{old}})$ can be computed by applying Bayes rule

$$p_n = \frac{\gamma e^{-\|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2/2\sigma^2}}{\gamma e^{-\|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2/2\sigma^2} + (1-\gamma) \frac{(2\pi\sigma^2)^{D/2}}{a}}. \tag{27}$$

The posterior probability p_n is a soft decision, which indicates to what degree the sample n agrees with the current estimated vector field $\mathbf{f}$.

M-step: We determine the revised parameter estimate $\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$. Taking derivative of $\mathcal{Q}(\boldsymbol{\theta})$ with respect to σ^2 and γ, and setting them to zero, we obtain

$$\sigma^2 = \frac{(\mathbf{Y} - \mathbf{V})^T \tilde{\mathbf{P}} (\mathbf{Y} - \mathbf{V})}{D \cdot \text{tr}(\mathbf{P})}, \tag{28}$$

$$\gamma = \text{tr}(\mathbf{P})/N, \tag{29}$$

where $\mathbf{V} = (\mathbf{f}(\mathbf{x}_1)^T, \ldots, \mathbf{f}(\mathbf{x}_N)^T)^T$ is a $DN \times 1$ dimensional vector, and $\text{tr}(\cdot)$ denotes the trace.

Considering the terms of objective function $\mathcal{Q}$ in Eq. (26) that are related to $\mathbf{f}$, the vector field can be estimated from minimizing an energy function as

$$\mathcal{E}(\mathbf{f}) = \frac{1}{2\sigma^2} \sum_{n=1}^{N} p_n \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2 + \frac{\lambda}{2} \|\mathbf{f}\|_{\mathcal{H}}^2. \tag{30}$$

This is a regularized risk functional (7) with ℓ_2 loss function (9). To estimate the vector field $\mathbf{f}$, it needs to solve a linear system similar to Eq. (10), which takes up most of the run-time and memory requirements of the algorithm. Obviously, the sparse approximation could be used here to reduce the time and space complexity.

Using the sparse approximation, we search a suboptimal $\mathbf{f}^s$ which has the form as in Eq. (12) with the coefficient $\mathbf{c}_n$ determined by a linear system similar to Eq. (16)

$$(\tilde{\mathbf{U}}^T \tilde{\mathbf{P}} \tilde{\mathbf{U}} + \lambda\sigma^2 \tilde{\boldsymbol{\Gamma}})\mathbf{C} = \tilde{\mathbf{U}}^T \tilde{\mathbf{P}} \mathbf{Y}. \tag{31}$$

Since it is a sparse approximation to the vector field consensus algorithm, we name our method *SparseVFC*. In summary, compared with the original VFC algorithm, we estimate a vector field $\mathbf{f}$ by solving a linear system (31) in SparseVFC rather than Eq. (10) in VFC. Our SparseVFC algorithm is outlined in Algorithm 1.

Table 1
Comparison of computational complexity.

	VFC [9]	FastVFC [9]	SparseVFC
Time	$O(N^3)$	$O(N^3)$	$O(N)$
Space	$O(N^2)$	$O(N^2)$	$O(N)$

Algorithm 1. The SparseVFC Algorithm.

Input: Training set $S = \{(\mathbf{x}_n, \mathbf{y}_n) : n \in \mathbb{N}_N\}$, matrix kernel $\boldsymbol{\Gamma}$, regularization constant λ, basis function number M
Output: Vector field $\mathbf{f}$, inlier set $\mathcal{I}$
1 Initialize a, γ, $\mathbf{V} = \mathbf{0}_{DN\times 1}$, $\mathbf{P} = \mathbf{I}_{N\times N}$, and σ^2 by equation (28)
2 Randomly choose M basis functions from the training inputs and construct kernel matrix $\tilde{\boldsymbol{\Gamma}}$
3 **repeat**
4 *E–step* :
5 Update $\mathbf{P} = \text{diag}(p_1, \ldots, p_N)$ by equation (27)
6 *M–step* :
7 Update $\mathbf{C}$ by solving linear system (31)
8 Update $\mathbf{V}$ by using equation (12)
9 Update σ^2 and γ by equations (28)and(29)
10 **until** $\mathcal{Q}$ *converges*
11 Vector field $\mathbf{f}$ is determined by equation (12)
12 The inlier set is $\mathcal{I} = \{n : p_n > \tau, n \in \mathbb{N}_N\}$ with τ being a predefined threshold.

It also presented a fast implementation for VFC named FastVFC in the original paper [9]. To reduce the time complexity, it first uses the low rank matrix approximation by computing the singular value decomposition (SVD) for the kernel matrix $\tilde{\boldsymbol{\Gamma}}$ of size $DN \times DN$, and then the Woodbury matrix identity to invert the coefficient matrix in the linear system for solving $\mathbf{C}$ [41].

Computational complexity. Notice that $\tilde{\mathbf{P}}$ is a diagonal matrix, we can compute $\tilde{\mathbf{U}}^T \tilde{\mathbf{P}}$ in linear system (31) by multiplying the n-th diagonal element of $\tilde{\mathbf{P}}$ to the n-th column of $\tilde{\mathbf{U}}^T$, and thus the time complexity of SparseVFC for mismatch removal is reduced to $O(mD^3M^2N + mD^3M^3)$, where m is the iterative times for EM. The space complexity of SparseVFC scales like $O(D^2MN + D^2M^2)$ due to the memory requirements for storing the matrix $\tilde{\mathbf{U}}$ and kernel $\tilde{\boldsymbol{\Gamma}}$.

Typically, the required number of basis functions for sparse approximation is much less than the number of data points, i.e. $M \ll N$. In this paper, we apply the SparseVFC to the mismatch removal problem on 2D and 3D images, in which the number of the point matches N is typically in the order of 10^3, and the required number of basis function M is in the order of 10^1. Therefore, both the time and space complexities of SparseVFC could be simply written as $O(N)$. Table 1 summarizes the time and space complexities of VFC, FastVFC and SparseVFC. Compared to VFC and FastVFC, the time and space complexities are reduced from $O(N^3)$ and $O(N^2)$ to both $O(N)$. This is significant for large training sets. Note that the time complexity of FastVFC is still $O(N^3)$, it is because the SVD operation of a matrix of size $N \times N$ has time complexity $O(N^3)$.

Relation to FastVFC. There is some relationship between our SparseVFC algorithm and the FastVFC algorithm. On the one hand, both these two algorithms use approximation to the matrix-valued kernel $\boldsymbol{\Gamma}$ to search a suboptimal solution rather than the optimal solution, and then reduce the time complexity. On the other hand, our algorithm is clearly superior to the FastVFC, which can be seen from both the time and space complexities as shown in Table 1. For FastVFC, it makes a low rank matrix approximation on the kernel matrix $\tilde{\boldsymbol{\Gamma}}$ itself, which does not change the size of $\tilde{\boldsymbol{\Gamma}}$.

Therefore, the memory requirement is not decreased, and it is still not implementable for large training sets. Moreover, the FastVFC algorithm has the same time complexity $O(N^3)$ as VFC, due to the SVD operation and kernel matrix inversion operation in FastVFC and VFC, respectively. The difference is that the SVD operation in FastVFC needs to perform only once while the matrix inversion operation in VFC needs to perform in each EM iteration, and hence an acceleration can be achieved in FastVFC. In contrast, the SparseVFC uses a sparse representation and chooses much less basis functions to approximate the function space, leading to a significant reduction of the size of the corresponding reproducing kernel. This sparse approximation (even with random chosen basis functions) not only significantly reduces both the time and space complexities, but also does not lead to sacrifice in accuracy, and in some situations it even gains a little better performance compared to the original VFC algorithm (as shown in our experiments).

3.2. Application to mismatch removal

In this section, we focus on establishing accurate point correspondences between two images of the same scene. Many of the computer vision tasks such as building 3D models, registration, object recognition, tracking, and structure and motion recovery start by assuming that the point correspondences have been successfully recovered [42].

Point correspondences between two images are in general established by first detecting interest points and then matching the detected points based on local descriptors [43]. This may result in a number of mismatches (outliers) due to viewpoint changes, occlusions, repeated structures, etc. The existence of mismatches is usually enough to ruin the traditional estimation methods. In this case, a robust estimator is desirable to remove mismatches [44–49]. In our SparseVFC, as shown in the last line of Algorithm 1, a sample being an inlier or outlier could be determined by its posterior probability after EM convergences. Using this property, we apply SparseVFC to the mismatch removal problem. Next, we point out some key issues.

Vector field introduced by image pairs. We first make a linear rescaling of the point correspondences so that the positions of feature points in the first and second images both have zero mean and unit variance. Let the input $\mathbf{x}\in\mathbb{R}^P$ be the location of a normalized point in the first image, and the output $\mathbf{y}\in\mathbb{R}^D$ be the corresponding displacement of that point in the second image; then the matches can be converted into motion field training set. For 2D images $P=D=2$; for 3D surfaces $P=D=3$. Fig. 1 illustrates schematically the 2D image case.

Kernel selection. Kernel plays a central role in regularization theory as it provides a flexible and computationally feasible way to choose an RKHS. Usually, for the mismatch removal problem, the structure of the generated vector field is relatively simple. We simply choose a diagonal decomposable kernel [8,9]: $\mathbf{\Gamma}(\mathbf{x}_i,\mathbf{x}_j)=e^{-\beta\|\mathbf{x}_i-\mathbf{x}_j\|^2}\mathbf{I}$. Then we can solve a more efficient linear system instead of Eq. (31) as

$$(\mathbf{U}^{\mathrm{T}}\mathbf{P}\mathbf{U}+\lambda\sigma^2\mathbf{\Gamma})\tilde{\mathbf{C}}=\mathbf{U}^{\mathrm{T}}\mathbf{P}\tilde{\mathbf{Y}}, \tag{32}$$

where the kernel matrix $\mathbf{\Gamma}\in\mathbb{R}^{M\times M}$ and $\mathbf{\Gamma}_{ij}=e^{-\beta\|\tilde{\mathbf{x}}_i-\tilde{\mathbf{x}}_j\|^2}$, $\mathbf{U}\in\mathbb{R}^{N\times M}$ and $\mathbf{U}_{ij}=e^{-\beta\|\mathbf{x}_i-\tilde{\mathbf{x}}_j\|^2}$, $\tilde{\mathbf{C}}=(\mathbf{c}_1,\ldots,\mathbf{c}_M)^{\mathrm{T}}$ and $\tilde{\mathbf{Y}}=(\mathbf{y}_1,\ldots,\mathbf{y}_N)^{\mathrm{T}}$ are $M\times D$ and $N\times D$ matrices, respectively. Here N is the number of putative matches, and M is the number of bases. It should be noted that solving the vector field learning problem with this diagonal decomposable kernel is not equivalent to solving D independent scalar problems, since the update of the posterior probability p_n and variance σ^2 in Eqs. (27) and (28) are determined by all components of the output $\mathbf{y}_n$.

When it is applied to mismatch removal, there is a problem which should draw attention. We must ensure that the point set $\{\tilde{\mathbf{x}}_m : m\in\mathbb{N}_M\}$ used to construct the basis functions does not contain two same points since in this case the coefficient matrix in linear system (32), i.e. $(\mathbf{U}^{\mathrm{T}}\mathbf{P}\mathbf{U}+\lambda\sigma^2\mathbf{\Gamma})$, will be singular. Obviously, this may appear in the mismatch removal problem, since in the putative match set there may exist one point in the first image matched to several points in the second image.

3.3. Implementation details

There are mainly four parameters in the SparseVFC algorithm: γ, λ, τ and M. Parameter γ reflects our initial assumption on the amount of inliers in the correspondence sets. Parameter λ reflects the amount of the smoothness constraint which controls the trade-off between the closeness to the data and the smoothness of the solution. Parameter τ is a threshold, which is used for deciding the correctness of a match. In general, our method is very robust to these parameters. We set $\gamma=0.9$, $\lambda=3$, and $\tau=0.75$ according to the original VFC algorithm throughout this paper. Parameter M is the number of the basis functions used for sparse approximation. The choice of M depends on both the data (i.e., the true vector field) and the assumed function space (i.e., the reproducing kernel), as shown in Eq. (24). We will discuss it in the experiment according to the specific application.

It should be noted that in practice we do not determine the value of M according to Eq. (24). On the one hand, it is derived in the context of providing a theoretic upper bound, and in practice to achieve a good approximation the required value of M may be much smaller than this bound. On the other hand, the upper bound in Eq. (24) is hard to compute, and it is costly to derive a value of M for each sample set.

4. Experimental setup

In our evaluation we consider synthetic 2D vector field estimation, mismatch removal on 2D real images and 3D surfaces. All the experiments are performed on a Intel Core2 2.5GHz PC with Matlab code. Next, we discuss about the datasets and evaluation criteria.

Synthetic 2D vector field: The synthetic vector field is constructed from a scalar function defined by a mixture of five Gaussians, which have the same covariance $0.25I$ and centered at (0, 0), (1, 0), (0, 1), (−1, 0) and (0, −1), respectively, as in [8]. Its

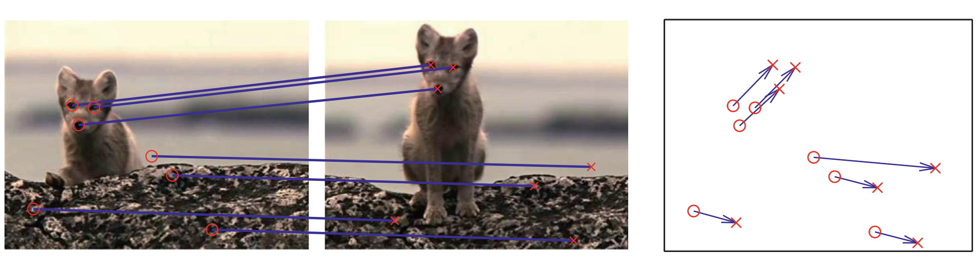

Fig. 1. Schematic illustration of motion field introduced by image pairs. Left: an image pair and its putative matches; right: motion field samples introduced by the point matches in the left figure. $\circ$ and $\times$ indicate feature points in the first and second images, respectively.

gradient and perpendicular gradient indicate a divergence-free and a curl-free field, respectively. The synthetic data is then constructed by taking a convex combination of these two vector fields. In our evaluation the combination coefficient is set to 0.5, and then we get the synthetic field as shown in Fig. 2. The field is computed on a 70×70 grid over the square $[-2,2]\times[-2,2]$.

The training inliers are uniformly sampled points from the grid, and we add Gaussian noise with zero mean and uniform standard deviation 0.1 on the outputs. The outliers are generated as follows: the input $\mathbf{x}$ is chosen randomly from the grid; the output $\mathbf{y}$ is generated randomly from a uniform distribution on the square $[-2,2]\times[-2,2]$. The performance for vector field learning is measured by an angular measure of error [50] between the learned vector of VFC and the ground truth. If $\mathbf{v}_g=(v_g^1,v_g^2)$ and $\mathbf{v}_e=(v_e^1,v_e^2)$ are the ground truth and estimated fields, we consider the transformation $\mathbf{v}\rightarrow\tilde{\mathbf{v}}=1/(\|(v^1,v^2,1)\|)(v^1,v^2,1)$. The error measure is defined as $err=\arccos(\tilde{\mathbf{v}}_e,\tilde{\mathbf{v}}_g)$.

2D image datasets: We tested our method on the dataset of Mikolajczyk et al. [51] and Tuytelaars and van Gool [52], and several image pairs of non-rigid objects. The images in the first dataset are either of planar scenes or the camera position was fixed during acquisition. Therefore, the images are always related by a homography. The test data of Tuytelaars et al. contains several wide baseline image pairs. The image pairs of non-rigid object are made in this paper.

The open source VLFeat toolbox [53] is used to determine the initial correspondences of SIFT [43]. All parameters are set as the default values except for the distance ratio threshold t. Usually, the greater value of t is, the smaller amount of matches with higher correct match percentage will be. The match correctness is determined by computing an overlap error [51], as in [9].

3D surfaces datasets: For 3D case, we consider two datasets used in [54]: the *Dino* and *Temple* datasets. Each surface pair are representations of the same rigid object which can be aligned using a rotation, translation and scale.

We determine the initial correspondences by using the method of Zaharescu et al. [54]. The feature point detector is called MeshDOG, which is a generalization of the difference of Gaussian (DOG) operator [43]. The feature descriptor is called MeshHOG, and it is a generalization of the histogram of oriented gradients (HOG) descriptor [55]. The match correctness is determined as follows. For these two datasets, the correspondences between the two surfaces can be formulated as $\mathbf{y}=s\mathbf{R}\mathbf{x}+\mathbf{t}$, where $\mathbf{R}_{3\times3}$ is a rotation matrix, s is a scaling parameter, and $\mathbf{t}_{3\times1}$ is a translation vector. We can use some robust rigid point registration methods such as the Coherent Point Drift (CPD) [41] to solve these three parameters, and then the match correctness can be accordingly determined.

5. Experimental results

To test the performance of the sparse approximation algorithm, we perform experiments on the proposed SparseVFC algorithm, and compare its approximate accuracy and time efficiency to VFC and FastVFC on both synthetic data and real-world images. The experiments are conducted from two aspects: (i) vector field learning performance comparison on synthetic vector field and (ii) mismatch removal performance comparison on real image datasets.

5.1. Vector field learning on synthetic vector field

We perform a representative experiment on a synthetic 2D vector field shown in Fig. 2. For each cardinality of the training set, we repeat the experimental process by 10 times with different randomly drawn examples. After the vector field is learned, we use it to predict the outputs on the whole grid and compare them to the ground truth. The experimental results are evaluated by means of test error (i.e. error between prediction and ground truth on the whole grid) and run-time speedup factor (i.e. ratio of run-time of VFC to run-time of FastVFC or SparseVFC) for clarity. The matrix-valued kernel is chosen to be a convex combination of the divergence-free kernel $\mathbf{\Gamma}_{df}$ and curl-free kernel $\mathbf{\Gamma}_{cf}$ [56] with width $\tilde{\sigma}=0.8$, and the combination coefficient is set to 0.5. The divergence-free and curl-free kernels have the following forms:

$$\mathbf{\Gamma}_{df}(\mathbf{x},\mathbf{x}')=\frac{1}{\tilde{\sigma}^2}e^{-\|\mathbf{x}-\mathbf{x}'\|^2/2\tilde{\sigma}^2}\left[\left(\frac{\mathbf{x}-\mathbf{x}'}{\tilde{\sigma}}\right)\left(\frac{\mathbf{x}-\mathbf{x}'}{\tilde{\sigma}}\right)^{\mathrm{T}}+\left((D-1)-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\tilde{\sigma}^2}\right)\cdot I\right], \tag{33}$$

$$\mathbf{\Gamma}_{cf}(\mathbf{x},\mathbf{x}')=\frac{1}{\tilde{\sigma}^2}e^{-\|\mathbf{x}-\mathbf{x}'\|^2/2\tilde{\sigma}^2}\left[I-\left(\frac{\mathbf{x}-\mathbf{x}'}{\tilde{\sigma}}\right)\left(\frac{\mathbf{x}-\mathbf{x}'}{\tilde{\sigma}}\right)^{\mathrm{T}}\right]. \tag{34}$$

In order to choose an appropriate value of M on this synthetic field, we assess the performance of SparseVFC with respect to M under different experimental setup. Rather than using the ground truth to assess performance, i.e. the mean of test error, we use the standard deviation of the estimated inliers [44]

$$\sigma^*=\left(\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\|\mathbf{y}_i-\mathbf{f}(\mathbf{x}_i)\|^2\right)^{1/2}, \tag{35}$$

where $\mathcal{I}$ is the estimated inlier set in the training samples, and $|\cdot|$ denotes the cardinality of a set. Generally speaking, smaller value of σ^* indicates better estimation. The result is presented in Fig. 3a, we can see that $M=60$ could reach a good compromise between

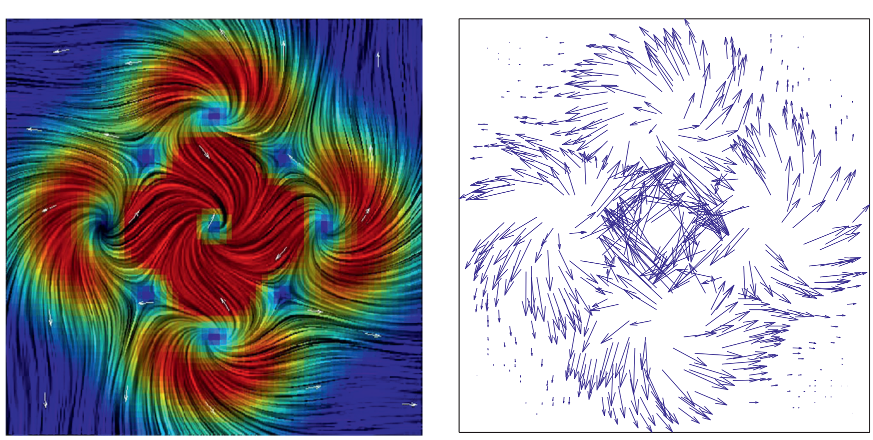

Fig. 2. Visualization of a synthetic 2D field without noise and its 500 random samples used in experiment.

the accuracy and computational complexity. For comparison, we also present the mean of test error in Fig. 3b. We can see that these two criterions tend to produce similar choices of M. For FastVFC, 150 largest eigenvalues are used for calculating the low rank matrix approximation.

We first give some intuitive impression on the performance of our method, as shown in Fig. 4. We see that SparseVFC can successfully recover the vector field from sparse training samples, and the performance is getting better as the number of inlier in the training set increases. Figs. 5 and 6 show the vector field learning performances of VFC, FastVFC and SparseVFC. We consider two scenarios for performance comparison: (i) fix the inlier number and change the inlier percentage and (ii) fix the inlier percentage and change the inlier number. Fig. 5 shows the performance comparison of the test error under different experimental setup, we see that both SparseVFC and FastVFC perform much the same as VFC. Fig. 6 shows a performance comparison on the average run-time. From Fig. 6a, we see that the computational cost of SparseVFC is about linear with respect to the training set size. The run-time speedup factors of FastVFC and SparseVFC with respect to VFC are presented in Fig. 6b. We see that the speedup factor of FastVFC is about a constant, since its time complexity is the same as VFC, both are $O(N^3)$. However,

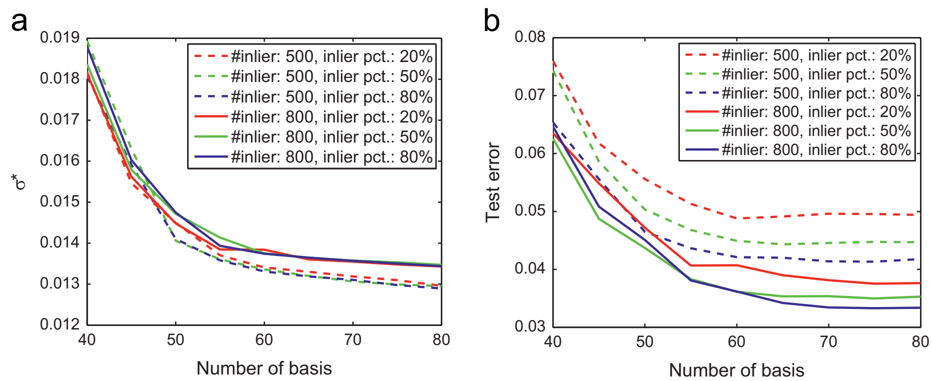

Fig. 3. Experiments for choosing the sample point number M used for sparse approximation. (a) The standard deviation of the estimated inliers σ^* with respect to M. (b) The test error with respect to M. The inlier number in the training set is set to 500 and 800, and the inlier percentage is varied among 20%, 50% and 80%.

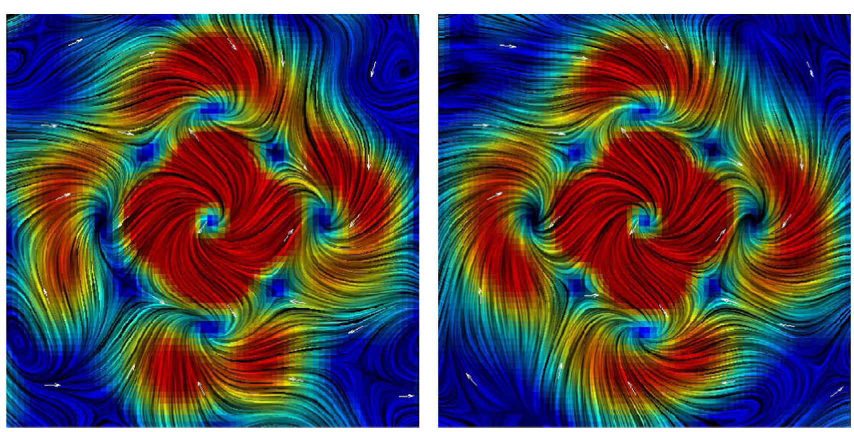

Fig. 4. Reconstruction of field learning shown in Fig. 2a via SparseVFC. Left: 200 inliers contained in the training set; right: 500 inliers contained in the training set. The inlier ratios are both 0.5. The means of test error is about 0.072 and 0.044, respectively.

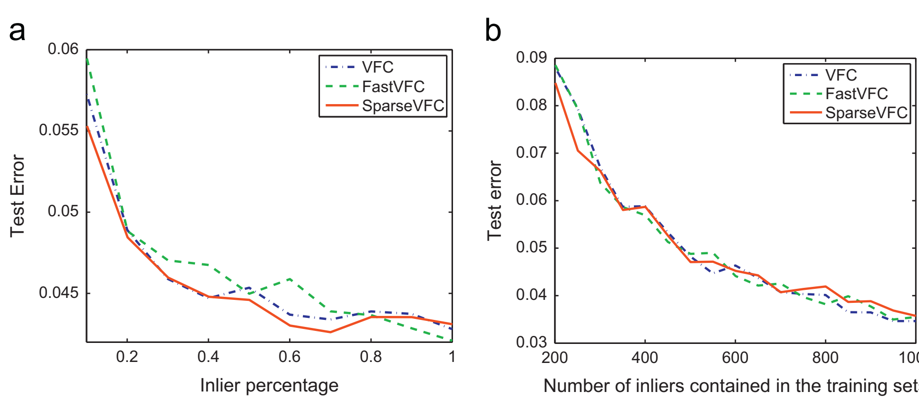

Fig. 5. Performance comparison on test error. (a) Comparison of test error through changing the inlier ratio in the training sets, the inlier number is fixed to 500. (b) Comparison of test error through changing the inlier number in the training sets, the inlier ratio is fixed to 0.2.

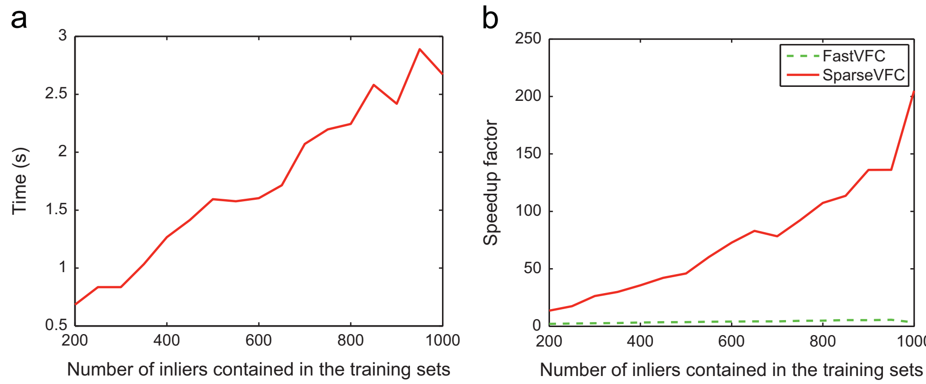

Fig. 6. Performance comparison on run-time under the experimental setup in Fig. 5b. (a) Run-time of SparseVFC. (b) Run-time speedup of FastVFC and SparseVFC with respect to VFC.

compared to FastVFC, the use of SparseVFC leads to an essential speedup, and this advantage will be significantly magnified with larger scale of training set.

In conclusion, when an appropriate number of basis functions are chosen, the SparseVFC algorithm can approximate the original VFC algorithm quite well, while with a significant run-time speedup.

5.2. Mismatch removal on real images

We notice that the algorithm demonstrates strong ability on mismatch removal. Thus we apply it to the mismatch removal problem and perform experiments on a wide range of real images. The performance is characterized by precision and recall. Besides VFC and FastVFC, we use three additional mismatch removal methods for comparison: RANdom SAmple Consensus (RANSAC) [57], Maximum Likelihood Estimation SAmple Consensus (MLESAC) [44] and Identifying point correspondences by Correspondence Function (ICF) [47]. RANSAC tries to get as small an outlier-free subset as feasible to estimate a given parametric model by resampling, while its variants MLESAC adopts a new cost function using weighted voting strategy based on M-estimator, and chooses the solution which maximize the likelihood rather than the inlier count in RANSAC. The ICF method uses support vector regression to learn a correspondence function pair which maps points in one image to their corresponding points in another, and then rejects the mismatches by checking whether they are consistent with the estimated correspondence functions.

The structure of the generated motion field is relatively simple in the mismatch removal problem. We simply choose a diagonal decomposable kernel with Gaussian form in the scalar part, and we set $\beta=0.1$ according to the original VFC algorithm. Moreover, the number of basis functions M used for sparse approximation is fixed in our evaluation, since choosing M adaptively would require some pre-processing which would increase the computational cost. We manage to tune the value of M by using a small test set, and we find that using 15 basis functions for sparse approximation is accurate enough. So we set $M=15$ throughout the experiments. For FastVFC, 10 largest eigenvalues are used for calculating the low rank matrix approximation.

5.2.1. Results on several 2D image pairs

Our first experiment involves mismatch removal on several image pairs, including wide three baseline image pairs (*Tree*, *Valbonne* and *Mex*) and three image pairs of non-rigid object (*DogCat*, *Peacock* and *T-shirt*). Table 2 presents the numbers of matches as well as the initial correct match percentages in these six image pairs.

Table 2
The numbers of matches and initial inlier percentages in the six image pairs.

	Tree	*Valbonne*	*Mex*	*DogCat*	*Peacock*	*T-shirt*
No. of matches	167	126	158	113	236	300
Inlier pct. (%)	56.29	54.76	51.90	82.30	71.61	60.67

The whole mismatch removal progress on these image pairs is illustrated schematically in Fig. 7. The columns show the iterative progress, the level of the blue color indicates to what degree a sample belongs to inlier, and it is also the posterior probability p_n in Eq. (30). In the beginning, all the SIFT matches in the first column are assumed to be inlier. We convert them into motion field training sets which are shown in the 2nd column. As the EM iterative process continues, progressively more refined matches are shown in the 3rd, 4th and 5th columns. The 5th column shows that the algorithm almost converges to a nearly binary decision on the match correctness. The SIFT matches reserved by the algorithm are presented in the last column. It should be noted that there is an underline assumption in our method that the vector field should be smooth, i.e. the norm of the field $\mathbf{f}$ in Eq. (13) should be small. However, for wide baseline image pairs such as *Tree*, *Valbonne* and *Mex*, the related motion fields are in general not continuous, and our method is still effective for mismatch removal. We give an explanation as follows: under the sparsity assumptions of the training data, it is not hard to seek a smooth vector field which can fit nearly all the inliers (sparse) well; if the goal is just to remove mismatches in the training data, the smoothness constraint can work well.

Next, we give a performance comparison with the other five methods in Table 3. The geometry model used in RANSAC and MLESAC is epipolar geometry. We see that MLESAC has slightly better precisions than the RANSAC with the cost of producing a slightly lower recall. The recall of ICF is quite low, although it has a satisfactory precision. However, VFC, FastVFC and SparseVFC can successfully distinguish inliers from outliers, and they have the best trade-off between precision and recall. The low rank matrix approximation used in FastVFC may slightly hurt the performance. While in SparseVFC, it seems that the sparse approximation does not lead to degenerated performance, on the contrary, it makes the algorithm more efficient.

Notice that we did not compare to RANSAC and MLESAC on the image pairs of non-rigid object. RANSAC and MLESAC depend on a parametric model, for example, fundamental matrix. If there exist some objects with deformation in the image pairs, they can no longer work, since the point pairs will no longer obey the epipolar geometry.

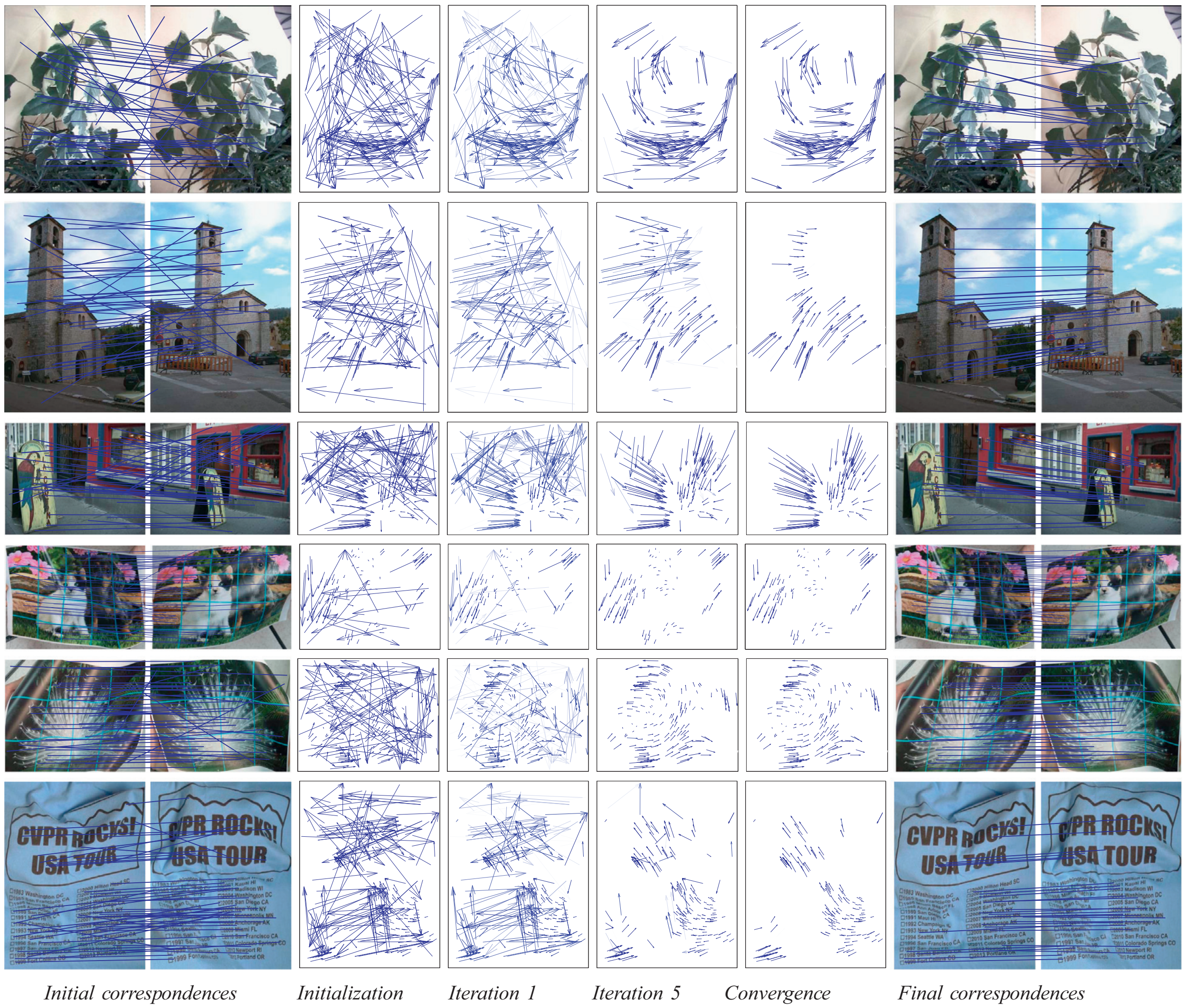

Fig. 7. Results on image pairs of *Tree*, *Valbonne*, *Mex*, *DogCat*, *Peacock* and *T-shirt*. The first three rows are wide baseline image pairs, and the rest three are image pairs of non-rigid object. The columns show the iterative mismatch removal progress, and the level of the blue color indicates to what degree a sample belongs to inlier. For visibility, only 50 randomly selected matches are presented in the first column. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Table 3
Performance comparison with different mismatch removal algorithms. The pairs in the table are precision–recall pairs (%).

	RANSAC [57]	MLESAC [44]	ICF [47]	VFC [9]	FastVFC [9]	SparseVFC
Tree	(94.68, 94.68)	(98.82, 89.36)	(92.75, 68.09)	(94.85, 97.87)	(94.79, 96.81)	(94.85, 97.87)
Valbonne	(94.52, 100.00)	(94.44, 98.55)	(91.67, 63.77)	(98.33, 85.51)	(98.33, 85.51)	(98.33, 85.51)
Mex	(91.76, 95.12)	(93.83, 92.68)	(96.15, 60.98)	(96.47, 100.00)	(96.47, 100.00)	(96.47, 100.00)
DogCat	–	–	(92.19, 63.44)	(100.00, 100.00)	(100.00, 100.00)	(100.00, 100.00)
Peacock	–	–	(99.12, 66.86)	(99.40, 98.82)	(99.40, 98.22)	(99.40, 98.82)
T-shirt	–	–	(99.07, 58.79)	(98.88, 96.70)	(98.84, 93.41)	(98.88, 96.70)

5.2.2. Results on a 2D image datasets

We test our method on the dataset of Mikolajczyk et al., which contains image transformations of viewpoint change, scale change, rotation, image blur, JPEG compression, and illumination. We use all the 40 image pairs, and for each pair, we set the SIFT distance ratio threshold t to 1.5, 1.3 and 1.0, respectively, as in [9]. The cumulative distribution function of original correct match percentage is shown in Fig. 8a. The initial average precision of all image pairs is 69.58%, and nearly 30 percent of the training sets have correct match percentage below 50%. Fig. 8b presents the cumulative distribution of the number of point matches contained in the experimental image pairs. We see that most of the image pairs have large scale of point matches (i.e. in the order of 1000's).

The precision–recall pairs on this dataset are summarized in Fig. 8. The average precision–recall pairs are (95.49%, 97.55%), (97.95%, 96.93%), (93.95%, 62.69%) and (98.57%, 97.78%) for RANSAC, MLESAC, ICF and SparseVFC, respectively. Here we choose homography as the geometry model in RANSAC and MLESAC. Note that the performances of VFC, FastVFC and SparseVFC are quite close, thus we omit the results of VFC and FastVFC in the figure for clarity. From the result, we see that ICF usually has high precision or recall, but not simultaneously. MLESAC performs a little better

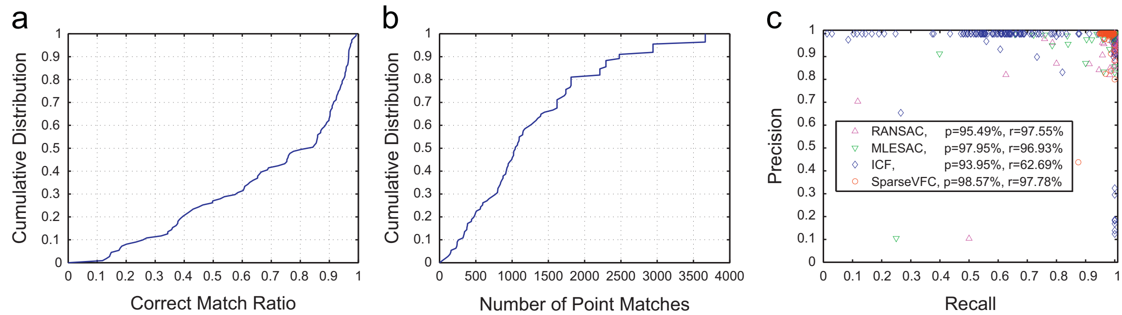

Fig. 8. Experimental results on the dataset of Mikolajczyk et al. (a) Cumulative distribution function of original correct match percentage. (b) Cumulative distribution function of number of point matches in the image pairs. (c) Precision–recall statistics for RANSAC, MLESAC, ICF and SparseVFC on the dataset of Mikolajczyk. Our method (red circles, upper right corner) has the best precision and recall overall. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

than RANSAC, and they both achieve quite satisfactory performance. This can be explained by the lack of complex constraints between the elements of the homography matrix. Our method has good performance in dealing with the mismatch removal problem, and it has the best trade-off between precision and recall compared to the other three methods.

We compare the approximate accuracy and time efficiency of SparseVFC to VFC and FastVFC in Table 4. As shown in it, both FastVFC and SparseVFC approximate VFC quite well, especially our method SparseVFC. Moreover, SparseVFC achieves a significant speedup with respect to the original VFC algorithm, of about 300 times on average. The run-time of RANSAC is also presented in Table 4, and we see that SparseVFC is much more efficient than RANSAC. To prevent the efficiency of RANSAC from decreasing drastically, usually a maximum sampling number is preset in the literature. We set the maximum sampling number to 5000.

The influence of different choices of basis functions for sparse approximation is also tested on this dataset. Besides the random sampling, we consider three other different methods: (i) simply use $\sqrt{M}\times\sqrt{M}$ uniform grid points over the bounded input space; (ii) find M clustering center of the training inputs via k-means clustering algorithm; (iii) pick M basis functions minimizing the residuals via sparse greedy matrix approximation [17]. The average precision–recall pairs of these three methods are (98.58%, 97.79%), (98.57%, 97.75%) and (98.58%, 97.79%), respectively. We see that all the three approximation methods produce almost the same result as the random sampling method. Therefore, for the mismatch removal problem, it does not seem that selecting the "optimal" subset using sophisticated methods improves the performance compared to a random subset. However, in the interests of computational efficiency, we may be better off simply choosing a random subset of the training data.

Next, we study on using different kernel functions for robust learning. Two additional matrix-valued kernels are tested, such as a decomposable kernel [8]: $\boldsymbol{\Gamma}(\mathbf{x}_i,\mathbf{x}_j)=e^{-\beta\|\mathbf{x}_i-\mathbf{x}_j\|^2}\mathbf{A}$ with $\mathbf{A}=\omega\mathbf{1}_{D\times D}+(1-\omega D)\mathbf{I}_{D\times D}$, and a convex combination of the divergence-free kernel $\boldsymbol{\Gamma}_{df}$ and curl-free kernel $\boldsymbol{\Gamma}_{cf}$. Note that the form kernel will degenerate into a diagonal kernel when we set $\omega=0$. In our evaluation, the combination coefficients in these two kernels are selected via cross-validation. The results are summarized in Table 5. We see that SparseVFC approximate VFC quite well under different matrix-valued kernels. Moreover, using a diagonal decomposable kernel is adequate for the mismatch removal problem; it can reduce the complexity of the linear system, i.e. Eq. (32), while with only a negligible decrease in accuracy.

Finally, we use this dataset to investigate the influence of the choice of M, the number of basis functions. Besides the default value 15, three additional values of M including 5, 10 and 20 are tested, and the results are summarized in Table 6. We see that

Table 4
Average precision–recall and run-time comparison of RANSAC, VFC, FastVFC and SparseVFC on the dataset of Mikolajczyk.

	RANSAC [57]	VFC [9]	FastVFC [9]	SparseVFC
(p, r)	(95.49, 97.55)	(98.57, 97.75)	(98.75, 96.71)	(98.57, 97.78)
t (ms)	3784	6085	402	21

Table 5
Performance comparison by using different matrix-valued kernels for robust learning. DK: decomposable kernel; DFK+CFK: combination of divergence-free and curl-free kernels.

	DK, $\omega=0$	DK, $\omega\neq0$	DFK+CFK
VFC [9]	(98.57, 97.75)	(98.66, 97.91)	(98.69, 97.65)
SparseVFC	(98.57, 97.78)	(98.66, 97.93)	(98.67, 97.71)

Table 6
Performance comparison by choosing different numbers of basis functions.

M	5	10	15	20
(p, r)	(98.10, 96.88)	(98.57, 97.73)	(98.57, 97.78)	(98.57, 97.79)
t (ms)	12	15	21	49

SparseVFC is not very sensitive to the choice of M, and even $M=5$ can achieve a satisfied performance. It should be noted that better approximate accuracy can be achieved by choosing M adaptively. For example, first compute the standard deviation of the estimated inliers σ^* (i.e. Eq. (35)) under different choices of M, and then choose the one with the smallest value of σ^*. However, such scenario would significantly increase the computational cost, since it requires to run the algorithm once for each choice of M. Therefore, fixing the value of M to a constant large enough, i.e. 15 in this paper, will achieve a good trade-off between the approximate accuracy and computational efficiency.

5.2.3. Results on 3D surface pairs

Since VFC is not influenced by the dimension of the input data, now we test the mismatch removal performance of SparseVFC on 3D surface pairs and compare it to the original algorithm. Here the parameters are set as the same values as in the 2D case.

The two surface pairs *Dino* and *Temple* are shown in Fig. 9. As shown, the capability of SparseVFC is not weakened in this case. For the *Dino* dataset, there are 325 initial correspondences with

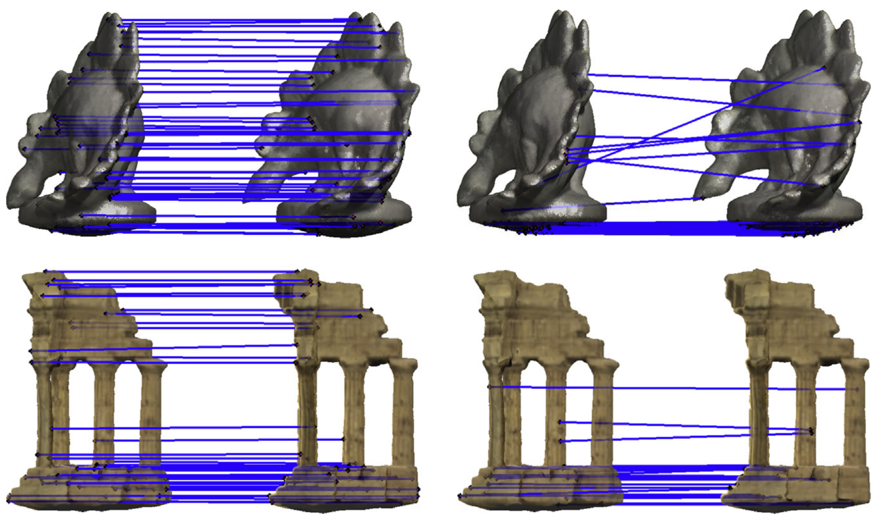

Fig. 9. Mismatch removal results of SparseVFC on two 3D surface pairs: *Dino* and *Temple*. There are 325 and 239 initial matches in these two pairs, respectively. For each group of results, the left pair denotes the identified putative correct matches (*Dino* 263, *Temple* 216), and the right pair denotes the removed putative mismatches (*Dino* 62, *Temple* 23). For visibility, at most 50 randomly selected correspondences are presented.

264 correct matches and 61 mismatches; after using the SparseVFC to remove mismatches, 263 matches are preserved and all of which are correct matches. That is to say, all the false matches are eliminated while discarding only 1 correct match. For the *Temple* dataset, there are 239 initial correspondences with 215 correct matches and 24 mismatches; after using the SparseVFC to remove mismatches, 216 matches are preserved, in which 214 are correct matches—that is, 22 of 24 false matches are eliminated while discarding only 1 correct match. Performance comparison are presented in Table 7, we see that both FastVFC and SparseVFC have a good approximation to the original VFC algorithm. Therefore, the sparse approximation used in our method works quite well, not only in the 2D case, but also in the 3D case.

Table 7
Performance comparison of VFC, FastVFC and SparseVFC on two 3D surface pairs: *Dino* and *Temple*. The initial correct match percentages are about 81.23% and 89.96%, respectively.

	VFC [9]	FastVFC [9]	SparseVFC
Dino	(98.87, 99.62)	(100.00, 99.62)	(100.00, 99.62)
Temple	(99.07, 99.53)	(99.07, 99.53)	(99.07, 99.53)

6. Conclusion

In this paper, we study a sparse approximation algorithm for vector-valued regularized least-squares. It searches a suboptimal solution under an assumption that the solution space can be represented sparsely with much less basis function. The time and space complexities of our algorithm are both linear in the scale of training samples, and the number of basis functions is manually assigned. We also present a new robust vector field learning method called *SparseVFC*, which is a sparse approximation to VFC, and apply it to solving the mismatch removal problem. The quantitative results on various experimental data demonstrate that the sparse approximation leads to a vast increase in speed with negligible decrease in performance, and it outperforms several state-of-the-art methods such as RANSAC from the perspective of mismatch removal.

Our sparse approximation addresses a generic problem for vector field learning, and it can be applied to other methodologies which can be converted into the vector field learning problem, for example, the Coherent Point Drift algorithm [41] designed for point registration. The sparse approximation of the Coherent Point Drift algorithm is described in detail in Appendix. Besides, our approach also has some limitations, for example, the sparse approximation is validated only in the case of ℓ_2 loss. Our future work shall focus on (i) determining the basis number automatically and efficiently and (ii) validating the sparse approximation under different types of loss functions.

Conflict of Interest

None

Acknowledgements

The authors gratefully acknowledge the financial supports from the National Natural Science Foundation of China (Nos. 61273279, 60903096, and 61222308) and China Scholarship Council (No. 201206160008). Xiang Bai was supported by the Program for New Century Excellent Talents in University. Zhuowen Tu was supported by NSF award IIS-0844566 and NSF award IIS-1216528.

Appendix A. Sparse approximation for coherent point drift

The Coherent Point Drift (CPD) algorithm [41] considers alignment of two point sets as a probability density estimation problem. Given two point sets $\mathbf{X}_{LD\times 1} = (\mathbf{x}_1^{\mathrm{T}}, \ldots, \mathbf{x}_L^{\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{Y}_{KD\times 1} = (\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_K^{\mathrm{T}})^{\mathrm{T}}$ with D being the data point's dimension, the algorithm assumes the points in $\mathbf{Y}$ are the Gaussian mixture model (GMM) centroids, and the points in $\mathbf{X}$ are the data points generated by the GMM. It then estimates the transformation T which yields the best alignment between the transformed GMM centroids and the data points by maximizing a likelihood. There the transformation T is defined as an initial position plus a displacement function $\mathbf{f}$: $T(\mathbf{y}) = \mathbf{y} + \mathbf{f}(\mathbf{y})$, where $\mathbf{f}$ is assumed to come from an RKHS $\mathcal{H}$ and hence has the form of Eq. (8).

Similar to our SparseVFC algorithm, the CPD algorithm also adopts an iterative EM algorithm to alternatively recover the spatial transformation and update the point correspondence.

And in the *M-step*, for recovering the displacement function $\mathbf{f}$, it needs to solve a linear system similar to Eq. (10), which takes up most of the run-time and memory requirements of the algorithm. The sparse approximation again could be used here to reduce the time and space complexity.

In the iteration of CPD algorithm, the displacement function $\mathbf{f}$ can be estimated from minimizing an energy function as

$$\mathcal{E}(\mathbf{f}) = \frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{k=1}^{K} p_{kl}\|\mathbf{x}_l-\mathbf{y}_k-\mathbf{f}(\mathbf{y}_k)\|^2+\frac{\lambda}{2}\|\mathbf{f}\|_{\mathcal{H}}^2, \tag{36}$$

where p_{kl} is the posterior probability of the correspondence between two points $\mathbf{y}_k$ and $\mathbf{x}_l$, and σ is the standard deviation of the GMM components.

Using the sparse approximation, we search a suboptimal $\mathbf{f}^s$ with the form Eq. (12). Due to the choice of a diagonal decomposable kernel $\mathbf{\Gamma}(\mathbf{y}_i,\mathbf{y}_j)=e^{-\beta\|\mathbf{y}_i-\mathbf{y}_j\|^2}\mathbf{I}$, Eq. (36) becomes

$$\mathcal{E}(\mathbf{C}) = \frac{1}{2\sigma^2}\sum_{l=1}^{L}\|(\mathrm{diag}(\mathbf{P}_{\cdot,l})\otimes\mathbf{I}_{D\times D})^{1/2}(\mathbf{X}_l^K-\mathbf{Y}-\tilde{\mathbf{U}}\mathbf{C})\|^2+\frac{\lambda}{2}\mathbf{C}^{\mathrm{T}}\tilde{\mathbf{\Gamma}}\mathbf{C}, \tag{37}$$

where $\mathbf{P}=\{p_{kl}\}$ and $\mathbf{P}_{\cdot,l}$ denotes the l-column of $\mathbf{P}$, kernel matrix $\tilde{\mathbf{\Gamma}}$ is a $M\times M$ block matrix with the (i,j)–th block $\mathbf{\Gamma}(\tilde{\mathbf{y}}_i,\tilde{\mathbf{y}}_j)$, $\tilde{\mathbf{U}}$ is a $K\times M$ block matrix with the (i,j)–th block $\mathbf{\Gamma}(\mathbf{y}_i,\tilde{\mathbf{y}}_j)$, $\mathbf{X}_l^K=(\mathbf{x}_l;\ldots;\mathbf{x}_l)$ is a $KD\times 1$ dimensional vector, and $\mathbf{C}=(\mathbf{c}_1;\ldots;\mathbf{c}_M)$ is the coefficient vector.

Taking the derivative of Eq. (37) with respect to the coefficient vector $\mathbf{C}$ and setting it to zero, we obtain a linear system

$$(\mathbf{U}^{\mathrm{T}}\mathrm{diag}(\mathbf{P1})\mathbf{U}+\lambda\sigma^2\mathbf{\Gamma})\tilde{\mathbf{C}}=\mathbf{U}^{\mathrm{T}}\mathbf{P}\tilde{\mathbf{X}}-\mathbf{U}^{\mathrm{T}}\mathrm{diag}(\mathbf{P1})\tilde{\mathbf{Y}}, \tag{38}$$

where the kernel matrix $\mathbf{\Gamma}\in\mathbb{R}^{M\times M}$ and $\mathbf{\Gamma}_{ij}=e^{-\beta\|\tilde{\mathbf{y}}_i-\tilde{\mathbf{y}}_j\|^2}$, $\mathbf{U}\in\mathbb{R}^{K\times M}$ and $\mathbf{U}_{ij}=e^{-\beta\|\mathbf{y}_i-\tilde{\mathbf{y}}_j\|^2}$, $\mathbf{1}$ is a column vector of all ones, $\tilde{\mathbf{C}}=(\mathbf{c}_1,\ldots,\mathbf{c}_M)^{\mathrm{T}}$, $\tilde{\mathbf{X}}=(\mathbf{x}_1,\ldots,\mathbf{x}_L)^{\mathrm{T}}$ and $\tilde{\mathbf{Y}}=(\mathbf{y}_1,\ldots,\mathbf{y}_K)^{\mathrm{T}}$. Here M is the number of bases.

Thus we obtain a suboptimal solution from the coefficient matrix $\tilde{\mathbf{C}}$. This corresponds to the optimal solution $\mathbf{f}^o$, i.e. Eq. (8), with the coefficient matrix $\tilde{\mathbf{C}}$ determined by the linear system [41]

$$(\mathbf{\Gamma}+\lambda\sigma^2\mathrm{diag}(\mathbf{P1})^{-1})\tilde{\mathbf{C}}=\mathrm{diag}(\mathbf{P1})^{-1}\mathbf{PX}-\mathbf{Y}, \tag{39}$$

where $\mathbf{\Gamma}\in\mathbb{R}^{K\times K}$ with $\mathbf{\Gamma}_{ij}=e^{-\beta\|\mathbf{y}_i-\mathbf{y}_j\|^2}$, and $\tilde{\mathbf{C}}=(\mathbf{c}_1,\ldots,\mathbf{c}_K)^{\mathrm{T}}$.

Since it is a sparse approximation of the CPD algorithm, we name this approach SparseCPD. We simply summarize the SparseCPD algorithm in Algorithm 2.

Algorithm 2. The SparseCPD Algorithm.

Input: Two point sets $\mathbf{X}$ and $\mathbf{Y}$
Output: Transformation $\mathcal{T}$
Parameter initialization, including M and all the parameters in CPD
repeat
  E–step :
  Update the point correspondence
  M–step :
  Update the coefficient vector $\tilde{\mathbf{C}}$ by solving linear system (38)
  Update other parameters in CPD
until converges;
The transformation $\mathcal{T}$ is determined according to the coefficient vector $\tilde{\mathbf{C}}$.

References

[1] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer Verlag, 1995.
[2] T. Evgeniou, M. Pontil, T. Poggio, Regularization networks and support vector machines, Computational Mathematics 13 (2000) 1–50.
[3] F. Girosi, M. Jones, T. Poggio, Regularization theory and neural networks architectures, Neural Computation 7 (1995) 219–269.
[4] N. Aronszajn, Theory of reproducing kernels, Transactions of the American Mathematical Society 68 (1950) 337–404.
[5] R. Rifkin, G. Yeo, T. Poggio, Regularized least-squares classification, in: Advances in Learning Theory: Methods, Model and Applications, 2003.
[6] C. Saunders, A. Gammermann, V. Vovk, Ridge regression learning algorithm in dual variables, in: Proceedings of International Conference on Machine Learning, 1998, pp. 515–521.
[7] J.A.K. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Processing Letters 9 (1999) 293–300.
[8] L. Baldassarre, L. Rosasco, A. Barla, A. Verri, Multi-Output Learning via Spectral Filtering, Technical Report, MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, 2011.
[9] J. Zhao, J. Ma, J. Tian, J. Ma, D. Zhang, A robust method for vector field learning with application to mismatch removing, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 2977–2984.
[10] C.A. Micchelli, M. Pontil, On learning vector-valued functions, Neural Computation 17 (2005) 177–204.
[11] E. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies, IEEE Transactions on Information Theory 52 (2005) 5406–5425.
[12] S.S. Keerthi, O. Chapelle, D. DeCoste, Building support vector machines with reduced classifier complexity, Journal of Machine Learning Research 7 (2006) 1493–1515.
[13] M. Wu, B. Scholkopf, G. Bakir, A direct method for building sparse kernel learning algorithms, Journal of Machine Learning Research 7 (2006) 603–624.
[14] P. Sun, X. Yao, Sparse approximation through boosting for learning large scale kernel machines, IEEE Transactions on Neural Networks 21 (2010) 883–894.
[15] E. Tsivtsivadze, T. Pahikkala, A. Airola, J. Boberg, T. Salakoski, A sparse regularized least-squares preference learning algorithm, in: Proceedings of the Conference on Tenth Scandinavian Conference on Artificial Intelligence, 2008, pp. 76–83.
[16] C.K.I. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: Advances in Neural Information Processing Systems, 2000, pp. 682–688.
[17] A.J. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in: Proceedings of International Conference on Machine Learning, 2000, pp. 911–918.
[18] Y.-J. Lee, O. Mangasarian, RSVM: reduced support vector machines, in: Proceedings of the SIAM International Conference on Data Mining, 2001.
[19] S. Fine, K. Scheinberg, Efficient SVM training using low-rank kernel representations, Journal of Machine Learning Research 2 (2001) 243–264.
[20] F. Girosi, An equivalence between sparse approximation and support vector machines, Neural Computation 10 (1998) 1455–1480.
[21] S. Chen, D. Donoho, M. Saunders, Atomic decomposition by basis pursuit, SIAM Journal of Scientific Computing 20 (1999) 33–61.
[22] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky, An interior-point method for large-scale ℓ_1–regularized least squares, IEEE Journal of Selected Topics in Signal Processing 1 (2007) 606–617.
[23] M.N. Gibbs, D.J.C. MacKay, Efficient Implementation of Gaussian Processes, Technical Report, Department of Physics, Cavendish Laboratory, Cambridge University, Cambridge, 1997.
[24] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods: Support Vector Learning, 1999, pp. 185–208.
[25] S.S. Keerthi, K. Duan, S. Shevade, A. Poo, A fast dual algorithm for kernel logistic regression, Machine Learning 61 (2005) 151–165.
[26] L. Bottou, O. Bousquet, The tradeoffs of large scale learning, in: Advances in Neural Information Processing Systems, 2008, pp. 161–168.
[27] M. Zinkevich, M. Weimer, A. Smola, L. Li, Parallelized stochastic gradient descent, in: Advances in Neural Information Processing Systems, 2010, pp. 2595–2603.
[28] Y. Tsuruoka, J. Tsujii, S. Ananiadou, Stochastic gradient descent training for $L1$-regularized log-linear models with cumulative penalty, in: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2009, pp. 477–485.
[29] C. Carmeli, E. De Vito, A. Toigo, Vector valued reproducing kernel hilbert spaces of integrable functions and mercer theorem, Analysis and Applications 4 (2006) 377–408.
[30] T. Poggio, F. Girosi, Networks for approximation and learning, Proceedings of the IEEE 78 (1990) 1481–1497.
[31] J. Quiñonero-Candela, C.E. Ramussen, C.K.I. Williams, Approximation methods for gaussian process regression, in: Large-Scale Kernel Machines, 2007, pp. 203–223.
[32] A.R. Barron, Universal approximation bounds for superpositions of a sigmoidal function, IEEE Transactions on Information Theory 39 (1993) 930–945.
[33] L.K. Jones, A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training, Annals of Statistics 20 (1992) 608–613.
[34] V. Kůrková, High-dimensional approximation by neural networks, in: Advances in Learning Theory: Methods, Models and Applications, 2003, pp. 69–88.
[35] V. Kůrková, M. Sanguineti, Comparison of worst case errors in linear and neural network approximation, IEEE Transactions on Information Theory 48 (2002) 264–275.

[36] V. Kůrková, M. Sanguineti, Learning with generalization capability by kernel methods of bounded complexity, Journal of Complexity 21 (2005) 350–367.
[37] S. Yu, V. Tresp, K. Yu, Robust multi-task learning with t-processes, in: Proceedings of International Conference on Machine Learning, 2007, pp. 1103–1110.
[38] S. Zhu, K. Yu, Y. Gong, Predictive matrix-variate t models, in: Advances in Neural Information Processing Systems, 2008, pp. 1721–1728.
[39] P.J. Huber, Robust Statistics, John Wiley & Sons, New York, 1981.
[40] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[41] A. Myronenko, X. Song, Point set registration: coherent point drift, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2010) 2262–2275.
[42] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed., Cambridge University Press, Cambridge, 2003.
[43] D. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2004) 91–110.
[44] P.H.S. Torr, A. Zisserman, MLESAC: a new robust estimator with application to estimating image geometry, Computer Vision and Image Understanding 78 (2000) 138–156.
[45] J.H. Kim, J.H. Han, Outlier correction from uncalibrated image sequence using the triangulation method, Pattern Recognition 39 (2006) 394–404.
[46] L. Goshen, I. Shimshoni, Guided sampling via weak motion models and outlier sample generation for epipolar geometry estimation, International Journal of Computer Vision 80 (2008) 275–288.
[47] X. Li, Z. Hu, Rejecting mismatches by correspondence function, International Journal of Computer Vision 89 (2010) 1–17.
[48] J. Ma, J. Zhao, Y. Zhou, J. Tian, Mismatch removal via coherent spatial mapping, in: Proceedings of IEEE International Conference on Image Processing, 2012, pp. 1–4.
[49] J. Ma, J. Zhao, J. Tian, Z. Tu, A. Yuille, Robust estimation of nonrigid transformation for point set registration, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2013.
[50] J. Barron, D. Fleet, S. Beauchemin, Performance of optical flow techniques, International Journal of Computer Vision 12 (1994) 43–77.
[51] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. van Gool, A comparison of affine region detectors, International Journal of Computer Vision 65 (2005) 43–72.
[52] T. Tuytelaars, L. van Gool, Matching widely separated views based on affine invariant regions, International Journal of Computer Vision 59 (2004) 61–85.
[53] A. Vedaldi, B. Fulkerson, VLFeat—an open and portable library of computer vision algorithms, in: Proceedings of the 18th Annual ACM International Conference on Multimedia, 2010, pp. 1469–1472.
[54] A. Zaharescu, E. Boyer, K. Varanasi, R. Horaud, Surface feature detection and description with applications to mesh matching, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 373–380.
[55] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 886–893.
[56] I. Macêdo, R. Castro, Learning Divergence-Free and Curl-Free Vector Fields with Matrix-Valued Kernels, Technical Report, Instituto Nacional de Matematica Pura e Aplicada, Brasil, 2008.
[57] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography, Communications of the ACM 24 (1981) 381–395.

Jiayi Ma received the B.S. degree from the Department of Mathematics, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2008. He is currently a Ph. D. student of the Institute for Pattern Recognition and Artificial Intelligence, HUST. His research interests are in the areas of computer vision and machine learning.

Ji Zhao received his BS degree in 2005 and his MS degree in 2008. Currently he is a PhD candidate at the Institute for Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology (HUST), China. His research area is computer vision.

Jinwen Tian received his PhD degree in pattern recognition and intelligent systems in 1998 from Huazhong University of Science and Technology (HUST), China. He is a professor and PhD supervisor of pattern recognition and artificial intelligence at HUST. His main research topics are remote sensing image analysis, wavelet analysis, image compression, computer vision, and fractal geometry.

Xiang Bai received the B.S., M.S., and Ph.D. degrees from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003, 2005, and 2009, respectively, all in electronics and information engineering.

He is currently an Associate Professor with the Department of Electronics and Information Engineering, HUST. From January 2006 to May 2007, he was with the Department of Computer Science and Information, Temple University, Philadelphia, PA. From October 2007 to October 2008, he was with the University of California, Los Angeles, as a joint Ph.D. student. His research interests include computer graphics, computer vision, and pattern recognition.

Zhuowen Tu received the M.E. degree from Tsinghua University, Beijing, China, and the Ph.D. degree from Ohio State University, Columbus.

He is currently an Assistant Professor with the Laboratory of Neuro Imaging (LONI), Department of Neurology, with a joint appointment in the Department of Computer Science, University of California, Los Angeles (UCLA). He is also affiliated with the UCLA Bioengineering Interdepartmental Program, the UCLA Bioinformatics Program, and with Microsoft Research Asia, Beijing, China. Before joining LONI, he was a Member of Technical Staff with Siemens Corporate Research and a Postdoctoral Fellow with the Department of Statistics, UCLA.

Prof. Tu was a recipient of the David Marr Prize in 2003.